Micro Fiche Scan

Name of device(s) tested:
    KDJ11-B/BF
Test description:
    KDJ11-B CLUSTER DIAGNOSTIC
MAINDEC Number or Package Identifier (after SEP 1977):
    COKDAF0
Fiche Document Part Number:
    AH-T854F-MC
Fiche preparation date unknown, using copyright year:
    1986
Image resolution:
    1-bit black&white, compressed for minimal file size

```
1                                                                   .REM &
2
3
4                                                   IDENTIFICATION
5                                                   ---- ---------
6
7                                           PRODUCT CODE:   AC-T853F-MC
8
9                                           PRODUCT NAME:   COKDAF0 KDJ11-B CLUSTER DIAG.
10
11                                          PRODUCT DATE:   MARCH, 1986
12
13                                          MAINTAINER:     DIAGNOSTIC ENGINEERING
14
15
16
17
18
19
20                  THE INFORMATION IN THIS DOCUMENT IS  SUBJECT  TO  CHANGE WITHOUT
21                  NOTICE  AND  SHOULD  NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL
22                  EQUIPMENT CORPORATION.  DIGITAL EQUIPMENT CORPORATION ASSUMES NO
23                  RESPONSIBILITY FOR ANY ERRORS THAT MAY APPEAR IN THIS DOCUMENT.
24
25                  NO RESPONSIBILITY IS ASSUMED  FOR  THE  USE  OR  RELIABILITY  OF
26                  SOFTWARE ON EQUIPMENT THAT IS NOT  SUPPLIED  BY  DIGITAL  OR ITS
27                  AFFILIATED COMPANIES.
28
29                  COPYRIGHT (C) 1984,1985,1986 BY DIGITAL EQUIPMENT CORPORATION
30
31
32                  THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION:
33
34                          DIGITAL         PDP             UNIBUS          MASSBUS
35                          DEC             DECUS           DECTAPE
36
37
38
39
40
41
42
```

TABLE OF CONTENTS

1. PROGRAM ABSRACT

2. SYSTEM REQUIREMENTS

3. LOADING AND STARTING PROCEDURES

4. SPECIAL ENVIRONMENTS

5. PROGRAM OPTIONS

6. EXECUTION TIMES

7. ERROR INFORMATION

8. EXAMPLES

9. PROGRAM DESCRIPTION

   9.1 J11 CODE
   9.2 CACHE CODE
   9.3 ON-BOARD ROM CODE
   9.4 LINE TIME CLOCK CODE
   9.5 SERIAL LINE UNIT CODE
   9.6 Q22BE CODE
   9.7 LIST of SUBTESTS showing CACHE/APT dependency

10. PROGRAM UPDATES AND MODIFICATIONS

```
 79
 80
 81
 82
 83
 84
 85
 86
 87
 88
 89
 90
 91
 92
 93
 94
 95
 96
 97
 98
 99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
```

1. PROGRAM ABSTRACT

**** see special instructions for CACHE testing under APT, ****
**** section 9.2, and EEROM testing under APT, section 9.3 ***

This program tests out KDJ11-B CPU board, including the J11 chip set, on board cache, on board ROM's, including 16 bit and the 8 bit EEROM, serial line unit, and line time clocks.

The KDJ11 B is a PDP-11 CPU that incorporates the J11 chip set as the heart of the processor. It is a quad height Q22 bus module. It has on-board cache, some of the functionality of the cache is hidden inside the J11 and the rest of the functions implemented in two on-board gate arrays. The storage capacity of the cache is 4K bytes of RAM, called data RAM's. The cache is implemented as a direct mapped cache with address bits 21 through 13 stored in a different set of RAM's called tag store.
The KDJ11-B also has two on-board ROM's. One of them, the 16-bit addressable ROM, contains the self-test and the boot codes. The other ROM, the 8-bit addressable one, contains the base area with hardware selection parameters, optional bootstraps, optional UFD (User Friendly Diagnostic) system desctiption area, and optional foreign language text.
The Serial Line Unit is implemented thru a Dlart chip which provides the standard console interface to the CPU. It has internal loop back mode and provides with three clock lines: 800HZ, 60HZ, and 50HZ.
The line time clock functions are implemented using those three lines and the BEVENT line. The line clock status register is implemented in one of the gate arrays.

2. SYSTEM REQUIREMENTS

Hardware Requirements

To run successfully the diagnostic needs:

1. KDJ11-B CPU module
2. console terminal
3. at least 28K of memory

In DVT, and stage one manufacturing (module assembly) the Q22 Bus exerciser is needed to check Q22 Bus logic.

```
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
```

3. LOADING AND STARTING PROCEDURES

To start up this program:

1. Boot XXDP+

2. Type "R NAME", where name is the name of the BIN or BIC file for this program.

The starting address of the program is 200.
Note: if trying to restart the program in an arbitrary place after HALT on Break the following registers should be set up:

17777572=0	to disable memmory management
17777520=1000	to clear diagnostic mode (bit 8), but still save HALT on Break
17777746=400	to flush the cache

4. SPECIAL ENVIRONMENTS

The program is APT compatible. It can also be run under the UFD monitor. In those cases none of the standard error printouts occur. Refer to corresponding documents on running procedures in APT and under UFD monitor.

**** see special instructions for CACHE testing under APT, **** **** section 9.2, and EEROM testing under APT, section 9.3 ****

5. PROGRAM OPTIONS

The Q22 Bus Exerciser is utilized if it is present in the system and the diagnostic is not running in UFD mode.
Standard capabilities of looping on test and on error are provided.
In order to run the extensive cache data RAM test bit 7 has to be set in the software switch register. To run the extensive cache tag RAM test bit 6 has to be set in the software switch register.

SWITCH REGISTER SELECTION:

	BIT NUMBER	USE
	15	HALT ON ERROR
	14	LOOP ON PRESENT TEST
	13	INHIBIT ERROR TYPEOUTS
***	12	EEROM subtest RUN switch
	11	INHIBIT ITERATIONS
	10	BELL ON ERROR
	9	LOOP ON ERROR
	8	LOOP ON TEST IN SWR<5-0>
	7	DO EXTENSIVE DATA RAM TEST
	6	DO EXTENSIVE TAG RAM TEST
	5-0	Subtest number to loop on (BIT 8)

*** running with this test will change all but the first 109 bytes of EEROM data! be SURE to read instructions in sect. 9.3

6. EXECUTION TIMES

Without extensive RAM tests, the diagnostic runs in under 15 seconds.
With them, it takes about 2 minutes.

In addition, when the EEROM test bit is on, pass 1 takes an additional amount of time of 1 minute for 15MHZ J-11, to about 40 sec. for the 20MHZ. version. These test times are for the 2K EEROM, for 8K parts, multiply by 4. If running under APT, the system manager must be sure the first pass run time allows adequate time for completion of this test!

7. ERROR INFORMATION

In the case of errors, a failing PC and test numbers are given.
Where it is possible, expected and received data are given.
For an example, see section 8.

8. EXAMPLES

After booting XXDP+ and starting the program, the following will appear on the terminal:

```
* KDJ11-B CPU DIAGNOSTIC - COKDAFO *

SWR = XXXXXX    NEW =
```

where XXXXXX correspond to present software switch register setting.
After "NEW" an operator can do one of the following:
1) type in a new software switch register setting followed by carrige return or
2) just type in carriage return in which case the software register will remain unchanged.

Example of error printout:

```
        ERROR IN TAG STORE
         TEST   ERROR   ADDRESS ADDRESS
           #      PC     <21-16> <15-0>
          27    105620  66600   000000
```

Note: this may not correspond to the actual Program Counter.

9. PROGRAM DESCRIPTION

9.1 J11 CODE

This portion of the code tests out the J11 chip set. It is broken into 3 pieces: CPU tests, which verify different instructions in different modes and different trap conditions; MMU tests, which verify different functions of MMU; and FFP tests, which do different floating point instructions.
This portion of the code have been written in close relationship with the J11 microcode. Therefore, even though not all possible instructions in all possible addressing mode have been tested, an attempt has been made to exercise all of the microcode.

9.2 CACHE CODE

This portion of the diagnostic verifies all cache functions.
Data and tag RAM's tests are also included.

****
Note: in order to run extensive cache RAM's tests the corresponding bits in the software switch register should be set. See section 5.
----In particular, the number in $USWR (sware reg #2) should equal the "first pass run time" set up at installation---- (the exception is when $USWR is set to ZERO, no condition applies)
****

TESTING CACHE FUNCTIONS UNDER APT

The testing of Cache related functions under APT is facilitated by use of $USWR containing a number which will indicate the number of passes with inclusive cache tests as follows:

(all numbers in decimal, $USWR loaded by APT manager)

default setting	$USWR = 0	one pass with all subtests followed by continuous testing of non-cache dependent subtests until APT Break is encountered (APT system must not BREAK in for this "First Pass Run Time", now assumed to be approx 16 sec.)
	$USWR < 16	only non-cache dependent tests will be run APT may break anytime
	$USWR = 16	same as $USWR = 0
	$USWR > 16	$USWR is divided by 16 (pass time) to get a

number of passes which include all subtests
(including cache) After this number of passes,
continuous testing with only non cache depend
ent sub-tests is continued during which time
APT may break with no bad effects.

In this case, the APT setup must not attempt to break for a length of time equal to $USWR contents, in seconds. The nominal 16 seconds per pass, and the actual pass time of about 10 seconds (which changes as tests may be added) allow a safty margin adding additional tests. The actual number of passes done will be equal to $USWR divided by 16, (if not 0, or < 16).

!!!!!!!!!!!!!!

***** The guarrantee that APT will not attempt to break into the diagnostic for a given number of seconds is provided by the APT system manager loading the variable $PASTM with the same number $USWR received.
If this is not done, the message "hung diagnostic" will likely be received*******

!!!!!!!!!!!!!!

9.3 ON-BOARD ROM AND EEROM CODE

These tests verify the checksums of the 16-bit ROM and the base area of the 8-bit EEROM, and a write and read of 1's and 0's to each location of the EEROM. The EEROM may be affected by multiple writes, (>10,000 cycles) so it is advisable to select this test with care.

The testing of the on-board EEROM is accomplished by the setting of bit 12 in the software switch register, which will see that the EEROM test is run once. (Pass 1) It should be noted that this bit may have another use in the SYSMAC macro $EOP, but is not so used within this program. The BCSR is also set for Halt on Break on during this test, as a troubleshooting aid in stand-alone modes.

*** WHEN RUNNING THIS TEST UNDER APT ****

After loading and starting the program with EEROM test switch on, APT must not (break) interrupt the test until the time given in section 6, EXECUTION TIMES, has passsed.

This test should never be "scripted" in a way which would cause the test to be run multiple times. It should be run instead once at the begining or end of a script.

****

!! The contents of the EEROM are lost, except for the 109 (decimal) byte
!! representing the SETUP area, which are restored at the end of the EEROM
!! subtest. If the test is interrupted before these locations are re-
!! stored, unpredictable results may occur.

9.4 LINE TIME CLOCKS CODE

This part of the program verifies the functions of all 4 clock lines: 3 of them from the serial line ch'p (50HZ, 60HZ, 800HZ) and BEVENT line.
Note: in UFD mode only functions corresponding to Boot Rom selection are checked.

9.5 SERIAL LINE UNIT CODE

These tests verify the functionality of the SLU chip utilizing the maintenance mode of the chip.

9.6 Q22BE CODE

These tests verify interrupt arbitration of the KDJ11-B, DMA protocol, and cache functions related to the DMA activities, includ'ng PMG counter.

9.7 LIST OF SUBTESTS PERFORMED

The following list represents the sequential order of subtests performed in COKDAFO..... subtests which are subject to the APT qualifications of section 9.2 are indicated by a Cache-APT label.

TEST NO.	TEST NAME/FUNCTION	APT-CACHE sel	NOTES.
test 1	base instruction set tests	.....	
test 2	check (cache) register access	yes	
test 3	CCR register bit test	yes	
test 4	force miss test	yes	
test 5	hit/miss register test part 1	yes	
test 6	hit/miss register test	yes	
test 7	byte allocation test	yes	
test 10	PDR bit 15 (bypass) test	yes	
test 11	flush cache test	yes	
test 12	unconditional bypass test	yes	
test 13	write wrong data parity test	yes	
test 14	write wrong tag parity	yes	
test 15	parity abort test	yes	
test 16	parity interrupt test	yes	
test 17	miscellaneous parity test	yes	
test 20	memory system error register test	yes	
test 21	check parity aborts blocked	yes	
test 22	multi-processing instruction	yes	
test 23	data store ram, tests	yes	

test 24	tag store ram tests	yes		
test 25	standalone mode tests	yes		
test 26	moving inversions test data RAM	yes		
test 27	moving inversions for tag store	yes		
test 30	PCR read/write bits	.....		
test 31	bcsr read/write bits	.....		
test 32	16 bit ROM checksum	.....	HOB on	
test 33	8 bit EEROM checksum test	.....	HOB on	
test 34	EEROM checkerboard mem test	.....	note 1.0	
test 35	LKS bit 7	.....		
test 36	LKS interrupt priority	.....		
test 37	line clock disable	.....		
test 40	unconditional clock interupts	.....		
test 41	resetting LKS	.....	only done pass 1	
test 42	line clock interrupts	yes	not CACHE dep. APT break sensitive	
test 43	maintenance register test	.....		
test 44	serial line unit registers	.....		
test 45	XCSR bit 7	.....		
test 46	RCSR bit 7 and XCSR bit 7	.....	done once in APT	
test 47	RESET and XCSR<2:0>	.....	first pass only	
test 50	RESET and interrupt enable	.....		
test 51	interrupt priority for SLU	.....	done once in APT	
test 52	Break condition	.....	done only once	
test 53	overrun condition	.....	done once in APT	
test 54	LED'S on test	.....	HOB on	
test 55	MEMORY mapping	not done chain mode, APT, other than pass 1		
test 56	wrong parity abort test	.....		
test 57	DMA TAG PARITY in standalone	yes	HOB disabled	
test 60	DMA tag parity w/o standalone	yes		
test 61	DMA write hit cycles	yes		
test 62	different levels of interrupts		not UFD, only if Q22BE found	
test 63	Arbitration bet PIRQ interrupt		not UFD, only if Q22BE found	
test 64	power down test		not UFD, only if Q22BE found	
test 65	arbitration between interrupt levels		not UFD, only if Q22BE found	
test 66	PMG counter	yes	not UFD, only if two Q22BE found	

as of july 1984, this is the list of subtests, any added subtests will require editing of this list....

note 1. This test will only be run in first pass, and then only if bit 12 is set in the SWR.... Running this test will DESTROY everything in the EEROM except the first 109 bytes..... UFD area, secondary boots, and local language area's would all need to be restored after running this test. The first pass run time must be adjusted accordingly when running under

```
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
```

10. PROGRAM UPDATES AND MODIFICATIONS

Version COKDADO 2 22-85 Howard L. Marshall
1. Changes made by Howard L. Marshall denoted by "$$$"
   in comment lines of added or changed code, as of 2-22-85

Added subroutine "DETFPA" (determine floating point accelerator)

This subroutine is called just prior to the error macro for all floating point errors. This subroutine prints 1 of 2 possible error messages. they are:

A.) ERROR DETECTED IN FLOATING POINT ACCELERATOR CHIP.

B.) ERROR DETECTED IN J11 FLOATING POINT PROCESSOR.

Of the 2 above error messages, the error message that is printed is determined by the state of the "FPA" flag, bit 8, in the maintenance register. this bit is set to a 1 if the floating point accelerator chip is installed in the cpu board. Otherwise, the bit is cleared which indicates that the floating point accelertor chip is not installed. Based solely on this determination, it is logical to assume that most, if not all floating point errors can be attributed to the FPA chip if the "FPA" bit is set or to the floating point processor within the j11 if the "FPA" bit is cleared.

Version COKDAE0 13-MAY-85 Howard L. Marshall
Changes made by Howard L. Marshall - denoted by "$$$"
Corrected error in 22-Bit MMU Tests (test #1) which used number 157776 as virtual address intended to address last word of RAM just below the I/O page. This virtual address in conjunction with a value of 177500 in KPAR6 actually maps to physical address 17767776 which is at 2046K (half-way into the I/O page). Changed that virtual address to 147776 which when used with the above value in KPAR6, maps to physical address 17757776 which is in the 2044K page, the last non I/O page address.

Modified test #53, Overrun condition test to allow this test to work properly when the console terminal is running at the minimum ORION baud rate, 300 baud. Changed wait argument from: #150000 to #175000.

Version COKDAF0 21-Mar-86 Jeffrey P. Belanger (JPB)
The latest version of the DCJ11-AB chip exposed an error in the diagnostic while testing a certain parameter with the ASH and ASHC instructions. This version corrects the problem. The contents of the following locations were changed from 177737 to 177761:
26506,27262, and 27516
This allows both processors to be tested without errors.

```
                        $SWR=167400
                        $SWRMK=300
                .TITLE  COKDAFO KDJ11 B CLUSTER DIAG.
                ;*COPYRIGHT (C) MAR 86
                ;*DIGITAL EQUIPMENT CORP.
                ;*MAYNARD, MASS. 01754
                ;*
                ;*PROGRAM BY DIAG. ENG.
                ;*
                ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                ;*PACKAGE (MAINDEC-11-DZQAC-C8), OCT, 1982.
                ;*
000001          $TN=1
                .SBTTL  OPERATIONAL SWITCH SETTINGS
                ;*
                ;*      SWITCH                  USE
                ;*      ---- -          ------------ ---- --
                ;*        15             HALT ON ERROR
                ;*        14             LOOP ON TEST
                ;*        13             INHIBIT ERROR TYPEOUTS
                ;*        11             INHIBIT ITERATIONS
                ;*        10             BELL ON ERROR
                ;*         9             LOOP ON ERROR
                ;*         8             LOOP ON TEST IN SWR<5:0>
                ;*         7             DO EXTENSIVE DATA RAM TEST
                ;*         6             DO EXTENSIVE TAG RAM TEST
                .SBTTL  MEMORY MANAGEMENT DEFINITIONS
                ;*KT11 VECTOR ADDRESS
000250          MMVEC=  250
                ;*KT11 STATUS REGISTER ADDRESSES
177572          SR0=    177572
177574          SR1=    177574
177576          SR2=    177576
172516          SR3=    172516
                ;*USER "I" PAGE DESCRIPTOR REGISTERS
177600          UIPDR0= 177600
177602          UIPDR1= 177602
177604          UIPDR2= 177604
177606          UIPDR3= 177606
177610          UIPDR4= 177610
177612          UIPDR5= 177612
177614          UIPDR6= 177614
177616          UIPDR7= 177616
                ;*USER "D" PAGE DESCRIPTOR REGISTORS
177620          UDPDR0= 177620
177622          UDPDR1= 177622
177624          UDPDR2= 177624
177626          UDPDR3= 177626
177630          UDPDR4= 177630
177632          UDPDR5= 177632
177634          UDPDR6= 177634
177636          UDPDR7= 177636
                ;*USER "I" PAGE ADDRESS REGISTERS
177640          UIPAR0= 177640
177642          UIPAR1= 177642
177644          UIPAR2= 177644
177646          UIPAR3= 177646
```

MEMORY MANAGEMENT DEFINITIONS

```
177650                  UIPAR4= 177650
177652                  UIPAR5= 177652
177654                  UIPAR6= 177654
177656                  UIPAR7= 177656
                        ;*USER "D" PAGE ADDRESS REGISTERS
177660                  UDPAR0= 177660
177662                  UDPAR1= 177662
177664                  UDPAR2= 177664
177666                  UDPAR3= 177666
177670                  UDPAR4= 177670
177672                  UDPAR5= 177672
177674                  UDPAR6= 177674
177676                  UDPAR7= 177676
                        ;*SUPERVISOR "I" PAGE DESCRIPTOR REGISTERS
172200                  SIPDR0= 172200
172202                  SIPDR1= 172202
172204                  SIPDR2= 172204
172206                  SIPDR3= 172206
172210                  SIPDR4= 172210
172212                  SIPDR5= 172212
172214                  SIPDR6= 172214
172216                  SIPDR7= 172216
                        ;*SUPERVISOR "D" PAGE DESCRIPTOR REGISTERS
172220                  SDPDR0= 172220
172222                  SDPDR1= 172222
172224                  SDPDR2= 172224
172226                  SDPDR3= 172226
172230                  SDPDR4= 172230
172232                  SDPDR5= 172232
172234                  SDPDR6= 172234
172236                  SDPDR7= 172236
                        ;*SUPERVISOR "I" PAGE ADDRESS REGISTERS
172240                  SIPAR0= 172240
172242                  SIPAR1= 172242
172244                  SIPAR2= 172244
172246                  SIPAR3= 172246
172250                  SIPAR4= 172250
172252                  SIPAR5= 172252
172254                  SIPAR6= 172254
172256                  SIPAR7= 172256
                        ;*SUPERVISOR "D" PAGE ADDRESS REGISTERS
172260                  SDPAR0= 172260
172262                  SDPAR1= 172262
172264                  SDPAR2= 172264
172266                  SDPAR3= 172266
172270                  SDPAR4= 172270
172272                  SDPAR5= 172272
172274                  SDPAR6= 172274
172276                  SDPAR7= 172276
                        ;*KERNEL "I" PAGE DESCRIPTOR REGISTERS
172300                  KIPDR0= 172300
172302                  KIPDR1= 172302
172304                  KIPDR2= 172304
172306                  KIPDR3= 172306
172310                  KIPDR4= 172310
172312                  KIPDR5= 172312
172314                  KIPDR6= 172314
```

MEMORY MANAGEMENT DEFINITIONS

```
172316                  KIPDR7= 172316
                        ;*KERNEL "D" PAGE DESCRIPTOR REGISTERS
172320                  KDPDR0= 172320
172322                  KDPDR1= 172322
172324                  KDPDR2= 172324
172326                  KDPDR3= 172326
172330                  KDPDR4= 172330
172332                  KDPDR5= 172332
172334                  KDPDR6= 172334
172336                  KDPDR7= 172336
                        ;*KERNEL "I" PAGE ADDRESS REGISTERS
172340                  KIPAR0= 172340
172342                  KIPAR1= 172342
172344                  KIPAR2= 172344
172346                  KIPAR3= 172346
172350                  KIPAR4= 172350
172352                  KIPAR5= 172352
172354                  KIPAR6= 172354
172356                  KIPAR7= 172356
                        ;*KERNEL "D" PAGE ADDRESS REGISTERS
172360                  KDPAR0= 172360
172362                  KDPAR1= 172362
172364                  KDPAR2= 172364
172366                  KDPAR3= 172366
172370                  KDPAR4= 172370
172372                  KDPAR5= 172372
172374                  KDPAR6= 172374
172376                  KDPAR7= 172376
   498                  .SBTTL  BASIC DEFINITIONS
                        ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
001100                  STACK=  1100
104000                  ERROR=  EMT             ;;BASIC DEFINITION OF ERROR CALL
000004                  SCOPE=  IOT             ;;BASIC DEFINITION OF SCOPE CALL
                        ;*MISCELLANEOUS DEFINITIONS
000011                  HT=     11              ;;CODE FOR HORIZONTAL TAB
000012                  LF=     12              ;;CODE FOR LINE FEED
000015                  CR=     15              ;;CODE FOR CARRIAGE RETURN
000200                  CRLF=   200             ;;CODE FOR CARRIAGE RETURN-LINE FEED
177776                  PS=     177776          ;;PROCESSOR STATUS WORD
177776                  PSW=    PS
177774                  STKLMT= 177774          ;;STACK LIMIT REGISTER
177772                  PIRQ=   177772          ;;PROGRAM INTERRUPT REQUEST REGISTER
177570                  DSWR=   177570          ;;HARDWARE SWITCH REGISTER
177570                  DDISP=  177570          ;;HARDWARE DISPLAY REGISTER
                        ;*GENERAL PURPOSE REGISTER DEFINITIONS
000000                  R0=     %0              ;;GENERAL REGISTER
000001                  R1=     %1              ;;GENERAL REGISTER
000002                  R2=     %2              ;;GENERAL REGISTER
000003                  R3=     %3              ;;GENERAL REGISTER
000004                  R4=     %4              ;;GENERAL REGISTER
000005                  R5=     %5              ;;GENERAL REGISTER
000006                  R6=     %6              ;;GENERAL REGISTER
000007                  R7=     %7              ;;GENERAL REGISTER
000006                  SP=     %6              ;;STACK POINTER
000007                  PC=     %7              ;;PROGRAM COUNTER
                        ;*PRIORITY LEVEL DEFINITIONS
000000                  PR0=    0               ;;PRIORITY LEVEL 0
```

BASIC DEFINITIONS

```
000040                  PR1=    40              ;;PRIORITY LEVEL 1
000100                  PR2=    100             ;;PRIORITY LEVEL 2
000140                  PR3=    140             ;;PRIORITY LEVEL 3
000200                  PR4=    200             ;;PRIORITY LEVEL 4
000240                  PR5=    240             ;;PRIORITY LEVEL 5
000300                  PR6=    300             ;;PRIORITY LEVEL 6
000340                  PR7=    340             ;;PRIORITY LEVEL 7
                        ;*"SWITCH REGISTER" SWITCH DEFINITIONS
100000                  SW15=   100000
040000                  SW14=   40000
020000                  SW13=   20000
010000                  SW12=   10000
004000                  SW11=   4000
002000                  SW10=   2000
001000                  SW09=   1000
000400                  SW08=   400
000200                  SW07=   200
000100                  SW06=   100
000040                  SW05=   40
000020                  SW04=   20
000010                  SW03=   10
000004                  SW02=   4
000002                  SW01=   2
000001                  SW00=   1
001000                  SW9=    SW09
000400                  SW8=    SW08
000200                  SW7=    SW07
000100                  SW6=    SW06
000040                  SW5=    SW05
000020                  SW4=    SW04
000010                  SW3=    SW03
000004                  SW2=    SW02
000002                  SW1=    SW01
000001                  SW0=    SW00
                        ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
100000                  BIT15=  100000
040000                  BIT14=  40000
020000                  BIT13=  20000
010000                  BIT12=  10000
004000                  BIT11=  4000
002000                  BIT10=  2000
001000                  BIT09=  1000
000400                  BIT08=  400
000200                  BIT07=  200
000100                  BIT06=  100
000040                  BIT05=  40
000020                  BIT04=  20
000010                  BIT03=  10
000004                  BIT02=  4
000002                  BIT01=  2
000001                  BIT00=  1
001000                  BIT9=   BIT09
000400                  BIT8=   BIT08
000200                  BIT7=   BIT07
000100                  BIT6=   BIT06
000040                  BIT5=   BIT05
000020                  BIT4=   BIT04
```

BASIC DEFINITIONS

```
                000010                  BIT3=   BIT03
                000004                  BIT2=   BIT02
                000002                  BIT1=   BIT01
                000001                  BIT0=   BIT00
                                        ;*BASIC "CPU" TRAP VECTOR ADDRESSES
                000004                  ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
                000010                  RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
                000014                  TBITVEC=14              ;;"T" BIT
                000014                  TRTVEC= 14              ;;TRACE TRAP
                000014                  BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
                000020                  IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
                000024                  PWRVEC= 24              ;;POWER FAIL
                000030                  EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
                000034                  TRAPVEC=34              ;;"TRAP" TRAP
                000060                  TKVEC=  60              ;;TTY KEYBOARD VECTOR
                000064                  TPVEC=  64              ;;TTY PRINTER VECTOR
                000240                  PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR
    499         000001                          UFDSET=         1               ;FLAG FOR UFD
    500                                 .SBTTL  TRAP CATCHER
                000000                          .=0
                                        ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
                                        ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
                                        ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
                000174                          .=174
    000174      000000              DISPREG:    .WORD   0               ;;SOFTWARE DISPLAY REGISTER
    000176      000000              SWREG:      .WORD   0               ;;SOFTWARE SWITCH REGISTER
    501         000200                          .=200
    502 000200  005037  001160                  CLR     $TMP0
    503 000204  000137  004024                  JMP     @#START
    504         000220                          .=220
    505 000220  012737  000777  001160          MOV     #777,$TMP0
    506 000226  000137  004024                  JMP     @#START
    507
    508                                 .SBTTL  ACT11 HOOKS
                                        ;;**************************************************************
                                        ;HOOKS REQUIRED BY ACT11
                                                $SVPC=.                 ;SAVE PC
                000232                          .=46
                000046                          $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
    000046      140440                          .=52
                000052                          .WORD   0               ;;2)SET LOC.52 TO ZERO
    000052      000000                          .=$SVPC                 ;; RESTORE PC
                000232
    509                                 .SBTTL  APT PARAMETER BLOCK
                                        ;;**************************************************************
                                        ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
                                        ;;**************************************************************
                000232                          .$X=.   ;;SAVE CURRENT LOCATION
                000024                          .=24    ;;SET POWER FAIL TO POINT TO START OF PROGRAM
    000024      000200                          200     ;;FOR APT START UP
                000044                          .=44    ;;POINT TO APT INDIRECT ADDRESS PNTR.
    000044      000232                          $APTHDR ;;POINT TO APT HEADER BLOCK
                000232                          .=.$X   ;;RESET LOCATION COUNTER
                                        ;;**************************************************************
                                        ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
                                        ;INTERFACE SPEC.
    000232                              $APTHD:
    000232      000000                  $HIBTS: .WORD   0       ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
```

D2

APT PARAMETER BLOCK

```
000234  001200                  $MBADR: .WORD   $MAIL    ;;ADDRESS OF APT MAILBOX (BITS 0 15)
000236  000000                  $TSTM:  .WORD            ;;RUN TIM OF LONGEST TEST
000240  000000                  $PASTM: .WORD            ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
000242  000000                  $UNITM: .WORD            ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
000244  000052                          .WORD   $ETEND $MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
```

```
    510                                          .SBTTL  COMMON TAGS
                                                 ;;*******************************************************************
                                                 ;*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
                                                 ;*USED IN THE PROGRAM.
            001100                                       .=1100
    001100                                       $CMTAG:                         ;;START OF COMMON TAGS
    001100  000000                                       .WORD   0
    001102     000                               $TSTNM: .BYTE   0               ;;CONTAINS THE TEST NUMBER
    001103     000                               $ERFLG: .BYTE   0               ;;CONTAINS ERROR FLAG
    001104  000000                               $ICNT:  .WORD   0               ;;CONTAINS SUBTEST ITERATION COUNT
    001106  000000                               $LPADR: .WORD   0               ;;CONTAINS SCOPE LOOP ADDRESS
    001110  000000                               $LPERR: .WORD   0               ;;CONTAINS SCOPE RETURN FOR ERRORS
    001112  000000                               $ERTTL: .WORD   0               ;;CONTAINS TOTAL ERRORS DETECTED
    001114     000                               $ITEMB: .BYTE   0               ;;CONTAINS ITEM CONTROL BYTE
    001115     001                               $ERMAX: .BYTE   1               ;;CONTAINS MAX. ERRORS PER TEST
    001116  000000                               $ERRPC: .WORD   0               ;;CONTAINS PC OF LAST ERROR INSTRUCTION
    001120  000000                               $GDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'GOOD' DATA
    001122  000000                               $BDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'BAD' DATA
    001124  000000                               $GDDAT: .WORD   0               ;;CONTAINS 'GOOD' DATA
    001126  000000                               $BDDAT: .WORD   0               ;;CONTAINS 'BAD' DATA
    001130  000000                                       .WORD   0               ;;RESERVED--NOT TO BE USED
    001132  000000                                       .WORD   0
    001134     000                               $AUTOB: .BYTE   0               ;;AUTOMATIC MODE INDICATOR
    001135     000                               $INTAG: .BYTE   0               ;;INTERRUPT MODE INDICATOR
    001136  000000                                       .WORD   0
    001140  177570                               SWR:    .WORD   DSWR            ;;ADDRESS OF SWITCH REGISTER
    001142  177570                               DISPLAY: .WORD  DDISP           ;;ADDRESS OF DISPLAY REGISTER
    001144  177560                               $TKS:   177560                  ;;TTY KBD STATUS
    001146  177562                               $TKB:   177562                  ;;TTY KBD BUFFER
    001150  177564                               $TPS:   177564                  ;;TTY PRINTER STATUS REG. ADDRFSS
    001152  177566                               $TPB:   177566                  ;;TTY PRINTER BUFFER REG. ADDRESS
    001154     000                               $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER FOR FILLS
    001155     002                               $FILLS: .BYTE   2               ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
    001156     012                               $FILLC: .BYTE   12              ;;INSERT FILL CHARS. AFTER A "LINE FEED"
    001157     000                               $TPFLG: .BYTE   0               ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
            000002                               .REPT   2
    001160  000000                               $TMP0:  .WORD   0               ;;USER DEFINED
    001162  000000                               $TMP1:  .WORD   0               ;;USER DEFINED
    001164  000000                               $TIMES: 0                       ;;MAX. NUMBER OF ITERATIONS
    001166  000000                               $ESCAPE:0                       ;;ESCAPE ON ERROR ADDRESS
    001170     207     377     377               $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
    001173     000
    001174     077                               $QUES:  .ASCII  /?/             ;;QUESTION MARK
    001175     015                               $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
    001176     012     000                       $LF:    .ASCIZ  <12>            ;;LINE FEED
                                                 ;;*******************************************************************
                                                 .SBTTL  APT MAILBOX-ETABLE
                                                 ;;*******************************************************************
                                                 .EVEN
    001200                                       $MAIL:                  ;;APT MAILBOX
    001200  000000                               $MSGTY: .WORD   AMSGTY  ;;MESSAGE TYPE CODE
    001202  000000                               $FATAL: .WORD   AFATAL  ;;FATAL ERROR NUMBER
    001204  000000                               $TESTN: .WORD   ATESTN  ;;TEST NUMBER
    001206  000000                               $PASS:  .WORD   APASS   ;;PASS COUNT
    001210  000000                               $DEVCT: .WORD   ADEVCT  ;;DEVICE COUNT
    001212  000000                               $UNIT:  .WORD   AUNIT   ;;I/O UNIT NUMBER
    001214  000000                               $MSGAD: .WORD   AMSGAD  ;;MESSAGE ADDRESS
```

F2

APT MAILBOX ETABLE

```
001216  000000                  $MSGLG: .WORD   AMSGLG  ;;MESSAGE LENGTH
001220                          $ETABLE:                ;;APT ENVIRONMENT TABLE
001220     000                  $ENV:   .BYTE   AENV    ;;ENVIRONMENT BYTE
001221     000                  $ENVM:  .BYTE   AENVM   ;;ENVIRONMENT MODE BITS
001222  000000                  $SWREG: .WORD   ASWREG  ;;APT SWITCH REGISTER
001224  000000                  $USWR:  .WORD   AUSWR   ;;USER SWITCHES
001226  000000                  $CPUOP: .WORD   ACPUOP  ;;CPU TYPE,OPTIONS
                                ;*                      BITS 15-11=CPU TYPE
                                ;*                             11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
                                ;*                             11/70=06,PDQ=07,Q=10
                                ;*                      BIT 10=REAL TIME CLOCK
                                ;*                      BIT  9=FLOATING POINT PROCESSOR
                                ;*                      BIT  8=MEMORY MANAGEMENT
001230     000                  $MAMS1: .BYTE   AMAMS1  ;;HIGH ADDRESS,M.S. BYTE
001231     000                  $MTYP1: .BYTE   AMTYP1  ;;MEM. TYPE,BLK#1
                                ;*                      MEM.TYPE BYTE   --  (HIGH BYTE)
                                ;*                              900 NSEC CORE=001
                                ;*                              300 NSEC BIPOLAR=002
                                ;*                              500 NSEC MOS=003
001232  000000                  $MADR1: .WORD   AMADR1  ;;HIGH ADDRESS,BLK#1
                                ;*                        MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
001234     000                  $MAMS2: .BYTE   AMAMS2  ;;HIGH ADDRESS,M.S. BYTE
001235     000                  $MTYP2: .BYTE   AMTYP2  ;;MEM.TYPE,BLK#2
001236  000000                  $MADR2: .WORD   AMADR2  ;;MEM.LAST ADDRESS,BLK#2
001240     000                  $MAMS3: .BYTE   AMAMS3  ;;HIGH ADDRESS,M.S.BYTE
001241     000                  $MTYP3: .BYTE   AMTYP3  ;;MEM.TYPE,BLK#3
001242  000000                  $MADR3: .WORD   AMADR3  ;;MEM.LAST ADDRESS,BLK#3
001244     000                  $MAMS4: .BYTE   AMAMS4  ;;HIGH ADDRESS,M.S.BYTE
001245     000                  $MTYP4: .BYTE   AMTYP4  ;;MEM.TYPE,BLK#4
001246  000000                  $MADR4: .WORD   AMADR4  ;;MEM.LAST ADDRESS,BLK#4
001250  000000                  $VECT1: .WORD   AVECT1  ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
001252  000000                  $VECT2: .WORD   AVECT2  ;;INTERRUPT VECTOR#2BUS PRIORITY#2
001254  000000                  $BASE:  .WORD   ABASE   ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
001256  000000                  $DEVM:  .WORD   ADEVM   ;;DEVICE MAP
001260  000000                  $CDW1:  .WORD   ACDW1   ;;CONTROLLER DESCRIPTION WORD#1
001262  000000                  $CDW2:  .WORD   ACDW2   ;;CONTROLLER DESCRIPTION WORD#2
001264  000000                  $DDW0:  .WORD   ADDW0   ;;DEVICE DESCRIPTOR WORD#0
001266  000000                  $DDW1:  .WORD   ADDW1   ;;DEVICE DESCRIPTOR WORD#1
001270  000000                  $DDW2:  .WORD   ADDW2   ;;DEVICE DESCRIPTOR WORD#2
001272  000000                  $DDW3:  .WORD   ADDW3   ;;DEVICE DESCRIPTOR WORD#3
001274  000000                  $DDW4:  .WORD   ADDW4   ;;DEVICE DESCRIPTOR WORD#4
001276  000000                  $DDW5:  .WORD   ADDW5   ;;DEVICE DESCRIPTOR WORD#5
001300  000000                  $DDW6:  .WORD   ADDW6   ;;DEVICE DESCRIPTOR WORD#6
001302  000000                  $DDW7:  .WORD   ADDW7   ;;DEVICE DESCRIPTOR WORD#7
001304  000000                  $DDW8:  .WORD   ADDW8   ;;DEVICE DESCRIPTOR WORD#8
001306  000000                  $DDW9:  .WORD   ADDW9   ;;DEVICE DESCRIPTOR WORD#9
001310  000000                  $DDW10: .WORD   ADDW10  ;;DEVICE DESCRIPTOR WORD#10
001312  000000                  $DDW11: .WORD   ADDW11  ;;DEVICE DESCRIPTOR WORD#11
001314  000000                  $DDW12: .WORD   ADDW12  ;;DEVICE DESCRIPTOR WORD#12
001316  000000                  $DDW13: .WORD   ADDW13  ;;DEVICE DESCRIPTOR WORD#13
001320  000000                  $DDW14: .WORD   ADDW14  ;;DEVICE DESCRIPTOR WORD#14
001322  000000                  $DDW15: .WORD   ADDW15  ;;DEVICE DESCRIPTOR WORD#15
001324                          $ETEND:
```

ERROR POINTER TABLE

```
                                        .SBTTL  ERROR POINTER TABLE
                                        ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
                                        ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
                                        ;*LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
                                        ;*NOTE1:        IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
                                        ;*NOTE2:        EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:
                                        ;*      EM              ;;POINTS TO THE ERROR MESSAGE
                                        ;*      DH              ;;POINTS TO THE DATA HEADER
                                        ;*      DT              ;;POINTS TO THE DATA
                                        ;*      DF              ;;POINTS TO THE DATA FORMAT
     001324                             $ERRTB:
  511
  512
  513                                   .SBTTL  ERROR DEFINITIONS
  514                                   ;ITEM 1
  515 001324  127002                            EM1             ;CPU ERROR
  516 001326  134621                            DH1             ;TEST #, ERROR PC
  517 001330  136026                            DT1             ;$TMP1,$ERRPC
  518 001332  000000                            0
  519                                   ;ITEM 2
  520 001334  127036                            EM2             ;MMU ERROR
  521 001336  134621                            DH1             ;TEST #, ERROR PC
  522 001340   36026                            DT1             ;$TMP1,$ERRPC
  523 001342  000000                            0
  524                                   ;ITEM 3
  525 001344  127050                            EM3             ;FPP ERROR
  526 001346  134621                            DH1             ;TEST #, ERROR PC
  527 001350  136026                            DT1             ;$TMP1,$ERRPC
  528 001352  000000                            0
  529                                   ;ITEM 4
  530 001354  127062                            EM4             ;ERROR IN READ-WRITE BITS OF CCR
  531 001356  134646                            DH4             ;TEST #, PC, EXPECTED DATA, RECEIVED DATA
  532 001360  136034                            DT4             ;$TMP1,$ERRPC,R1,CCR
  533 001362  000000                            0
  534                                   ;ITEM 5
  535 001364  127122                            EM5             ;FORCE MISS WRITES TO CACHE
  536 001366  134733                            DH5             ;TEST #, PC, HIT/MISS, DATA IN CACHE, DATA IN MEMORY
  537 001370  136046                            DT5             ;$TMP1,$ERRPC, R2, R1, $GDDAT
  538 001372  000000                            0
  539                                   ;ITEM 6
  540 001374  127155                            EM6             ;FORCE MISS WRITE INVALIDATES CACHE
  541 001376  134733                            DH5             ;TEST #, PC, HIT/MISS, DATA IN CACHE, DATA IN MEMORY
  542 001400  136046                            DT5             ;$TMP1,$ERRPC, R2, R1, $GDDAT
  543 001402  000000                            0
  544                                   ;ITEM 7
  545 001404  127220                            EM7             ;UNEXPECTED PARITY INTERRUPT
  546 001406  135032                            DH7             ;TEST #, PC, ADDRESS ACCESSED, MSER
  547 001410  136062                            DT7             ;$TMP1,$ERRPC,$BDADR,MSER
  548 001412  000000                            0
  549                                   ;ITEM 10
  550 001414  127254                            EM10            ;TAG PARITY ERROR
  551 001416  135032                            DH7             ;TEST #, PC, ADDRESS ACCESSED, MSER
  552 001420  136062                            DT7             ;$TMP1,$ERRPC,$BDADR,MSER
  553 001422  000000                            0
  554                                   ;ITEM 11
  555 001424  127275                            EM11            ;DATA PARITY ERROR
  556 001426  135032                            DH7             ;TEST #, PC, ADDRESS ACCESSED, MSER
```

H2

ERROR DEFINITIONS

```
    557 001430  136062                    DT7             ;$TMP1,$ERRPC,$BDADR,MSER
    558 001432  000000                    0
    559                           ;ITEM 12
    560 001434  127317                    EM12            ;LOW BYTE PARITY ERROR
    561 001436  135032                    DH7             ;TEST #, PC, ADDRESS ACCESSED, MSER
    562 001440  136062                    DT7             ;$TMP1,$ERRPC,$BDADR,MSER
    563 001442  000000                    0
    564                           ;ITEM 13
    565 001444  127345                    EM13            ;HIGH BYTE PARITY ERROR
    566 001446  135032                    DH7             ;TEST #, PC, ADDRESS ACCESSED, MSER
    567 001450  136062                    DT7             ;$TMP1,$ERRPC,$BDADR,MSER
    568 001452  000000                    0
    569                           ;ITEM 14
    570 001454  127374                    EM14            ;ERROR DATA PATH
    571 001456  134646                    DH4             ;TEST #, PC, EXPECTED DATA, RECEIVED DATA
    572 001460  136074                    DT14            ;$TMP1,$ERRPC,$GDDAT,TSTLOC
    573 001462  000000                    0
    574                           ;ITEM 15
    575 001464  127417                    EM15            ;FORCE MISS READS FROM CACHE
    576 001466  134733                    DH5             ;TEST #, PC, HIT/MISS, DATA IN CACHE, DATA IN MEMORY
    577 001470  136046                    DT5             ;$TMP1,$ERRPC, R2, R1, $GDDAT
    578 001472  000000                    0
    579                           ;ITEM 16
    580 001474  127453                    EM16            ;FORCE M.. S READS FROM CACHE AND MISS
    581 001476  134733                    DH5             ;TEST #, PC, HIT/MISS, DATA IN CACHE, DATA IN MEMORY
    582 001500  136046                    DT5             ;$TMP1,$ERRPC, R2, R1, $GDDAT
    583 001502  000000                    0
    584                           ;ITEM 17
    585 001504  127520                    EM17            ;ERROR IN RECORDING HITS IN HIT/MISS
    586 001506  134646                    DH4             ;TEST #, PC, EXPECTED DATA, RECEIVED DATA
    587 001510  136106                    DT17            ;$TMP1,$ERRPC,$GDDAT,RECDAT
    588 001512  000000                    0
    589                           ;ITEM 20
    590 001514  127564                    EM20            ;WRITE BYTE ALLOCATES CACHE
    591 001516  134621                    DH1             ;TEST #, PC
    592 001520  136026                    DT1             ;$TMP1,$ERRPC
    593 001522  000000                    0
    594                           ;ITEM 21
    595 001524  127617                    EM21            ;WRITE BYTE HIT DOES NOT RECORD HIT
    596 001526  134621                    DH1             ;TEST #, PC
    597 001530  136026                    DT1             ;$TMP1,$ERRPC
    598 001532  000000                    0
    599                           ;ITEM 22
    600 001534  127662                    EM22            ;BYTES REVERSED ON WRITE CYCLES
    601 001536  134621                    DH1             ;TEST #, PC
    602 001540  136026                    DT1             ;$TMP1,$ERRPC
    603 001542  000000                    0
    604                           ;ITEM 23
    605 001544  127721                    EM23            ;CONDITIONAL BYPASS DOESN'T INVALIDATE CACHE
    606 001546  134733                    DH5             ;TEST #, PC, HIT/MISS, DATA IN CACHE, DATA IN MEMORY
    607 001550  136046                    DT5             ;$TMP1,$ERRPC, R2, R1, $GDDAT
    608 001552  000000                    0
    609                           ;ITEM 24
    610 001554  127775                    EM24            ;HITS RECORDED AFTER FLUSHING CACHE
    611 001556  135105                    DH24            ;TEST #, PC, NUMBER OF HITS
    612 001560  136120                    DT24            ;$TMP1,$ERRPC,R3
    613 001562  000000                    0
```

ERROR DEFINITIONS

```
614
615 001564  130040          ;ITEM 25
                                EM25            ;BYPASS DOESN'T INVALIDATE CACHE
616 001566  134621              DH1             ;TEST #, PC
617 001570  136026              DT1             ;$TMP1,$ERRPC
618 001572  000000              0
619                         ;ITEM 26
620 001574  130100              EM26            ;MSER DOES NOT CLEAR ON WRITE REFERENCE
621 001576  134621              DH1             ;TEST #, PC
622 001600  136026              DT1             ;$TMP1,$ERRPC
623 001602  000000              0
624                         ;ITEM 27
625 001604  130147              EM27            ;PARITY ERROR DON'T CAUSE A MISS
626 001606  135151              DH27            ;TEST #, PC, MSER, HIT/MISS
627 001610  136130              DT27            ;$TMP1,$ERRPC,MSER,R3,0
628 001612  000000              0
629                         ;ITEM 30
630 001614  130207              EM30            ;PARITY ERROR DON'T SET MSER WITH CCR<7>=0
631 001616  135151              DH27            ;TEST #, PC, MSER, HIT/MISS
632 001620  136130              DT27            ;$TMP1,$ERRPC,MSER,R3,0
633 001622  000000              0
634                         ;ITEM 31
635 001624  130261              EM31            ;PARITY ERROR IGNORED
636 001626  134621              DH1             ;TEST #, PC
637 001630  136026              DT1             ;$TMP1,$ERRPC
638 001632  000000              0
639                         ;ITEM 32
640 001634  130306              EM32            ;PARITY ERROR IGNORED ON LOW BYTE
641 001636  134621              DH1             ;TEST #, PC
642 001640  136026              DT1             ;$TMP1,$ERRPC
643 001642  000000              0
644                         ;ITEM 33
645 001644  130347              EM33            ;PARITY ERROR IGNORED ON HIGH BYTE
646 001646  134621              DH1             ;TEST #, PC
647 001650  136026              DT1             ;$TMP1,$ERRPC
648 001652  000000              0
649                         ;ITEM 34
650 001654  130411              EM34            ;PARITY ABORT LOGIC DOESN'T WORK
651 001656  134621              DH1             ;TEST #, PC
652 001660  136026              DT1             ;$TMP1,$ERRPC
653 001662  000000              0
654                         ;ITEM 35
655 001664  130451              EM35            ;MSER NOT SET PROPERLY
656 001666  134646              DH4             ;TEST #, PC, EXPECTED DATA, RECEIVED DATA
657 001670  136142              DT35            ;$TMP1,$ERRPC,$GDDAT,MSER,0
658 001672  000000              0
659                         ;ITEM 36
660 001674  130477              EM36            ;PARITY INTERRUPT DOESN'T WORK
661 001676  134621              DH1             ;TEST #, PC
662 001700  136026              DT1             ;$TMP1,$ERRPC
663 001702  000000              0
664                         ;ITEM 37
665 001704  130543              EM37            ;NXM AND PARITY ABORT DIN'T HAPPEN
666 001706  134621              DH1             ;TEST #, PC
667 001710  136026              DT1             ;$TMP1,$ERRPC
668 001712  000000              0
669                         ;ITEM 40
670 001714  130605              EM40            ;PARITY ABORT NOT BLOCKED BY NXM TRAP
```

ERROR DEFINITIONS

```
                                        DH1             ;TEST #, PC
                                        DT1             ;$TMP1,$ERRPC
                                        0
                                ;ITEM 41
                                        EM41            ;MULTI-PROCESSOR HOOK INSTRUCTION DOESN T CAUSE MISS
                                        DH41            ;TEST #, PC, INSTRUCTION OPCODE
                                        DT41            ;$TMP1,$ERRPC,$BDDAT
                                        0
                                ;ITEM 42
                                        EM42            ;ERROR IN PARITY LOGIC
                                        DH1             ;TEST #, PC
                                        DT1             ;$TMP1,$ERRPC
                                        0
                                ;ITEM 43
                                        EM43            ;ERROR IN CACHE DATA RAMS
                                        DH43            ;TEST #, PC, EXPECTED DATA, RECEIVED DATA, CACHE LOCATION
                                        DT43            ;$TMP1,$ERRPC,R1,RECDAT,$BDADR
                                        0
                                ;ITEM 44
                                        EM44            ;ERROR IN NXM IN STANDALONE MODE
                                        DH1             ;TEST #, PC
                                        DT1             ;$TMP1,$ERRPC
                                        0
                                ;ITEM 45
                                        EM45            ;HITS NOT RECORDED PROPERLY THRU HIT/MISS
                                        DH1             ;TEST #, PC
                                        DT1             ;$TMP1,$ERRPC
                                        0
                                ;ITEM 46
                                        EM46            ;HIT RECORDED FOR A LOCATION NOT IN CACHE
                                        DH47            ;TEST #, PC, LOCATION ACCESSED
                                        DT47            ;$TMP1,$ERRPC,KIPAR6,$BDADR
                                        0
                                ;ITEM 47
                                        EM47            ;MISS RECORDED FOR A LOCATION THAT SHOULD BE IN CACHE
                                        DH47            ;TEST #, PC, LOCATION ACCESSED
                                        DT47            ;$TMP1,$ERRPC,KIPAR6,$BDADR
                                        0
                                ;ITEM 50
                                        EM50            ;ERROR IN TAG STORE
                                        DH47            ;TEST #, PC, ADDRESS
                                        DT50            ;$TMP1,$ERRPC,R1,$BDADR
                                        0
                                ;ITEM 51
                                        EM51            ;ERROR PCR READ-WRITE BITS
                                        DH4             ;TEST #, PC, EXPECTED DATA, RECEIVED DATA
                                        DT51            ;$TMP1,$ERRPC,$GDDAT,PCR
                                        0
                                ;ITEM 52
                                        EM52            ;ERROR IN BCSR READ-WRITE BITS
                                        DH4             ;TEST #, PC, EXPECTED DATA, RECEIVED DATA
                                        DT52            ;$TMP1,$ERROC,$GDDAT,BCSR
                                        0
                                ;ITEM 53
                                        EM53            ;RESET DOESN'T CLEAR BCSR<4>
                                        DH1             ;TEST #, PC
                                        DT1             ;$TMP1,$ERRPC
```

K2

ERROR DEFINITIONS

```
728 002052  000000                    0
729                             ;ITEM 54
730 002054  131463                  EM54        ;CHECKSUM ERROR IN 16-BIT ROM
731 002056  134621                  DH1         ;TEST #, PC
732 002060  136026                  DT1         ;$TMP1,$ERRPC
733 002062  000000                    0
734                             ;ITEM 55
735 002064  131521                  EM55        ;CHCKSUM ERROR IN 8-BIT ROM
736 002066  134621                  DH1         ;TEST #, PC
737 002070  136026                  DT1         ;$TMP1,$ERRPC
738 002072  000000                    0
739                             ;ITEM 56
740 002074  131555                  EM56        ;TIMEOUT READING LKS
741 002076  134621                  DH1         ;TEST #, PC
742 002100  136026                  DT1         ;$TMP1,$ERRPC
743 002102  000000                    0
744                             ;ITEM 57
745 002104  131601                  EM57        ;LKS<07> DOES NOT BECOME 1
746 002106  134621                  DH1         ;TEST #, PC
747 002110  136026                  DT1         ;$TMP1,$ERRPC
748 002112  000000                    0
749                             ;ITEM 60
750 002114  131633                  EM60        ;WRITE REFERENCE DOESN'T CLEAR LKS<07>
751 002116  134621                  DH1         ;TEST #, PC
752 002120  136026                  DT1         ;$TMP1,$ERRPC
753 002122  000000                    0
754                             ;ITEM 61
755 002124  131701                  EM61        ;ILLEGAL LKS INTERRUPTS
756 002126  134621                  DH1         ;TEST #, PC
757 002130  136026                  DT1         ;$TMP1,$ERRPC
758 002132  000000                    0
759                             ;ITEM 62
760 002134  131730                  EM62        ;PROCESSOR INTERRUPTS DON'T CLEAR LKS<07>
761 002136  134621                  DH1         ;TEST #, PC
762 002140  136026                  DT1         ;$TMP1,$ERRPC
763 002142  000000                    0
764                             ;ITEM 63
765 002144  132001                  EM63        ;LKS READY DOESN'T GO LOW
766 002146  134621                  DH1         ;TEST #, PC
767 002150  136026                  DT1         ;$TMP1,$ERRPC
768 002152  000000                    0
769                             ;ITEM 64
770 002154  132032                  EM64        ;WRONG NUMBER OF LKS INTERRUPTS
771 002156  134621                  DH1         ;TEST #, PC
772 002160  136026                  DT1         ;$TMP1,$ERRPC
773 002162  000000                    0
774                             ;ITEM 65
775 002164  132071                  EM65        ;LKS INTERRUPTS HAPPEN AT WRONG PRIORITY
776 002166  135452                  DH65        ;TEST #, PC, PRIORITY
777 002170  136262                  DT65        ;$TMP1,$ERRPC,$GUDAT
778 002172  000000                    0
779                             ;ITEM 66
780 002174  132141                  EM66        ;BCSR<12> DOES NOT DISABLES LKS
781 002176  134621                  DH1         ;TEST #, PC
782 002200  136026                  DT1         ;$TMP1,$ERRPC
783 002202  000000                    0
784                             ;ITEM 67
```

ERROR DEFINITIONS

```
    785 002204  132200                  EM67            ;BCSR<13> DOESN'T SET LKS<06>
    786 002206  134621                  DH1             ;TEST #, PC
    787 002210  136026                  DT1             ;$TMP1,$ERRPC
    788 002212  000000                  0
    789                         ;ITEM 70
    790 002214  132235                  EM70            ;RESET DOESN'T SET LKS<7>
    791 002216  134621                  DH1             ;TEST #, PC
    792 002220  136026                  DT1             ;$TMP1,$ERRPC
    793 002222  000000                  0
    794                         ;ITEM 71
    795 002224  132266                  EM71            ;RESET DOESN'T CLEAR LKS<06>
    796 002226  134621                  DH1             ;TEST #, PC
    797 002230  136026                  DT1             ;$TMP1,$ERRPC
    798 002232  000000                  0
    799                         ;ITEM 72
    800 002234  132322                  EM72            ;TIMEOUT READING SLU REGISTERS
    801 002236  135516                  DH72            ;TEST #, PC, ADDRESS FAILED
    802 002240  136154                  DT41            ;$TMP1,$ERRPC,$BDDAT
    803 002242  000000                  0
    804                         ;ITEM 73
    805 002244  132360                  EM73            ;XMIT READY DIDN'T GO LOW
    806 002246  134621                  DH1             ;TEST #, PC
    807 002250  136026                  DT1             ;$TMP1,$ERRPC
    808 002252  000000                  0
    809                         ;ITEM 74
    810 002254  132404                  EM74            ;RCSR DOESN'T BECOME 1
    811 002256  134621                  DH1             ;TEST #, PC
    812 002260  136026                  DT1             ;$TMP1,$ERRPC
    813 002262  000000                  0
    814                         ;ITEM 75
    815 002264  132435                  EM75            ;WRONG CHARACTER RECEIVED
    816 002266  134646                  DH4             ;TEST #, PC, EXPECTED DATA, RECEIVED DATA
    817 002270  136272                  DT75            ;$TMP1,$ERRPC,$GDDAT,$BDDAT
    818 002272  000000                  0
    819                         ;ITEM 76
    820 002274  132466                  EM76            ;RCSR<07> NOT CLEARED AFTER READING RBUF
    821 002276  134621                  DH1             ;TEST #, PC
    822 002300  136026                  DT1             ;$TMP1,$ERRPC
    823 002302  000000                  0
    824                         ;ITEM 77
    825 002304  132536                  EM77            ;XCSR<07> NOT SET ON RESET
    826 002306  134621                  DH1             ;TEST #, PC
    827 002310  136026                  DT1             ;$TMP1,$ERRPC
    828 002312  000000                  0
    829                         ;ITEM 100
    830 002314  132570                  EM100           ;RCSR<07> NOT CLEARED ON RESET
    831 002316  134621                  DH1             ;TEST #, PC
    832 002320  136026                  DT1             ;$TMP1,$ERRPC
    833 002322  000000                  0
    834                         ;ITEM 101
    835 002324  132626                  EM101           ;SLU INTERRUPTS HAPPEN AT 4
    836 002326  134621                  DH1             ;TEST #, PC
    837 002330  136026                  DT1             ;$TMP1,$ERRPC
    838 002332  000000                  0
    839                         ;ITEM 102
    840 002334  132661                  EM102           ;RESET DOES NOT CLEAR XCSR<6> AND RSCR<6>
    841 002336  134621                  DH1             ;TEST #, PC
```

M2

ERROR DEFINITIONS

```
842 002340  136026                      DT1             ;$TMP1,$ERRPC
843 002342  000000                      0
844                             ;ITEM 103
845 002344  132743                      EM103           ;TRANSMIT INTERRUPT DOES NOT CLEAR XCSR<07>
846 002346  134621                      DH1             ;TEST #, PC
847 002350  136026                      DT1             ;$TMP1,$ERRPC
848 002352  000000                      0
849                             ;ITEM 104
850 002354  133016                      EM104           ;RECEIVE INTERRUPTS DON'T CLEAR RCSR<07>
851 002356  134621                      DH1             ;TEST #, PC
852 002360  136026                      DT1             ;$TMP1,$ERRPC
853 002362  000000                      0
854                             ;ITEM 105
855 002364  133066                      EM105           ;BREAK CONDITION DOES NOT SET RBUF PROPERLY
856 002366  135562                      DH105           ;TEST #, PC, RBUF
857 002370  136304                      DT105           ;$TMP1,$ERRPC,RBUF
858 002372  000000                      0
859                             ;ITEM 106
860 002374  133141                      EM106           ;RBUF WASN'T CLEARED ON NEXT CHAR.
861 002376  135562                      DH105           ;TEST #, PC, RBUF
862 002400  136304                      DT105           ;$TMP1,$ERRPC,RBUF
863 002402  000000                      0
864                             ;ITEM 107
865 002404  133217                      EM107           ;ERROR IN WRITING TO XCSR<0>
866 002406  134621                      DH1             ;TEST #, PC
867 002410  136026                      DT1             ;$TMP1,$ERRPC
868 002412  000000                      0
869                             ;ITEM 110
870 002414  133253                      EM110           ;RESET DOES NOT CLEAR XCSR<00>
871 002416  134621                      DH1             ;TEST #, PC
872 002420  136026                      DT1             ;$TMP1,$ERRPC
873 002422  000000                      0
874                             ;ITEM 111
875 002424  133315                      EM111           ;FIRST CHARACTER WAS NOT OVERRUN BY THE SECOND
876 002426  134646                      DH4             ;TEST #, PC, EXPECTED DATA, RECEIVED DATA
877 002430  136272                      DT75            ;$TMP1,$ERRPC,$GDDAT,$BDDAT
878 002432  000000                      0
879                             ;ITEM 112
880 002434  133373                      EM112           ;OVERRUN CONDITION DOES NOT SET PROPER BITS IN RBUF
881 002436  135562                      DH105           ;TEST #, PC, RBUF
882 002440  136304                      DT105           ;$TMP1,$ERRPC,RBUF
883 002442  000000                      0
884                             ;ITEM 113
885 002444  133456                      EM113           ;RBUF WAS NOT CLEARED ON THE NEXT CHARACTER
886 002446  135562                      DH105           ;TEST #, PC, RBUF
887 002450  136304                      DT105           ;$TMP1,$ERRPC,RBUF
888 002452  000000                      0
889                             ;ITEM 114
890 002454  133542                      EM114           ;ERROR IN XCSR<2>
891 002456  134621                      DH1             ;TEST #, PC
892 002460  136026                      DT1             ;$TMP1,$ERRPC
893 002462  000000                      0
894                             ;ITEM 115
895 002464  133563                      EM115           ;ERROR IN TAG STORE FROM STANDALONE MODE
896 002466  135614                      DH115           ;TEST #, PC, MSER, ADDRESS ACCESSED
897 002470  136314                      DT115           ;$TMP1,$ERRPC,$BDDAT,KIPAR6,$BDADR
898 002472  000000                      0
```

ERROR DEFINITIONS

```
899
900 002474  133633                ;ITEM 116
901 002476  134621                        EM116           ;DMA TAG PARITY DOES NOT SET MSER<4>
902 002500  136026                        DH1             ,TEST #, PC
903 002502  000000                        DT1             ;$TMP1,$ERRPC
904                                       0
905 002504  133714                ;ITEM 117
906 002506  134621                        EM117           ;MSER<13>NOT SET IN STANDALONE MODE
907 002510  136026                        DH1             ;TEST #, PC
908 002512  000000                        DT1             ;$TMP1,$ERRPC
909                                       0
910 002514  133757                ;ITEM 120
911 002516  134621                        EM120           ;DMA WRITE HITS DON'T INVALIDATE CACHE
912 002520  136026                        DH1             ;TEST #, PC
913 002522  000000                        DT1             ;$TMP1,$ERRPC
914                                       0
915 002524  134025                ;ITEM 121
916 002526  134621                        EM121           ;IN BLOCK MODE ON WRITE DMA HITS NOT EVERYTHING IS INVALIDATED
917 002530  136026                        DH1             ;TEST #, PC
918 002532  000000                        DT1             ;$TMP1,$ERRPC
919                                       0
920 002534  134123                ;ITEM 122
921 002536  134621                        EM122           ;READ DMA HIT IS MESSED UP
922 002540  136026                        DH1             ;TEST #, PC
923 002542  000000                        DT1             ;$TMP1,$ERRPC
924                                       0
925 002544  134151                ;ITEM 123
926 002546  134621                        EM123           ;ERROR IN DMA CYCLES FROM Q22BE
927 002550  136026                        DH1             ;TEST #, PC
928 002552  000000                        DT1             ;$TMP1,$ERRPC
929                                       0
930 002554  134203                ;ITEM 124
931 002556  134621                        EM124           ;PIRQ INTERRUPTS DON'T TAKE PRIORITY OVER Q BUS INTERRUPTS
932 002560  136026                        DH1             ;TEST #, PC
933 002562  000000                        DT1             ;$TMP1,$ERRPC
934                                       0
935 002564  134275                ;ITEM 125
936 002566  134621                        EM125           ;NO POWER DOWN TRAP TO 24 OCCUR
937 002570  136026                        DH1             ;TEST #, PC
938 002572  000000                        DT1             ;$TMP1,$ERRPC
939                                       0
940 002574  134334                ;ITEM 126
941 002576  134621                        EM126           ;ERROR IN INTERRUPTS FROM Q22BE
942 002600  136026                        DH1             ;TEST #, PC
943 002602  000000                        DT1             ;$TMP1,$ERRPC
944                                       0
945 002604  134371                ;ITEM 127
946 002606  134621                        EM127           ;ERROR IN PMG COUNTER
947 002610  136026                        DH1             ;TEST #, PC
948 002612  000000                        DT1             ;$TMP1,$ERRPC
949                                       0
950 002614  134433                ;ITEM 130
951 002616  134621                        EM130           ;UNEXPECTED TIMEOUT
952 002620  136330                        DH1             ;TEST #, PC
953 002622  000000                        DT130           ;$TMP1,$ERRPC
954                                       0
955 002624  134460                ;ITEM 131
                                          EM131           ;ERROR WRITING TO LKS<6>
```

ERROR DEFINITIONS

```
                                        DH1             ;TEST #, PC
                                        DT1             ;$TMP1,$ERRPC
                                        0
                                ;ITEM 132
                                        EM132           ;MAINTENANCE REGISTER ERROR
                                        DH1             ;TEST #, PC
                                        DT1             ;$TMP1,$ERRPC
                                        0
                                ;ITEM 133
                                        EM133           ; EEROM TYPE ERROR
                                        DH1             ;TEST #, PC
                                        DT1             ;$TMP1,$ERRPC
                                        0
                                ;ITEM 134
                                        EM134           ; EROM WRITE/READ ERROR
                                        DH134           ;TEST #, PC, ERROR COUNT, PATTERN, DATA READ, ADDRESS
                                        DT134           ;$TMP1,$ERRPC,$TMP0,R3,$BDDAT,$BDADR
                                        0

                                .SBTTL  GLOBAL VARIABLES AND REGISTER NAMES

                                ;REGISTERS FOR THE FIRST Q22BE
000000                          CSR1:   .WORD   0               ;CONTROL REGISTER 1 FOR Q22BE
000000                          CSR2:   .WORD   0               ;CONTROL/STATUS REGISTER 2
000000                          BA:     .WORD   0               ;DMA ADDRESS FOR Q22BE
000000                          WC:     .WORD   0               ;WORD COUNT REGISTER
000000                          DATA:   .WORD   0               ;DMA DATA FOR Q22BE
000000                          VQBE1:  .WORD   0               ;ADDRESS OF VECTOR FOR Q22BE
000000                          VQPR1:  .WORD   0               ;PRIORITY
000000                          SIMGOA: .WORD   0               ;SIMULTANEUOS GO ADDRESS REGISTER

                                ;REGISTERS FOR THE SECOND Q22BE
000000                          CSR12:  .WORD   0               ;CONTROL REGISTER 1 FOR Q22BE
000000                          CSR22:  .WORD   0               ;CONTROL/STATUS REGISTER 2
000000                          BA2:    .WORD   0               ;DMA ADDRESS FOR Q22BE
000000                          WC2:    .WORD   0               ;WORD COUNT REGISTER
000000                          DATA2:  .WORD   0               ;DMA DATA FOR Q22BE
000000                          VQBE2:  .WORD   0               ;ADDRESS OF VECTOR FOR Q22BE
000000                          VQPR2:  .WORD   0               ;PRIORITY

000000                          LKSFL:  .WORD   0
000000                          ACTCHS: .WORD   0               ;ACTUAL CHECKSUM
000000                          SAVPCR: .WORD   0
000000                          SAVBR:  .WORD   0
002740                          .=2740
000000                          TEMP:   .WORD   0
                                        .BLKW   15.             ;RESERVED FOR BLOCK MODE TRANSFER
000000                          TIMOUT: .WORD   0
000032  000012  000006          Q22EN:  .WORD   32,12,6,2       ;PRIORITY 7-4 FOR Q22BE
000002

177524                                  BCR=            177524  ;BOOT/DIAGNOSTICS CONFIGURATION
177524                                  BDR=            177524  ;BOOT/DIAGNOSTICS DISPLAY
177520                                  BCSR=           177520  ;BOOT/DIAGNOSTICS STATUS
177746                                  CCR=            177746  ;CACHE CONTROL REGISTER
177752                                  HITMIS=         177752  ;HIT OR MISS REGISTER
```

```
GLOBAL VARIABLES AND REGISTER NAMES

1012            177734                          KW 3=           177734          ;UNIBUS CONFIGURATION REGISTER
1013            177546                          LKS=            177546          ;CLOCK STATUS REGISTER
1014            177750                          MAIREG=         177750          ;MAINTENANCE REGISTER
1015            177744                          MSER=           177744          ;MEMORY SYSTEM ERROR
1016            177522                          PCR=            177522          ;PAGE CONTROL REGISTER
1017            177772                          PIR=            177772          ;PROGRAM INTERRUPT REQUEST
1018            177562                          RBUF=           177562          ;RECEIVER DATA BUFFER
1019            177560                          RCSR=           177560          ;RECEIVER STATUS REGISTER
1020            177566                          XBUF=           177566          ;TRANSMITTER DATA BUFFER
1021            177564                          XCSR=           177564          ;TRANSMITTER STATUS REGISTER
1022
1023            177766                          CPEREG=         177766          ;CPU ERROR REGISTER
1024
1025            177572                          MMR0=SR0                        ;MEMORY MANAGEMENT REG. 0
1026            177574                          MMR1=SR1                        ;MEMORY MANAGEMENT REG. 1
1027            177576                          MMR2=SR2                        ;MEMORY MANAGEMENT REG. 2
1028            172516                          MMR3=SR3                        ;MEMORY MANAGEMENT REG. 3
1029            120001                          POLY= 120001
1030            000000                          NULL=           0
1031
1032                                    .SBTTL  GLOBAL DATA SECTION
1033
1034                                    ;++
1035                                    ; THE GLOBAL DATA SECTION CONTAINS DATA THAT ARE USED
1036                                    ; IN MORE THAN ONE TEST.
1037                                    ;--
1038
1039                                    ;THESE LOCATIONS ARE USED IN MORE THAN ONE TEST TO STORE VECTOR DATA
1040                                    ;WHEN THE TEST NEEDS TO HAVE AN ERROR CONDITION RESPOND DIFFERENTLY
1041                                    ;FROM THE DEFAULT RESPONCE.
1042 003012    000000                  SLOC00: .WORD   0
1043 003014    000000                  SLOC01: .WORD   0
1044
1045                                    ;THESE LOCATIONS ARE USED IN MORE THAN ONE TEST TO STORE WORKING DATA.
1046 003016    000000                  LOWADD: .WORD   0               ;STORES LOW ADDRESS FOR RAM TESTS
1047 003020    000000                  GOODAD: .WORD   0               ;STORES GOOD ADDRESS FOR RAM TESTS
1048 003022    000000                  ERRCNT: .WORD   0               ; CONTAINS TOTAL NO. OF EEROM ERRORS
1049 003024    000000                  TSTADD: .WORD   0               ;ADDRESS STORE FOR RAM TESTS
1050 003026    000000                  NEWADD: .WORD   0               ;ADDRESS STORE FOR RAM TEST
1051 003030    000000                  FLAG:   .WORD   0               ;USED TO STORE "FLAG" CONDITIONS
1052 003032    000000                  CCHPAS: .WORD   0               ; flag-counter for control of Cache subtests
1053 003034    000000                  EEPAS:  .WORD   0               ; flag-counter for control of EEROM subtest
1054 003036    000000                  SAVSUP: .WORD   0               ;USED TO STORE SUPERVISOR STACK VALUE
1055 003040    000000                  SAVUSE: .WORD   0               ;USED TO STORE USER STACK VALUE
1056 003042    000000                  SAVMR0: .WORD   0               ;USED TO STORE MMU STATUS REGISTER 0 DATA
1057 003044    000000                  SAVMR1: .WORD   0               ;USED TO STORE MMU STATUS REGISTER 1 DATA
1058 003046    000000                  SAVMR2: .WORD   0               ;USED TO STORE MMU STATUS REGISTER 2 DATA
1059 003050    000000                  SAVSWR: .WORD   0               ; SAVE SFTWRE SWTCH REG DURING EEROM TEST
1060 003052                            FLOAT:  .BLKW   4               ;USED TO STORE VALUES FOR MMU TESTS
1061 003062                            FLO:    .BLKW   4               ;USED TO STORE VALUES FOR MMU TESTS
1062 003072    000000                  SEQ:    .WORD   0               ;STORES SEQUENCE NUMBER FOR JUMP TESTS
1063 003074    000000                  SPS:    .WORD   0               ;STORES STACK POINTER FOR JUMP TESTS
1064 003076    000000                  SPSJ:   .WORD   0               ;STORES STACK POINTER FOR JUMP TESTS
1065 003100                            BTEXP:  .BLKW   4               ;STORES EXPONENT DURING BIT TESTS
1066 003110                            BTRES:  .BLKW   4               ;STORES RECIEVED DATA FOR BIT TESTS
1067 003120    000000                  COUNT:  .WORD   0               ;ERROR INDICATOR FOR FLOATING POINT TESTS
1068 003122                            RECFEC: .BLKW   4               ;RECIEVED FLOATING POINT EXCEPTION CODE
```

GLOBAL DATA SECTION

```
1069 003132                          RECST:  .BLKW   4               ;RECIEVED FLOATING POINT STATUS
1070 003142                          RECDST: .BLKW   4               ;DESTINATION ADDRESS FOR FLOATING POINT TESTS
1071
1072                                 ;THESE LOCATIONS ARE USED BY MORE THAN ONE TEST AS LOOP COUNTERS
1073 003152  000000                  ALLCTR: .WORD   0
1074 003154  000000                  LOOPIN: .WORD   0
1075
1076                                 ;SOME MORE TEMPORARY STORAGE FOR RAM TESTS
1077 003156  000000                  SAVPOS: .WORD   0               ;STORES TEMPORARY BIT POSITIONS FOR RAM TESTS
1078 003160  000000                  MASK:   .WORD   0               ;STORES BIT MASK FOR ERROR ISOLATION
1079
1080                                 ;!!!!!THIS IS IT. THE PROGRAM TEST LOCATION!!!!!!!!!!!!!!!!!!
1081 003162  000000                  TSTLOC: .WORD   0
1082 003164                                  .BLKW   20.
1083                                         ;FPP REGISTER DEFINITIONS
1084         000000                          AC0= %0
1085         000001                          AC1= %1
1086         000002                          AC2= %2
1087         000003                          AC3= %3
1088         000004                          AC4= %4
1089         000005                          AC5= %5
1090         000006                          AC6= %6
1091         000007                          AC7= %7
1092
1093                                         ;FPP INTERRUPT VECTOR
1094
1095         000244                          FPVEC=244
1096
1097
1098         001000                          STBOT=  1000
1099
1100
1102 003234  123456                  TAB1:   .WORD   123456
1103 003236  000000                          .WORD   000000
1104 003240  000000                          .WORD   0
1105 003242  000001                          .WORD   1
1106 003244  055555                  TAB2:   .WORD   055555
1107 003246  177777                          .WORD   177777
1108 003250  145671                          .WORD   145671
1109 003252  100000                          .WORD   100000
1110 003254  003000                  TAB3:   .WORD   003000
1111 003256  123456                          .WORD   123456
1112 003260  000000                          .WORD   0
1113 003262  000000                          .WORD   0
1114 003264  055555                  TAB4:   .WORD   55555
1115 003266  177777                          .WORD   -1
1116 003270  000000                          .WORD   0
1117 003272  000000                          .WORD   0
1118 003274  043243                  TAB5:   .WORD   43243
1119 003276  000000                          .WORD   0
1120 003300  000000                          .WORD   0
1121 003302  000000                          .WORD   0
1122 003304  162400                  TAB5A:  .WORD   162400
1123 003306  000000                          .WORD   0
1124 003310  000000                          .WORD   0
1125 003312  000000                          .WORD   0
1126 003314  000000                  TAB6:   .WORD   0
```

E3

GLOBAL DATA SECTION

```
1127 003316  000000                          .WORD   0
1128 003320  000000                          .WORD   0
1129 003322  000000                          .WORD   0
1130 003324  047050                TAB6A:    .WORD   47050
1131 003326  010000                          .WORD   10000
1132 003330  000000                          .WORD   0
1133 003332  000000                          .WORD   0
1134 003334  000200                TAB7:     .WORD   200
1135 003336  000000                          .WORD   0
1136 003340  000000                          .WORD   0
1137 003342  000000                          .WORD   0
1138 003344  000200                TAB8:     .WORD   200
1139 003346  000000                          .WORD   0
1140 003350  000000                          .WORD   0
1141 003352  000001                          .WORD   1
1142 003354  000400  000000  000000  TAB9:     .WORD   400,0,0,0
     003362  000000
1143 003364  030000                TAB10:    .WORD   30000
1144 003366  003000                          .WORD   3000
1145 003370  000000                          .WORD   0
1146 003372  000000                          .WORD   0
1147 003374  016400                TAB11:    .WORD   16400
1148 003376  000000                          .WORD   0
1149 003400  000000                          .WORD   0
1150 003402  000000                          .WORD   0
1151 003404  030000  003000  000002  TAB11A:  .WORD   30000,3000,2,0
     003412  000000
1152 003414  016100  000000  000000  TAB12:    .WORD   16100,0,0,1
     003422  000001
1153 003424  016200                TAB13:    .WORD   16200
1154 003426  000000                          .WORD   0
1155 003430  000000                          .WORD   0
1156 003432  000001                          .WORD   1
1157 003434  030000  003000  000000  TAB13B:  .WORD   30000,3000,0,140000
     003442  140000
1158 003444  030000                TAB14:    .WORD   30000
1159 003446  000000                          .WORD   0
1160 003450  000000                          .WORD   0
1161 003452  000000                          .WORD   0
1162 003454  024700                TAB15:    .WORD   24700
1163 003456  000000                          .WORD   0
1164 003460  000000                          .WORD   0
1165 003462  000000                          .WORD   0
1166 003464  025000                TAB16:    .WORD   25000
1167 003466  175363                          .WORD   175363
1168 003470  123456                          .WORD   123456
1169 003472  123456                          .WORD   123456
1170 003474  030000                TAB17:    .WORD   30000
1171 003476  007020                          .WORD   7020
1172 003500  000000  000000                  .WORD   0,0
1173 003504  023456                TAB18:    .WORD   23456
1174 003506  000000                          .WORD   0
1175 003510  000000                          .WORD   0
1176 003512  000001                          .WORD   1
1177 003514  100200  000000  000000  TAB21:    .WORD   100200,0,0,0
     003522  000000
1178 003524  100400  000000  000000  TAB22:    .WORD   100400,0,0,0
```

F3

GLOBAL DATA SECTION

```
                003532  000000
1179            003534  000200  000000  000000  TAB23:  .WORD   200,0,0,1
                003542  000001
1180            003544  062400  000000  000000  TAB24:  .WORD   62400,0,0,0
                003552  000000
1181            003554  001100  000000  000000  TAB25:  .WORD   1100,0,0,0
                003562  000000
1182            003564  100600  000000  000000  TAB26:  .WORD   100600,0,0,0
                003572  000000
1183            003574  001000  000000  000000  TAB27:  .WORD   1000,0,0,0
                003602  000000
1184            003604  000600  000000  000000  TAB28:  .WORD   600,0,0,0
                003612  000000
1185            003614  010100  000000  000000  TAB29:  .WORD   10100,0,0,0
                003622  000000
1186            003624  010100  000000  002000  TAB29A: .WORD   10100,0,2000,0
                003632  000000
1187
1188            003634  000500  000000  000000  TAB30:  .WORD   500,0,0,0
                003642  000000
1189            003644  100400  000000  000000  TAB31:  .WORD   100400,0,0,0
                003652  000000
1190            003654  016000  000000  000000  TAB32:  .WORD   16000,0,0,0
                003662  000000
1191            003664  011600  000000  000000  TAB33:  .WORD   11600,0,0,0
                003672  000000
1192            003674  000640  000000  000000  TAB34:  .WORD   640,0,0,0
                003702  000000
1193            003704  077600  000000  000000  TAB40:  .WORD   77600,0,0,0
                003712  000000
1194            003714  100200  000000  000000  TAB41:  .WORD   100200,0,0,1
                003722  000001
1195            003724  000340  000000  000000  TAB42:  .WORD   340,0,0,0
                003732  000000
1196            003734  000077  177777  177777  TAB43:  .WORD   77,177777,177777,177776
                003742  177776
1197            003744  000577  177777  177777  TAB45:  .WORD   577,-1,-1,-1
                003752  177777
1198            003754  000577  177777  000000  TAB46:  .WORD   577,-1,0,0
                003762  000000
1199            003764  173737  124242  052525  TAB47:  .WORD   173737,124242,052525,12346
                003772  012346
1200            003774  000000  000000  052525  TAB47A: .WORD   0,0,052525,12346
                004002  012346
1201            004004  173737  124242  000000  TAB48:  .WORD   173737,124242,0,0
                004012  000000
1202            004014  000600  000000  000000  TAB49:  .WORD   600,0,0,0
                004022  000000
1203
1204            004024                          START:
                                                ;; LCP/ORION ROUTINE TO SAVE EMTULATOR AND PRIORITY
                004024  005737  004112          EMTSAV: TST     SAV30           ;; FIRST TIME THROUGH ?
                004030  001034                          BNE     VMKOR           ;; BRANCH IF BEEN HERE ALREADY
                004032  032737  000040  000052          BIT     #BIT5,@#52      ;; ARE WE IN UFD MODE ?
                004040  001430                          BEQ     VMKOR           ;; LEAVE IF NOT
                004042  012737  177777  004116          MOV     #-1,UFDFLG      ;; SET UFD FLAG
                004050  032737  000100  000052          BIT     #BIT6,@#52      ;; ARE WE IN QUIET MODE ?
```

G3

GLOBAL DATA SECTION

```
        004056  001403                          BEQ     1$                      ;; BR IF NOT
        004060  012737  177777  004120          MOV     #-1,UQUIET              ;; SET QUIET MODE
        004066  104042                  1$:     EMT     42                      ;; GET ADDRESS OF XXDP DCA TABLE
        004070  005060  000042                  CLR     42(R0)                  ;; CLR XXDP+ "DRSERR"
        004074  013737  000030  004112          MOV     30,SAV30                ;; SAVE EMULATOR ADDRESS
        004102  013737  000032  004114          MOV     32,SAV32                ;; SAVE EMULATOR PRIORITY LEVEL
        004110  000404                          BR      VMKOR                   ;; GET AROUND TAG AREA
        004112  000000                  SAV30:  .WORD   0                       ;; PUT EMULATOR INFO HERE
        004114  000000                  SAV32:  .WORD   0                       ;; PUT PRIORITY LOCATION        HERE
        004116  000000                  UFDFLG: .WORD   0                       ;; USER FRIENDLY MODE FLAG
        004120  000000                  UQUIET: .WORD   0                       ;; UFD QUIET MODE FLAG
        004122                          VMKOR:
                                        ;*******************************   ****************************************
   1205 004122                          1$:
                                        .SBTTL  INITIALIZE THE COMMON
                                        ;;CLEAR THE COMMON TAGS ($CMT       ZA
        004122  012706  001100                  MOV     #$CMTAG,R6          ;FIRST LOCATION TO BE CLEARED
        004126  005026                          CLR     (R6)+               ;;CLEAR MEMORY LOCATION
        004130  022706  001140                  CMP     #SWR,R6 ;;DONE?
        004134  001374                          BNE     .-6                 ;;LOOP BACK IF NO
        004136  012706  001100                  MOV     #STACK,SP           ;;SETUP THE STACK POINTER
                                        ;;INITIALIZE A FEW VECTORS
        004142  012737  140474  000020          MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
        004150  012737  000340  000022          MOV     #340,@#IOTVEC+2 ;;LEVEL 7
        004156  012737  140776  000030          MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
        004164  012737  000340  000032          MOV     #340,@#EMTVEC+2 ;;LEVEL 7
        004172  012737  143522  000034          MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
        004200  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
        004206  012737  143604  000024          MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
        004214  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;LEVEL 7
        004222  013737  140406  140400          MOV     $ENDCT,$EOPCT   ;;SETUP END-OF-PROGRAM COUNTER
        004230  005037  001164                  CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
        004234  005037  001166                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
        004240  112737  000001  001115          MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
        004246  012737  004246  001106          MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
        004254  012737  004254  001110          MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
                                        ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
                                        ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
        004262  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
        004266  012737  004322  000004          MOV     #30000$,@#ERRVEC        ;;SET UP ERROR VECTOR
        004274  012737  177570  001140          MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
        004302  012737  177570  001142          MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
        004310  022777  177777  174622          CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
        004316  001012                          BNE     30002$          ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                                        ;;AND  THE HARDWARE SWR IS NOT = -1
        004320  000403                          BR      30001$          ;;BRANCH IF NO TIMEOUT
        004322  012716  004330          30000$: MOV     #30001$,(SP)    ;;SET UP FOR TRAP RETURN
        004326  000002                          RTI
        004330  012737  000176  001140  30001$: MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
        004336  012737  000174  001142          MOV     #DISPREG,DISPLAY
        004344  012637  000004          30002$: MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR
        004350  005037  001206                  CLR     $PASS           ;;CLEAR PASS COUNT
        004354  132737  000200  001221          BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
        004362  001403                          BEQ     30003$          ;;YES,USE NON-APT SWITCH
        004364  012737  001222  001140          MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
        004372                  30003$:
   1206 004372  013737  004116  004120          MOV     UFDFLG,UQUIET   ;ABORT IN UFD ON ERROR
```

INITIALIZE THE COMMON TAGS

```
1207 004400  012737  137542  000004          MOV     #TOUT,@#ERRVEC  ;POINT TO TIMEOUT ROUTINE
1208 004406  012737  000340  000006          MOV     #340,@#ERRVEC+2 ;AT PRIORITY 7
1209 004414  012737  137012  000114          MOV     #RAMPAR,@#114   ;POINT PARITY ABORT
1210 004422  012737  000340  000116          MOV     #340,@#116      ;AT PRIORITY7
1211 004430  012737  137724  000250          MOV     #MMUTRP,@#250   ;POINT MMU TRAP VECTOR
1212 004436  012737  000340  000252          MOV     #340,@#252      ;
1213 004444  005037  177766                  CLR     @#177766        ;CLEAR CPU ERROR REGISTER
1214 004450  032737  000100  000052          BIT     #BIT06,@#52     ;IN UFD QUIET MODE ?
1215 004456  001130                          BNE     LOOP            ;IF SO, SKIP PRINTOUT
1216 004460  012701  004472                  MOV     #.+12,R1        ;STORE LOCATION POINTER
1217 004464  012711  177777                  MOV     #-1,(R1)        ;MAKE SURE TO PRINT NOT ONLY AT FIRST TIME
1218                                 .SBTTL  TYPE PROGRAM NAME
                                     ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
     004470  005227  177777                  INC     #-1             ;;FIRST TIME?
     004474  001052                          BNE     30004$          ;;BRANCH IF NO
     004476  022737  140440  000042          CMP     #$ENDAD,@#42    ;;ACT-11?
     004504  001446                          BEQ     30004$          ;;BRANCH IF YES
     004506  104401  004554                  TYPE    ,30005$         ;;TYPE ASCIZ STRING
                                 .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
     004512  005737  000042                  TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
     004516  001012                          BNE     30006$          ;;BRANCH IF YES
     004520  123727  001220  000001          CMPB    $ENV,#1         ;;ARE WE RUNNING UNDER APT?
     004526  001406                          BEQ     30006$          ;;BRANCH IF YES
     004530  023727  001140  000176          CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
     004536  001005                          BNE     30007$          ;;BRANCH IF NO
     004540  104406                          GTSWR                   ;;GET SOFT-SWR SETTINGS
     004542  000403                          BR      30007$
     004544  112737  000001  001134  30006$: MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
     004552                       30007$:
     004552  000423                          BR      30004$          ;;GET OVER THE ASCIZ
                                 ;;30005$:   .ASCIZ  <CRLF>* KDJ11-B CPU DIAGNOSTIC - COKDAFO *<CRLF>
     004622                       30004$:
1219 004622  000240                      NOP                         ; debug aid
1220 004624  123727  001220  000001      CMPB    $ENV,#1             ; running under APT?
1221 004632  001020                      BNE     20$                 ; default no-APT initialization
1222 004634  013700  001224              MOV     $USWR,R0            ; work copy pass calculation
1223 004640  005700                      TST     R0                  ; if = 0 default value
1224 004642  001420                      BEQ     25$                 ; setup default value
1225
1226 004644  042700  000017              BIC     #17,R0              ; clear low order nibble
1227 004650  000241                      CLC                         ; assure no unknowns
1228 004652  006000                      ROR     R0                  ; 4 rotates = divide
1229 004654  006000                      ROR     R0                  ; by 16 (=pass time)
1230 004656  006000                      ROR     R0                  ; this area subroutined
1231 004660  006000                      ROR     R0                  ; with general purpose
1232 004662  005700                      TST     R0                  ; divide, this test to
1233 004664  001413                      BEQ     30$                 ; determine skip altogether
1234
1235 004666  010037  003032              MOV     R0,CCHPAS           ;residue = no. of desired passes
1236 004672  000413                      BR      35$                 ; continue on
1237 004674  012737  177777  003032  20$: MOV     #-1,CCHPAS          ; largest number no apt mode??
1238 004702  000407                      BR      35$
1239 004704  012737  000001  003032  25$: MOV     #1,CCHPAS           ; normal default = 1
1240 004712  000403                      BR      35$
1241 004714  012737  000000  003032  30$: MOV     #0,CCHPAS           ; no cache tests included
1242 004722  000240                  35$: nop                        ; debug aid
1243
```

GET VALUE FOR SOFTWARE SWITCH REGISTER

```
1244 004724  032737  000200  000052          BIT     #BIT07,@#52     ;UFD MODE?
1245 004732  001002                          BNE     LOOP            ;IF YES, BRANCH
1246 004734  004737  137070                  JSR     PC,Q22SIZ       ;SIZE FOR Q22BE
1247 004740                          LOOP:
1248                                 .DSABLE AMA
1250                                 ;;*****************************************************************
     004740  000004                  TST1:   SCOPE
1251                                 .SBTTL  BASE INSTRUCTION SET TESTS
1252                                 ;*********************************************************************************
1253                                 ;*********************************************************************************
1254                                 ;                       BEGIN BASE INSTRUCTION SET TESTING
1255                                 ;*********************************************************************************
1256                                 ;*********************************************************************************
1265 004742                          FRSTST:
1266
1267                                 ;TEST BEQ BNE INSTRUCTIONS
1268                                 ;;*****************************************************************
                                     ;THESE TWO INSTRUCTIONS ARE FUNDAMENTAL TO RECOGNIZING ERROR CONDITIONS
1269 004742  000277                          SCC
1270 004744  000244                          CLZ                     ;CC=0100 - Z BIT CLEARED
1271 004746  001401                          BEQ     1$              ;*TEST INSTR -TRY TO CAUSE A BEQ ERROR
1272 004750  001001                          BNE     2$              ;BRANCH IF GOOD
1273                                                                 ;THE Z FLAG DIDNT CLEAR OR BRANCH FAILED.
1274                                                                 ;FAILURE AT THIS LOCATION
1275                                                                 ;COULD MEAN A BUS PROBLEM, MICRO  )DE PROBLEM
1276                                                                 ;CONDITION CODE PROBLEM OR JUST ABOUT ANYTHING
1277                                                                 ;ELSE.
1278 004752  104001                  1$:     ERROR   +1              ;CPU ERROR
1279 004754  000257                  2$:     CCC                     ;
1280 004756  000264                          SEZ                     ;COND CODES = 0100 (ZERO)
1281 004760  001001                          BNE     3$              ;*TEST INSTR* TRY TO BRANCH ON ZERO FLAG
1282 004762  001401                          BEQ     4$              ;*TEST INSTR* BRANCH IF GOOD
1283                                                                 ;BRANCH FAILURE WITH Z BIT SET
1284 004764  104001                  3$:     ERROR   +1              ;CPU ERROR
1285 004766                          4$:                             ;END OF TEST
1295 004766                          M2:
1296
1297                                 ;       TEST BRANCH ON CARRY
1298                                 ;;*****************************************************************
                                     ;THIS IS A TEST TO SEE IF THE MODULE FORM ANTICIPATED IS FEASIBLE.
1299 004766  000257                          CCC                     ;CC=0000
1300 004770  103001                          BCC     2$              ;*TEST INSTR*
1301                                                                 ;BRANCH CARRY CLEAR FAILED
1302 004772  104001                  1$:     ERROR   +1              ;CPU ERROR
1303 004774  000261                  2$:     SEC                     ;CC=1111
1304 004776  103401                          BCS     4$              ;*TFST INSTR*
1305                                                                 ; BRANCH CARRY SET FAILED
1306 005000  104001                  3$:     ERROR   +1              ;CPU ERROR
1307 005002                          4$:
1308
1311 005002                          M3:
1312
1313                                 ;       TEST DATA PATHS
1314 005002  005000                          CLR     R0
1315 005004  001005                          BNE     1$              ;TRY TO INSURE WE ARE TESTING
1316                                                                 ;THE DATA PATH AND NOT THE "CLR R0" INSTRUCTION
1317 005006  005010                          C R     (R0)            ;FORCE LOCATION TO ZERO
```

BASE INSTRUCTION SET TESTS

```
1318 005010  001003                          BNE     1$              ;TRY TO INSURE 0=0
1319 005012  005737  000000                  TST     @#0             ;AGAIN, TRY TO INSURE THAT 0=0
1320 005016  001401                          BEQ     2$              ;BRANCH IF GOOD
1321                                                                 ;LOCATION 0 NOT SETUP PROPERLY
1322 005020  104001                  1$:     ERROR   +1              ;CPU ERROR
1323 005022                          2$:
1326 005022                          M4:
1327
1328                                 ;       TEST DATA PATHS   ONES AND ZEROS
1329 005022  012737  125252  000000          MOV     #125252,@#0             ;0=125252
1330 005030  022737  125252  000000          CMP     #125252,@#0             ;SEE IF DATA MADE IT
1331 005036  001401                          BEQ     2$                      ;BRANCH IF IF DATA IS GOOD
1332                                                                         ;ERROR! EITHER THE BUS IS BAD,
1333                                                                         ;OR THE MOV OR COMPARE
1334                                                                         ;INSTRUCTIONS FAILED
1335 005040  104001                  1$:     ERROR   +1              ;CPU ERROR
1336 005042                          2$:                                     ;END OF TEST
1337
1338                                 ;
1341 005042                          M5:
1342
1343                                 ;       TEST DATA PATHS - DATA 0'S AND 1'S
1344 005042  012737  052525  000000          MOV     #052525,@#0             ;SETUP DATA
1345 005050  023727  000000  052525          CMP     @#0,#052525             ; TEST FOR CORRECT DATA
1346 005056  001401                          BEQ     2$
1347 005060  104001                  1$:     ERROR   +1              ;CPU ERROR
1348 005062                          2$:
1349                                 ;
1352 005062                          M6:
1353                                 ;       TEST DATA PATHS - 1'S
1354 005062  005037  000000                  CLR     @#0
1355 005066  005137  000000                  COM     @#0             ;SET UP MEMORY LOCATION 0 = 111111
1356 005072  023727  000000  177777          CMP     @#0,#177777     ; TEST DATA
1357 005100  001401                          BEQ     2$              ;BRANCH IF NO ERROR
1358 005102  104001                  1$:     ERROR   +1              ;CPU ERROR
1359 005104                          2$:
1360
1361                                 ;
1364 005104                          GPROTS:
1365                                 ;       R0 BIT TESTS
1366 005104  012700  177777                  MOV     #177777,R0              ;R0=177777
1367 005110  020027  177777                  CMP     R0,#177777              ;DOES R0=177777
1368 005114  001401                          BEQ     1$                      ;YES GO ON
1369                                                                         ;NO GO TO ERROR
1370 005116  104001                          ERROR   +1              ;CPU ERROR
1371 005120  005000                  1$:     CLR     R0                      ;R0=0
1372 005122  020027  000000                  CMP     R0,#0                   ;DOES R0=0
1373 005126  001401                          BEQ     2$                      ;YES GO ON
1374                                                                         ;NO GO TO ERROR
1375 005130  104001                          ERROR   +1              ;CPU ERROR
1376 005132  012700  125252          2$:     MOV     #125252,R0              ;R0=125252
1377 005136  020027  125252                  CMP     R0,#125252              ;DOES R0=125252
1378 005142  001401                          BEQ     3$                      ;YES GO ON
1379                                                                         ;NO GO TO ERROR
1380 005144  104001                          ERROR   +1              ;CPU ERROR
1381 005146  012700  052525          3$:     MOV     #52525,R0               ;R0=52525
1382 005152  020027  052525                  CMP     R0,#52525               ;DOES R0=52525
```

BASE INSTRUCTION SET TESTS

```
1383 005156  001401                        BEQ     4$                      ;YES GO ON
1384                                                                       ;NO GO TO ERROR
1385 005160  104001                        ERROR   +1              ;CPU ERROR
1386 005162                          4$:
1387                                 ;
1390 005162                          GPR1TS:
1391                                 ;       R1 BIT TESTS
1392 005162  012701  177777                MOV     #177777,R1              ;R1=177777
1393 005166  020127  177777                CMP     R1,#177777              ;DOES R1=177777
1394 005172  001401                        BEQ     1$                      ;YES GO ON
1395                                                                       ;NO GO TO ERROR
1396 005174  104001                        ERROR   +1              ;CPU ERROR
1397 005176  005001                  1$:   CLR     R1                      ;R1=0
1398 005200  020127  000000                CMP     R1,#0                   ;DOES R1=0
1399 005204  001401                        BEQ     2$                      ;YES GO ON
1400                                                                       ;NO GO TO ERROR
1401 005206  104001                        ERROR   +1              ;CPU ERROR
1402 005210  012701  125252          2$:   MOV     #125252,R1              ;R1=125252
1403 005214  020127  125252                CMP     R1,#125252              ;DOES R1=125252
1404 005220  001401                        BEQ     3$                      ;YES GO ON
1405                                                                       ;NO GO TO ERROR
1406 005222  104001                        ERROR   +1              ;CPU ERROR
1407 005224  012701  052525          3$:   MOV     #52525,R1               ;R1=52525
1408 005230  020127  052525                CMP     R1,#52525               ;DOES R1=52525
1409 005234  001401                        BEQ     4$                      ;YES GO ON
                                                                           ;NO GO TO ERROR
     005236  104001                        ERROR   +1              ;CPU ERROR
     005240                          4$:
                                     ;
1410 005240                          GPR2TS:
1411                                 ;       R2 BIT TESTS
1418 005240  012702  177777                MOV     #177777,R2              ;R2=177777
1419 005244  020227  177777                CMP     R2,#177777              ;DOES R2=177777
1420 005250  001401                        BEQ     1$                      ;YES GO ON
1421                                                                       ;NO GO TO ERROR
1422 005252  104001                        ERROR   +1              ;CPU ERROR
1423 005254  005002                  1$:   CLR     R2                      ;R2=0
1424 005256  020227  000000                CMP     R2,#0                   ;DOES R2=0
1425 005262  001401                        BEQ     2$                      ;YES GO ON
1426                                                                       ;NO GO TO ERROR
1427 005264  104001                        ERROR   +1              ;CPU ERROR
1428 005266  012702  125252          2$:   MOV     #125252,R2              ;R2=125252
1429 005272  020227  125252                CMP     R2,#125252              ;DOES R2=125252
1430 005276  001401                        BEQ     3$                      ;YES GO ON
1431                                                                       ;NO GO TO ERROR
1432 005300  104001                        ERROR   +1              ;CPU ERROR
1433 005302  012702  052525          3$:   MOV     #52525,R2               ;R2=52525
1434 005306  020227  052525                CMP     R2,#52525               ;DOES R2=52525
1435 005312  001401                        BEQ     4$                      ;YES GO ON
1436                                                                       ;NO GO TO ERROR
1437 005314  104001                        ERROR   +1              ;CPU ERROR
1438 005316                          4$:
1439                                 ;
1442 005316                          GPR3TS:
1443                                 ;       R3 BIT TESTS
1444 005316  012703  177777                MOV     #177777,R3              ;R3=177777
1445 005322  020327  177777                CMP     R3,#177777              ;DOES R3=177777
```

L3

BASE INSTRUCTION SET TESTS

```
1446 005326  001401                        BEQ     1$                      ;YES GO ON
1447                                                                       ;NO GO TO ERROR
1448 005330  104001                        ERROR   +1              ;CPU ERROR
1449 005332  005003                1$:     CLR     R3                      ;R3=0
1450 005334  020327  000000              CMP     R3,#0                   ;DOES R3=0
1451 005340  001401                        BEQ     2$                      ;YES GO ON
1452                                                                       ;NO GO TO ERROR
1453 005342  104001                        ERROR   +1              ;CPU ERROR
1454 005344  012703  125252        2$:     MOV     #125252,R3              ;R3=125252
1455 005350  020327  125252              CMP     R3,#125252              ;DOES R3=125252
1456 005354  001401                        BEQ     3$                      ;YES GO ON
1457                                                                       ;NO GO TO ERROR
1458 005356  104001                        ERROR   +1              ;CPU ERROR
1459 005360  012703  052525        3$:     MOV     #52525,R3               ;R3=52525
1460 005364  020327  052525              CMP     R3,#52525               ;DOES R3=52525
1461 005370  001401                        BEQ     4$                      ;YES GO ON
1462                                                                       ;NO GO TO ERROR
1463 005372  104001                        ERROR   +1              ;CPU ERROR
1464 005374                        4$:
1465                               ;
1468 005374                        GPR4TS:
1469                               ;       R4 BIT TESTS
1470 005374  012704  177777              MOV     #177777,R4              ;R4=177777
1471 005400  020427  177777              CMP     R4,#177777              ;DOES R4=177777
1472 005404  001401                        BEQ     1$                      ;YES GO ON
1473                                                                       ;NO GO TO ERROR
1474 005406  104001                        ERROR   +1              ;CPU ERROR
1475 005410  005004                1$:     CLR     R4                      ;R4=0
1476 005412  020427  000000              CMP     R4,#0                   ;DOES R4=0
1477 005416  001401                        BEQ     2$                      ;YES GO ON
1478                                                                       ;NO GO TO ERROR
1479 005420  104001                        ERROR   +1              ;CPU ERROR
1480 005422  012704  125252        2$:     MOV     #125252,R4              ;R4=125252
1481 005426  020427  125252              CMP     R4,#125252              ;DOES R4=125252
1482 005432  001401                        BEQ     3$                      ;YES GO ON
1483                                                                       ;NO GO TO ERROR
1484 005434  104001                        ERROR   +1              ;CPU ERROR
1485 005436  012704  052525        3$:     MOV     #52525,R4               ;R4=52525
1486 005442  020427  052525              CMP     R4,#52525               ;DOES R4=52525
1487 005446  001401                        BEQ     4$                      ;YES GO ON
1488                                                                       ;NO GO TO ERROR
1489 005450  104001                        ERROR   +1              ;CPU ERROR
1490 005452                        4$:
1491                               ;
1494 005452                        GPR5TS:
1495                               ;       R5 BIT TESTS
1496 005452  012705  177777              MOV     #177777,R5              ;R5=177777
1497 005456  020527  177777              CMP     R5,#177777              ;DOES R5=177777
1498 005462  001401                        BEQ     1$                      ;YES GO ON
1499                                                                       ;NO GO TO ERROR
1500 005464  104001                        ERROR   +1              ;CPU ERROR
1501 005466  005005                1$:     CLR     R5                      ;R5=0
1502 005470  020527  000000              CMP     R5,#0                   ;DOES R5=0
1503 005474  001401                        BEQ     2$                      ;YES GO ON
1504                                                                       ;NO GO TO ERROR
1505 005476  104001                        ERROR   +1              ;CPU ERROR
1506 005500  012705  125252        2$:     MOV     #125252,R5              ;R5=125252
```

M3

PASE INSTRUCTION SET TESTS

```
1507 005504  020527  125252                CMP     R5,#125252              ;DOES R5=125252
1508 005510  001401                        BEQ     3$                      ;YES GO ON
1509                                                                       ;NO GO TO ERROR
1510 005512  104001                        ERROR   +1              ;CPU ERROR
1511 005514  012705  052525        3$:     MOV     #52525,R5               ;R5=52525
1512 005520  020527  052525                CMP     R5,#52525               ;DOES R5=52525
1513 005524  001401                        BEQ     4$                      ;YES GO ON
1514                                                                       ;NO GO TO ERROR
1515 005526  104001                        ERROR   +1              ;CPU ERROR
1516 005530                        4$:
1517                                       ;
1520 005530                        GPR6TS:
1521                                       ;       R6 BIT TESTS
1522 005530  012706  177777                MOV     #177777,R6              ;R6=177777
1523 005534  020627  177777                CMP     R6,#177777              ;DOES R6=177777
1524 005540  001401                        BEQ     1$                      ;YES GO ON
1525                                                                       ;NO GO TO ERROR
1526 005542  104001                        ERROR   +1              ;CPU ERROR
1527 005544  005006                1$:     CLR     R6                      ;R6=0
1528 005546  020627  000000                CMP     R6,#0                   ;DOES R6=0
1529 005552  001401                        BEQ     2$                      ;YES GO ON
1530                                                                       ;NO GO TO ERROR
1531 005554  104001                        ERROR   +1              ;CPU ERROR
1532 005556  012706  125252        2$:     MOV     #125252,R6              ;R6=125252
1533 005562  020627  125252                CMP     R6,#125252              ;DOES R6=125252
1534 005566  001401                        BEQ     3$                      ;YES GO ON
1535                                                                       ;NO GO TO ERROR
1536 005570  104001                        ERROR   +1              ;CPU ERROR
1537 005572  012706  052525        3$:     MOV     #52525,R6               ;R6=52525
1538 005576  020627  052525                CMP     R6,#52525               ;DOES R6=52525
1539 005602  001401                        BEQ     4$                      ;YES GO ON
1540                                                                       ;NO GO TO ERROR
1541 005604  104001                        ERROR   +1              ;CPU ERROR
1542 005606  012706  001000        4$:     MOV     #STBOT,R6               ;RESTORE SP
1543
1544                                       ;
1547 005612                        PSWBTS:
1548                                       ;       PSW LOW BYTE BIT TESTS
1549 005612  012737  000377  177776        MOV     #377,@#177776           ;PS=357 T BIT SHOULDN'T SET
1550 005620  022737  000357  177776        CMP     #357,@#177776           ;DOES PS=357
1551 005626  001401                        BEQ     1$                      ;YES GO ON
1552                                                                       ;NO GO TO ERROR
1553 005630  104001                        ERROR   +1              ;CPU ERROR
1554 005632  005037  177776        1$:     CLR     @#177776                ;PS=0
1555 005636  022737  000000  177776        CMP     #0,@#177776             ;DOES PS=0
1556 005644  001401                        BEQ     2$                      ;YES GO ON
1557                                                                       ;NO GO TO ERROR
1558 005646  104001                        ERROR   +1              ;CPU ERROR
1559 005650  012737  000105  177776  2$:   MOV     #105,@#177776           ;PS=105
1560 005656  022737  000105  177776        CMP     #105,@#177776           ;DOES PS=105
1561 005664  001401                        BEQ     3$                      ;YES GO ON
1562                                                                       ;NO GO TO ERROR
1563 005666  104001                        ERROR   +1              ;CPU ERROR
1564 005670  012737  000252  177776  3$:   MOV     #252,@#177776           ;PS=252
1565 005676  022737  000252  177776        CMP     #252,@#177776           ;DOES PS=252
1566 005704  001401                        BEQ     4$                      ;YES GO ON
1567                                                                       ;NO GO TO ERROR
```

BASE INSTRUCTION SET TESTS

```
1568 005706  104CJ1                    ERROR   +1              ;CPU ERROR
1569 005710                    4$:
1579
1580 005710                    MSPO:
1581
1582                           ;       TEST SINGLE OPERAND INSTRUCTIONS  MODE 0
1583                           ;;*****************************************************************
                               ;THE INC, COM, CLR, AND DECREMENT INSTRUCTIONS ARE VERIFED.
1584 005710  005004                    CLR     R4                      ;INITIALIZE R4 WITH DATA
1585 005712  005104                    COM     R4                      ;
1586 005714  005004                    CLR     R4                      ;*TEST INSTRUCTION
1587 005716  001401                    BEQ     2$                      ;BRANCH IF R4 CLEARED
1588 005720  104001            1$:     ERROR   +1              ;CPU ERROR
1589 005722  005104            2$:     COM     R4                      ;*TEST COMPLIMENT INSTRUCTION
1590 005724  005204                    INC     R4                      ;*TEST INCREMENT INSTRUCTION
1591 005726  001401                    BEQ     4$                      ;BRANCH IF R4 =0
1592                                                                   ;COMPLIMENT OR INCREMENT FAILED
1593 005730  104001            3$:     ERROR   +1              ;CPU ERROR
1594 005732                    4$:
1595
1596                           ;
1599 005732                    MSPB:
1600
1601                           ;       TEST SINGLE OPS - EVEN BYTE OF CLRB, DECB, AND COMB
1602 005732  005004                    CLR     R4
1603 005734  005104                    COM     R4                      ;SETUP TEST REGISTER
1604 005736  105004                    CLRB    R4                      ;*TEST CLEAR BYTE INSTRUCTION
1605 005740  001401                    BEQ     2$                      ;BRANCH IF GOOD
1606                                                                   ;CLEAR EVEN BYTE FAILED
1607 005742  104001            1$:     ERROR   +1              ;CPU ERROR
1608 005744  105304            2$:     DECB    R4              ;*TEST DECREMENT BYTE
1609 005746  100002                    BPL     3$              ;DECREMENT BYTE FAILED
1610 005750  105104                    COMB    R4              ;*TEST COMPLIMENT BYTE
1611 005752  001401                    BEQ     4$              ;BRANCH IF GOOD
1612                                                           ;COMPLIMENT OR DECREMENT FAILED TO WORK
1613 005754  104001            3$:     ERROR   +1              ;CPU ERROR
1614 005756                    4$:
1615
1616                           ;
1619 005756                    MSPC:
1620                           ;       TEST SINGLE OPS - MODE 1 CLRB, COMB, AND INCB
1621 005756  005004                    CLR     R4
1622 005760  005014                    CLR     (R4)
1623 005762  005114                    COM     (R4)            ;SETUP TEST DATA
1624 005764  005014                    CLR     (R4)            ;*TEST INSTRUCTION
1625 005766  001401                    BEQ     2$              ;BRANCH IF GOOD
1626                                                           ;MODE 1 FAILED
1627 005770  104001            1$:     ERROR   +1              ;CPU ERROR
1628 005772  005114            2$:     COM     (R4)            ;*TEST INSTRUCTION
1629 005774  001403                    BEQ     3$              ;(0)SHOULD = -1
1630 005776  100002                    BPL     3$              ;
1631 006000  005214                    INC     (R4)            ;*TEST INSTRUCTION
1632 006002  001401                    BEQ     4$              ;BRANCH IF GOOD
1633                                                           ;COM OR INC FAILED TO ALTER LOC 0 CORRECTLY
1634 006004  104001            3$:     ERROR   +1              ;CPU ERROR
1635 006006                    4$:
1636
```

```
                                ;
006006                          MSPD:
                                ;       TEST SINGLE OPS MODE1 EVEN BYTE CLRB,COMB,INCB
006006  005004                          CLR     R4
006010  005014                          CLR     (R4)
006012  005114                          COM     (R4)            ;SETUP TEST DATA
006014  105014                          CLRB    (R4)            ;*TEST INSTRUCTION
006016  105014                          CLRB    (R4)            ;*TEST INSTRUCTION
006020  001401                          BEQ     2$              ;BRANCH IF GOOD
                                                                ;CLEAR (0) EVEN BYTE FAILED
006022  104001                  1$:     ERROR   +1              ;CPU ERROR
006024  105214                  2$:     INCB    (R4)            ;*TEST INSTRUCTION
006026  100405                          BMI     3$              ; TEST FLAGS
006030  001404                          BEQ     3$
006032  105114                          COMB    (R4)            ;*TEST INSTRUCTION
006034  105214                          INCB    (R4)
006036  105214                          INCB    (R4)
006040  001401                          BEQ     4$              ;BRANCH IF GOOD
                                                                ;COMB OR INCB FAILED
006042  104001                  3$:     ERROR   +1              ;CPU ERROR
006044                          4$:
                                ;
006044                          MSPEO:
                                ;       TEST SINGLE OPS - ODD BYTE - CLRB, COMB, DECB
006044  005004                          CLR     R4
006046  005014                          CLR     (R4)
006050  005114                          COM     (R4)            ;SETUP TEST DATA
006052  005204                          INC     R4              ;POINT TO ODD BYTE
006054  105014                          CLRB    (R4)            ;*TEST INSTRUCTION
006056  001401                          BEQ     1$              ;BRANCH IF GOOD
                                                                ;CLEAR ODD BYTE FAILED
006060  104001                          ERROR   +1              ;CPU ERROR
006062  005304                  1$:     DEC     R4              ;POINT TO EVEN BYTE
006064  005214                          INC     (R4)            ;LOC 0=1 0
006066  005204                          INC     R4              ;POINT TO ODD BYTE
006070  105114                          COMB    (R4)            ;*TEST INSTRUCTION
006072  105214                          INCB    (R4)            ;LOC 0=-1 0
006074  100003                          BPL     2$              ;BRANCH IF ERROR
006076  001402                          BEQ     2$              ;
006100  105214                          INCB    (R4)            ;*TEST INSTRUCTION
006102  00140'                         BEQ     3$              ;BRANCH IF GOOD
                                                                ;MODE 1, ODD BYTE FAILED
006104  104001                  2$:     ERROR   +1              ;CPU ERROR
006106                          3$:
                                ;
006106                          MSPF:
                                ;       TEST SINGLE OP - MODE 2 - CLR, COM, INC
006106  005004                          CLR     R4
006110  105104                          COMB    R4
006112  005204                          INC     R4              ;R4=400
006114  005014                          CLR     (R4)            ;400=0
006116  005114                          COM     (R4)            ;400=-1
006120  005024                          CLR     (R4)+           ;*TEST INSTRUCTION
```

BASE INSTRUCTION SET TESTS

```
1700 006122  001401                    BEQ     1$              ;BRANCH IF GOOD
1701                                                           ;MODE 2 CLEAR FAILED
1702 006124  104001                    ERROR   +1              ;CPU ERROR
1703 006126  005304            1$:     DEC     R4
1704 006130  005304                    DEC     R4              ;R4=400
1705 006132  005124                    COM     (R4)+           ;*TEST INSTRUCTION
1706 006134  100004                    BPL     2$              ;BRANCH IF FAILURE
1707 006136  005304                    DEC     R4
1708 006140  005304                    DEC     R4              ;R4=400
1709 006142  005224                    INC     (R4)+           ;*TEST INSTRUCTION
1710 006144  001401                    BEQ     3$              ;BRANCH IF GOOD
1711                                                           ;MODE 2 FAILURE
1712 006146  104001            2$:     ERROR   +1              ;CPU ERROR
1713 006150                    3$:
1714
1715                           ;
1718 006150                    MSPG:
1719
1720                           ;       TEST CLRB, COMB, DECB, MODE 2 - EVEN BYTE
1721 006150  005004                    CLR     R4
1722 006152  105104                    COMB    R4
1723 006154  005204                    INC     R4              ;R4=400
1724 006156  005014                    CLR     (R4)
1725 006160  005114                    COM     (R4)            ;400=-1
1726 006162  105024                    CLRB    (R4)+           ;*TEST INSTRU_  JN
1727 006164  001401                    BEQ     1$              ;BRANCH IF GOOD
1728                                                           ;MODE 2 EVEN BYTE FAILED
1729 006166  104001                    ERROR   +1              ;CPU ERROR
1730 006170  005304            1$:     DEC     R4
1731 006172  105324                    DECB    (R4)+           ;*TEST INSTRUCTION
1732 006174  100003                    BPL     2$              ;BRANCH IF BAD
1733 006176  005304                    DEC     R4              ;POINT TO EVEN BYTE
1734 006200  105124                    COMB    (R4)+           ;*TEST INSTRUCTION
1735 006202  001401                    BEQ     3$              ;BRANCH IF GOOD
1736                                                           ;MODE 2, EVEN BYTE FAILED
1737 006204  104001            2$:     ERROR   +1              ;CPU ERROR
1738 006206                    3$:
1739
1740                           ;
1743 006206                    MSPH:
1744
1745                           ;       TEST CLRB, COMB, INCB MODE 2 - ODD BYTE
1746 006206  005004                    CLR     R4
1747 006210  105104                    COMB    R4
1748 006212  005204                    INC     R4              ;R4=400
1749 006214  005014                    CLR     (R4)
1750 006216  005114                    COM     (R4)            ;400=-1 -1
1751 006220  005214                    INC     (R4)            ;POINT TO ODD BYTE
1752 006222  105024                    CLRB    (R4)+           ;*TEST INSTRUCTION
1753 006224  001401                    BEQ     1$              ;BRANCH IF GOOD
1754                                                           ;MODE 2,ODD BYTE FAILED
1755 006226  104001                    ERROR   +1              ;CPU ERROR
1756 006230  005304            1$:     DEC     R4              ;POINT TO ODD BYTE
1757 006232  005304                    DEC     R4
1758 006234  105224                    INCB    (R4)+           ;400=1 0
1759 006236  105124                    COMB    (R4)+           ;*TEST INSTRUCTION
1760 006240  100003                    BPL     2$              ;BRANCH IF MODE 2 FAILED
```

BASE INSTRUCTION SET TESTS

```
1761 006242  005304                    DEC     R4              ;POINT TO ODD BYTE
1762 006244  105224                    INCB    (R4)+
1763 006246  001401                    BEQ     3$              ;BRANCH IF GOOD
1764                                                           ;MODE 2,ODD BYTE FAILED
1765 006250  104001            2$:     ERROR   +1              ;CPU ERROR
1766 006252                    3$:
1767
1768                           ;
1771 006252                    MSPI:
1772
1773                           ;       TEST CLR, COM, INC   MODE 3
1774 006252  005004                    CLR     R4              ;
1775 006254  005014                    CLR     (R4)            ;0=0
1776 006256  105114                    COMB    (R4)            ;
1777 006260  005214                    INC     (R4)            ;0=400
1778 006262  005034                    CLR     @(R4)+          ;*TEST INSTRUCTION
1779 006264  001401                    BEQ     1$              ;BRANCH IF GOOD
1780                                                           ;MODE 3 FAILED, 400 SHOULD=0
1781 006266  104001                    ERROR   +1              ;CPU ERROR
1782 006270  005304            1$:     DEC     R4
1783 006272  005304                    DEC     R4              ;R4=0
1784 006274  005134                    COM     @(R4)+          ;*TEST INSTRUCTION
1785 006276  100004                    BPL     2$              ;BRANCH IF BAD
1786 006300  005304                    DEC     R4
1787 006302  005304                    DEC     R4              ;REPOSITION POINTER
1788 006304  005234                    INC     @(R4)+          ;*TEST INSTRUCTION
1789 006306  001401                    BEQ     3$              ;BRANCH IF GOOD
1790                                                           ;MODE 3 FAILED
1791 006310  104001            2$:     ERROR   +1              ;CPU ERROR
1792 006312                    3$:
1793
1794                           ;
1797 006312                    MSPJ:
1798
1799                           ;       TEST CLRB, COMB, INCB   MODE 3, EVEN/ODD BYTE
1800 006312  005004                    CLR     R4              ;R4=0
1801 006314  005001                    CLR     R1
1802 006316  105101                    COMB    R1
1803 006320  005201                    INC     R1              ;R1=400
1804 006322  005011                    CLR     (R1)
1805 006324  005121                    COM     (R1)+           ;400=-1
1806 006326  005011                    CLR     (R1)            ;
1807 006330  105111                    COMB    (R1)            ;402=000 377
1808 006332  005014                    CLR     (R4)            ;
1809 006334  105114                    COMB    (R4)            ;
1810 006336  005214                    INC     (R4)            ;0=400
1811 006340  105034                    CLRB    @(R4)+          ;*TEST INSTRUCTION 400=377 000
1812 006342  001401                    BEQ     1$              ;BRANCH IF MODE 3 EVEN BYTE CLEARED
1813                                                           ; TEST INSTRUCTION FAILED
1814 006344  104001                    ERROR   +1              ;CPU ERROR
1815 006346  005304            1$:     DEC     R4              ;REPOSTION POINTER
1816 006350  005304                    DEC     R4
1817 006352  105134                    COMB    @(R4)+          ;*TEST INSTRUCTION
1818 006354  005304                    DEC     R4
1819 006356  005304                    DEC     R4              ;REPOSITION POINTER
1820 006360  105234                    INCB    @(R4)+          ;*TEST INSTRUCTION
1821 006362  001401                    BEQ     3$              ;BRANCH IF GOOD
```

BASE INSTRUCTION SET TESTS

```
1822                                                    ;MODE 3, EVEN B    FAILED
1823 006364  104001          2$:    ERROR   +1          ;CPU ERROR
1824 006366  005304          3$:    DEC     R4
1825 006370  005304                 DEC     R4
1826 006372  005214                 INC     (R4)        ;R4=401
1827 006374  105234                 INCB    @(R4)+      ;*TEST INSTRUCTION
1828 006376  001004                 BNE     4$          ;BRANCH IF 402 NEQ 0
1829 006400  005304                 DEC     R4
1830 006402  005304                 DEC     R4          ;R4=401
1831 006404  105034                 CLRB    @(R4)+      ;401=0
1832 006406  001401                 BEQ     5$          ;BRANCH IF GOOD
1833                                                     ;ODD BYTE FAILED
1834 006410  104001          4$:    ERROR   +1          ;CPU ERROR
1835 006412  005304          5$:    DEC     R4
1836 006414  005304                 DEC     R4          ;R4=401
1837 006416  105134                 COMB    @(R4)+      ;403=377
1838 006420  005304                 DEC     R4
1839 006422  005304                 DEC     R4
1840 006424  105234                 INCB    @(R4)+      ;*TEST INSTRUCTION
1841 006426  001401                 BEQ     7$          ;BRANCH IF GOOD
1842                                                     ;MODE3 ODD BYTE FAILED.
1843 006430  104001          6$:    ERROR   +1          ;CPU ERROR
1844 006432                  7$:
1845
1846                          ;
1849 006432                  MSPL:
1850
1851                          ;      TEST CLR, COM, DEC - MODE 4
1852 006432  005004                 CLR     R4
1853 006434  105104                 COMB    R4
1854 006436  005204                 INC     R4          ;R4=400
1855 006440  005014                 CLR     (R4)        ;
1856 006442  005124                 COM     (R4)+       ;400=-1
1857 006444  005014                 CLR     (R4)        ;
1858 006446  005224                 INC     (R4)+       ;402=1
1859 006450  005044                 CLR     -(R4)       ;*TEST INSTRUCTION
1860 006452  001401                 BEQ     1$          ;BRANCH IF GOOD
1861                                                     ;MODE 4 FAILED
1862 006454  104001                 ERROR   +1          ;CPU ERROR
1863 006456  005344          1$:    DEC     -(R4)       ;*TEST INSTRUCTION 400=-2
1864 006460  005114                 COM     (R4)        ;400=1
1865 006462  001405                 BEQ     2$          ;BRANCH IF BAD
1866 006464  100404                 BMI     2$          ;BRANCH IF BAD
1867 006466  005204                 INC     R4
1868 006470  005204                 INC     R4          ;R4=400
1869 006472  005344                 DEC     -(R4)       ;*TEST INSTRUCTION
1870 006474  001401                 BEQ     3$          ;BRANCH IF GOOD
1871                                                     ;MODE 4 FAILED
1872 006476  104001          2$:    ERROR   +1          ;CPU ERROR
1873 006500                  3$:
1874
1875                          ;
1878 006500                  MSPM:
1879
1880                          ;      TEST COMB, INCB, CLRB - MODE 4, ODD BYTE
1881 006500  005004                 CLR     R4
1882 006502  105104                 COMB    R4          ;
```

BASE INSTRUCTION SET TESTS

```
1883 006504  005204                    INC     R4              ;R4=400
1884 006506  005044                    CLR     -(R4)           ;376=0
1885 006510  105114                    COMB    (R4)
1886 006512  005224                    INC     (R4)+           ;376=001 000
1887 006514  005014                    CLR     (R4)
1888 006516  005124                    COM     (R4)+           ;400=1
1889 006520  005204                    INC     R4              ;R4=403
1890 006522  105044                    CLRB    -(R4)           ; TEST INST. CLEAR ODD BYTE (401)
1891 006524  001401                    BEQ     2$              ;BRANCH IF GOOD
1892                                                           ;MODE 4 BYTE FAILED
1893 006526  104001            1$:     ERROR   +1              ;CPU ERROR
1894 006530  005204            2$:     INC     R4
1895 006532  005204                    INC     R4              ;R4=403
1896 006534  105144                    COMB    -(R4)           ; TEST INST. 401=377
1897 006536  005304                    DEC     R4
1898 006540  005304                    DEC     R4
1899 006542  105244                    INCB    (R4)            ; TEST INST. 401=0
1900 006544  001401                    BEQ     4$              ;BRANCH IF GOOD
1901                                                           ;MODE 4 ODD BYTE FAILED
1902 006546  104001            3$:     ERROR   +1              ;CPU ERROR
1903 006550  105344            4$:     DECB    -(R4)           ;*TEST INST.
1904 006552  001401                    BEQ     6$              ;BRANCH IF GOOD
1905                                                           ;MODE 4 DECREMENT ODD BYTE FAILED
1906 006554  104001            5$:     ERROR   +1              ;CPU ERROR
1907 006556                    6$:
1908
1909                           ;
1912 006556                    MSPN:
1913
1914                           ;       TEST CLR, COM, INC   MODE 5
1915 006556  005004                    CLR     R4
1916 006560  005014                    CLR     (R4)
1917 006562  105114                    COMB    (R4)
1918 006564  005224                    INC     (R4)+           ;0=400
1919 006566  005054                    CLR     @-(R4)          ;*TEST INST. 400=0
1920 006570  001401                    BEQ     1$              ;BRANCH IF GOOD
1921                                                           ;MODE 5 FAILED
1922 006572  104001                    ERROR   +1              ;CPU ERROR
1923 006574  005204            1$:     INC     R4
1924 006576  005204                    INC     R4              ;RESET POINTER TO 0
1925 006600  005104                    COM     @-(R4)          ;*TEST INST. 376=1
1926 006602  001003                    BEQ     2$              ;BRANCH IF BAD
1927 006604  005204                    INC     R4
1928 006606  005204                    INC     R4              ;REPOSITION POINTER
1929 006610  005354                    DEC     @-(R4)          ;*TEST INST. 376=0
1930 006612  001403                    BEQ     2$              ;BRANCH IF BAD
1931 006614  005224                    INC     (R4)+           ;0=401 R4=2
1932 006616  105254                    INCB    @-(R4)          ;*TEST INST.,400= 0 376
1933 006620  001401                    BEQ     3$              ;BRANCH IF GOOD
1934                                                           ;MODE 5 FAILED
1935 006622  104001            2$:     ERROR   +1              ;CPU ERROR
1936 006624                    3$:
1937
1938                           ;
1941 006624                    MSPO:
1942
1943                           ;       TEST NEG MODE 5
```

BASE INSTRUCTION SET TESTS

```
1944 006624  005004                    CLR     R4
1945 006626  105104                    COMB    R4
1946 006630  005204                    INC     R4              ;R4=400
1947 006632  005001                    CLR     R1
1948 006634  105101                    COMB    R1
1949 006636  005301                    DEC     R1              ;R1=376
1950 006640  005002                    CLR     R2              ;R2=0
1951 006642  005012                    CLR     (R2)            ;0=0
1952 006644  005014                    CLR     (R4)            ;
1953 006646  005114                    COM     (R4)            ;400=-1
1954 006650  005011                    CLR     (R1)            ;376=0
1955 006652  005454                    NEG     @-(R4)          ;0=0
1956 006654  001401                    BEQ     2$              ;BRANCH IF GOOD
1957                                                           ;NEG FAILED
1958 006656  104001            1$:     ERROR   +1              ;CPU ERROR
1959 006660  005334            2$:     DEC     @(R4)+          ;0=-1
1960 006662  005454                    NEG     @-(R4)          ;0=1
1961 006664  001403                    BEQ     3$              ;BRANCH IF BAD
1962 006666  102402                    BVS     3$              ;BRANCH IF BAD
1963 006670  100401                    BMI     3$              ;BRANCH IF BAD
1964 006672  103401                    BCS     4$              ;BRANCH IF GOOD
1965                                                           ;NEG FAILED
1966 006674  104001            3$:     ERROR   +1              ;CPU ERROR
1967 006676  005334            4$:     DEC     @(R4)+          ; TEST RESULT OF NEGATE
1968 006700  001401                    BEQ     6$              ;BRANCH IF GOOD
1969                                                           ;RESULT OF NEGATE BAD
1970 006702  104001            5$:     ERROR   +1              ;CPU ERROR
1971 006704  105212            6$:     INCB    (R2)            ;0=1
1972 006706  005454                    NEG     @-(R4)          ;0=-1
1973 006710  001403                    BEQ     7$              ;
1974 006712  102402                    BVS     7$
1975 006714  103001                    BCC     7$              ;
1976 006716  100401                    BMI     8$              ;BRANCH IF GOOD
1977                                                           ;BAD NEGATE
1978 006720  104001            7$:     ERROR   +1              ;CPU ERROR
1979 006722  105212            8$:     INCB    (R2)            ;0=0
1980 006724  001401                    BEQ     10$             ;BRANCH IF GOOD
1981 006726  104001            9$:     ERROR   +1              ;CPU ERROR
1982 006730                    10$:
1983
1984                           ;
1986
1988 006730                    MSPP:
1989
1990                           ;       TEST CLR, COM, INC - MODE 6
1991 006730  005004                    CLR     R4
1992 006732  005204                    INC     R4
1993 006734  005204                    INC     R4              ;R4=2
1994 006736  005001                    CLR     R1
1995 006740  105101                    COMB    R1
1996 006742  005201                    INC     R1              ;R1=400
1997 006744  005011                    CLR     (R1)            ;
1998 006746  005121                    COM     (R1)+           ;400=-1
1999 006750  005011                    CLR     (R1)            ;R1=402
2000 006752  005211                    INC     (R1)            ;402=1
2001 006754  005002                    CLR     R2              ;R2=0
2002 006756  005012                    CLR     (R2)            ;0=0
```

BASE INSTRUCTION SET TESTS

```
2003 006760  005064  000376                CLR     376(R4)         ;400=0
2004 006764  001401                        BEQ     2$              ;BRANCH IF GOOD
2005 006766  104001                1$:     ERROR   +1              ;CPU ERROR
2006 006770  005364  000376        2$:     DEC     376(R4)         ;400= 1
2007 006774  005164  000400                COM     400(R4)         ;402=-1
2008 007000  001405                        BEQ     3$              ;BRANCH IF BAD
2009 007002  005264  000400                INC     400(R4)         ;402=-2
2010 007006  005264  000400                INC     400(R4)
2011 007012  001401                        BEQ     4$              ;BRANCH IF GOOD
2012                                                               ;MODE 6 FAILED
2013 007014  104001                3$:     ERROR   +1              ;CPU ERROR
2014 007016  005261  177776        4$:     INC     -2(R1)          ;400=0
2015 007022  001401                        BEQ     6$              ;BRANCH IF GOOD
2016                                                               ;INC MODE 6 FAILED
2017 007024  104001                5$:     ERROR   +1              ;CPU ERROR
2018 007026                        6$:
2019
2020                               ;
2022                               ;
2023                               ;
2025 007026                        MSPQ:
2026
2027                               ;       TEST NEG MODE 6
2028 007026  005001                        CLR     R1              ;R1=0
2029 007030  005004                        CLR     R4
2030 007032  105104                        COMB    R4
2031 007034  005204                        INC     R4              ;R4=400
2032 007036  005014                        CLR     (R4)            ;
2033 007040  005114                        COM     (R4)            ;400=-1
2034 007042  005044                        CLR     -(R4)           ;376=0
2035 007044  005044                        CLR     (R4)            ;
2036 007046  005224                        INC     (R4)+           ;374=1 R4=376
2037 007050  005464  000002                NEG     2(R4)           ;400=1
2038 007054  001403                        BEQ     1$              ;NEGATE FAILED
2039 007056  102402                        BVS     1$
2040 007060  100401                        BMI     1$
2041 007062  103401                        BCS     2$              ;BRANCH IF GOOD
2042                                                               ;NEGATE FAILED
2043 007064  104001                1$:     ERROR   +1              ;CPU ERROR
2044 007066  005364  000002        2$:     DEC     2(R4)           ; TEST RESULT OF NEGATE
2045 007072  001401                        BEQ     4$              ;BRANCH IF GOOD
2046                                                               ;RESULT OF NEGATE FAILED
2047 007074  104001                3$:     ERROR   +1              ;CPU ERROR
2048 007076  005464  000000        4$:     NEG     0(R4)           ;*0=0
2049 007102  001401                        BEQ     5$              ; BRANCH IF GOOD
2050                                                               ;NEGATE FAILED
2051 007104  104001                        ERROR   +1              ;CPU ERROR
2052 007106  105461  000374        5$:     NEGB    374(R1)         ;374=0 377
2053 007112  102403                        BVS     6$              ;
2054 007114  001402                        BEQ     6$
2055 007116  100001                        BPL     6$
2056 007120  103401                        BCS     7$              ;BRANCH IF GOOD
2057                                                               ;NEGATE FAILED
2058 007122  104001                6$:     ERROR   +1              ;CPU ERROR
2059 007124  105261  000374        7$:     INCB    374(R1)         ;374=0
2060 007130  001401                        BEQ     9$              ;BRANCH IF GOOD
2061                                                               ;NEGATE FAILED
```

BASE INSTRUCTION SET TESTS

```
2062 007132  104001                  8$:     ERROR   +1              ;CPU ERROR
2063 007134                          9$:
2064
2065                                 ;
2068 007134                          MSPR:
2069
2070                                 ;       TEST CLR, COM, INC   MODE 7
2071 007134  005001                          CLR     R1              ;R1=0
2072 007136  005004                          CLR     R4              ;
2073 007140  105104                          COMB    R4              ;
2074 007142  005204                          INC     R4              ;R4=400
2075 007144  005011                          CLR     (R1)
2076 007146  105111                          COMB    (R1)
2077 007150  005211                          INC     (R1)
2078 007152  005211                          INC     (R1)
2079 007154  005211                          INC     (R1)            ;;0=402
2080 007156  005014                          CLR     (R4)            ;400=0
2081 007160  005064  000002                  CLR     2(R4)           ;
2082 007164  005164  000002                  COM     2(R4)           ;402=-1
2083 007170  005074  177400                  CLR     @ 400(R4)       ;402=0
2084 007174  001401                          BEQ     2$              ;BRANCH IF GOOD
2085 007176  104001                  1$:     ERROR   +1              ;CPU ERROR
2086                                                                 ;INSTRUCTION FAILED
2087 007200  005171  000000          2$:     COM     @0(R1)          ;402=-1
2088 007204  100401                          BMI     4$              ;BRANCH IF GOOD
2089 007206  104001                  3$:     ERROR   +1              ;CPU ERROR
2090
2091 007210  005104                  4$:     COM     R4
2092 007212  005274  000401                  INC     @401(R4)                ;402=0
2093 007216  001401                          BEQ     6$              ;BRANCH IF GOOD
2094 007220  104001                  5$:     ERROR   +1              ;CPU ERROR
2095                                                                 ;MODE 7 FAILED
2096 007222                          6$:
2097
2098                                 ;
2101 007222                          MSPS:
2102
2103                                 ;       TEST NEG MODE 7
2104 007222  005004                          CLR     R4
2105 007224  005014                          CLR     (R4)            ;0=0
2106 007226  005002                          CLR     R2              ;
2107 007230  105102                          COMB    R2
2108 007232  005202                          INC     R2              ;R2=400
2109 007234  005012                          CLR     (R2)            ;400=0
2110 007236  005472  177400                  NEG     @-400(R2)               ;NEG OF 0=0
2111 007242  103401                          BCS     1$              ;****
2112 007244  001401                          BEQ     2$              ;BRANCH IF GOOD
2113 007246  104001                  1$:     ERROR   +1              ;CPU ERROR
2114                                                                 ;
2115 007250  005314                  2$:     DEC     (R4)            ;0=-1
2116 007252  005474  000400                  NEG     @+400(R4)               ;0=1
2117 007256  001403                          BEQ     3$              ;BRANCH IF ERROR
2118 007260  102402                          BVS     3$              ;
2119 007262  100401                          BMI     3$              ;
2120 007264  103401                          BCS     4$              ;BRANCH IF GOOD
2121 007266  104001                  3$:     ERROR   +1              ;CPU ERROR
2122                                                                 ;NEGATE MODE 7 FAILED
```

BASE INSTRUCTION SET TESTS

```
2123 007270                        4$:
2124
2125                               ;
2128 007270                        MSPT.
2129
2130                               ;       TEST SINGLE OPERAND MODE 2 REG 7
2131 007270  005004                        CLR     R4
2132 007272  105104                        COMB    R4
2133 007274  005204                        INC     R4              ;R4=400
2134 007276  005027                        CLR     (R7)+           ;CLEAR NEXT LOCATION
2135 007300  177777                1$:     .WORD   -1              ;SETUP INITIAL DATA
2136 007302  001401                        BEQ     3$              ;BRANCH IF GOOD
2137 007304  104001                2$:     ERROR   +1              ;CPU ERROR
2138 007306                        3$:
2139
2140                               ;
2142
2144 007306                        MSPU:
2145
2146                               ;       TEST TST MODE 0
2147 007306  005004                        CLR     R4              ;R4=0
2148 007310  000277                        SCC                     ;CONDITION CODES =1111
2149 007312  000244                        CLZ                     ;CC=1011
2150 007314  005704                        TST     R4              ;*TEST INSTRUCTION
2151 007316  103403                        BCS     1$
2152 007320  102402                        BVS     1$
2153 007322  100401                        BMI     1$              ;BRANCH IF ERROR
2154 007324  001401                        BEQ     2$              ;BRANCH IF GOOD
2155 007326  104001                1$:     ERROR   +1              ;CPU ERROR
2156                                                               ;TST MODE 0 FAILED
2157 007330  005304                2$:     DEC     R4              ;R4=-1
2158 007332  000277                        SCC
2159 007334  000250                        CLN                     ;CC=0111
2160 007336  005704                        TST     R4              ;*TEST INSTRUCTION MODE 0
2161 007340  103403                        BCS     3$              ;BRANCH IF ERROR
2162 007342  102402                        BVS     3$
2163 007344  001401                        BEQ     3$
2164 007346  100401                        BMI     4$              ;BRANCH IF GOOD
2165 007350  104001                3$:     ERROR   +1              ;CPU ERROR
2166                                                               ;TST FAILED
2167 007352                        4$:
2168
2169                               ;
2172 007352                        MSPV0:
2173
2174                               ;       TEST TST MODE 0 BYTE
2175 007352  005004                        CLR     R4
2176 007354  105104                        COMB    R4              ;0=000 377
2177 007356  000277                        SCC
2178 007360  000250                        CLN                     ;CC=0111
2179 007362  105704                        TSTB    R4              ;*TEST INSTRUCTION ON EVEN BYTE
2180 007364  102403                        BVS     1$              ;BRANCH IF ERROR
2181 007366  103402                        BCS     1$              ;
2182 007370  102401                        BVS     1$              ;
2183 007372  100401                        BMI     2$              ;BRANCH IF GOOD
2184 007374  104001                1$:     ERROR   +1              ;CPU ERROR
2185                                                               ;
```

BASE INSTRUCTION SET TESTS

```
2186 007376  005204                2$:     INC     R4              ;POINT TO 1
2187 007400  105704                        TSTB    R4              ; TEST INSTRUCTION
2188 007402  001401                        BEQ     4$              ;BRANCH IF GOOD
2189 007404  104001                3$:     ERROR   +1              ;CPU ERROR
2190                                                               ;TST FAILED ON BYTE
2191 007406                        4$:
2192
2193                               ;
2196 007406                        MSPV:
2197
2198                               ;       TEST TST MODE 1
2199 007406  005004                        CLR     R4
2200 007410  005014                        CLR     (R4)            ;0=0
2201 007412  000277                        SCC
2202 007414  000244                        CLZ                     ;CC=1011
2203 007416  005714                        TST     (R4)            ;*TEST INSTRUCTION IN MODE 1
2204 007420  103403                        BCS     1$              ;BRANCH IF ERROR
2205 007422  102402                        BVS     1$
2206 007424  100401                        BMI     1$              ;
2207 007426  001401                        BEQ     2$              ;BRANCH IF GOOD
2208 007430  104001                1$:     ERROR   +1              ;CPU ERROR
2209
2210 007432  005214                2$:     INC     (R4)            ;0=1
2211 007434  000277                        SCC
2212 007436  005714                        TST     (R4)            ; TEST INSTRUCTION
2213 007440  001403                        BEQ     3$              ;BRANCH IF ERROR
2214 007442  102402                        BVS     3$
2215 007444  103401                        BCS     3$
2216 007446  100001                        BPL     4$              ;BRANCH IF GOOD
2217 007450  104001                3$:     ERROR   +1              ;CPU ERROR
2218                                                               ;TST FAILED MODE 1
2219 007452                        4$:
2220
2221                               ;
2224 007452                        MSPX:
2225
2226                               ;       TEST TST MODE 1 BYTE
2227 007452  005004                        CLR     R4              ;R4=0
2228 007454  005014                        CLR     (R4)
2229 007456  105114                        COMB    (R4)
2230 007460  005214                        INC     (R4)            ;0=001 000
2231 007462  000277                        SCC
2232 007464  000244                        CLZ                     ;CC=1011
2233 007466  105714                        TSTB    (R4)            ;*TEST INTRUCTION
2234 007470  103403                        BCS     1$              ;BRANCH IF ERROR
2235 007472  102402                        BVS     1$              ;
2236 007474  100401                        BMI     1$
2237 007476  001401                        BEQ     2$              ;BRANCH IF GOOD
2238 007500  104001                1$:     ERROR   +1              ;CPU ERROR
2239
2240 007502  005204                2$:     INC     R4              ;R4=1
2241 007504  000277                        SCC
2242 007506  105714                        TSTB    (R4)            ; TEST INSTRUCTION
2243 007510  001403                        BEQ     3$              ;BRANCH IF ERROR
2244 007512  100402                        BMI     3$
2245 007514  102401                        BVS     3$
2246 007516  103001                        BCC     4$              ;BRANCH IF GOOD
```

BASE INSTRUCTION SET TESTS

```
2247 007520  104001                3$:     ERROR   +1              ;CPU ERROR
2248                                                               ;
2249 007522                        4$:
2250
2251                               ;
2254 007522                        MSPY:
2255
2256                               ;       TEST TST MODE 2
2257 007522  005004                        CLR     R4              ;
2258 007524  005024                        CLR     (R4)+           ;0=0
2259 007526  005014                        CLR     (R4)
2260 007530  005114                        COM     (R4)            ;2= 1
2261 007532  005004                        CLR     R4              ;R4=0
2262 007534  000277                        SCC             ;
2263 007536  000244                        CLZ             ;CC=1011
2264 007540  005724                        TST     (R4)+           ; TEST INSTRUCTION
2265 007542  103403                        BCS     1$              ;BRANCH IF ERROR
2266 007544  102402                        BVS     1$
2267 007546  100401                        BMI     1$
2268 007550  001401                        BEQ     2$              ;BRANCH IF GOOD
2269 007552  104001                1$:     ERROR   +1              ;CPU ERROR
2270                                                               ;MODE 2 TEST FAILED
2271 007554  005724                2$:     TST     (R4)+           ;TST LOC2
2272 007556  103403                        BCS     3$
2273 007560  102402                        BVS     3$
2274 007562  001401                        BEQ     3$
2275 007564  100401                        BMI     4$
2276 007566  104001                3$:     ERROR   +1              ;CPU ERROR
2277                                                               ;MODE 2 FAILED
2278 007570                        4$:
2279
2280                               ;
2283 007570                        MSPZ:
2284
2285                               ;       TEST TST MODE 2 BYTE
2286 007570  005004                        CLR     R4
2287 007572  005024                        CLR     (R4)+           ;
2288 007574  105144                        COMB    -(R4)           ;0=377 000
2289 007576  005304                        DEC     R4              ;R4=0
2290 007600  000277                        SCC
2291 007602  000244                        CLZ                     ;CC=1011
2292 007604  105724                        TSTB    (R4)+           ;
2293 007606  102403                        BVS     1$              ;BRANCH IF ERROR
2294 007610  103402                        BCS     1$
2295 007612  100401                        BMI     1$
2296 007614  001401                        BEQ     2$              ;BRANCH IF GOOD
2297 007616  104001                1$:     ERROR   +1              ;CPU ERROR
2298                                                               ;MODE 2 EVEN BYTE FAILED
2299 007620  000277                2$:     SCC
2300 007622  000250                        CLN                     ;CC=0111
2301 007624  105724                        TSTB    (R4)+           ;
2302 007626  001403                        BEQ     3$              ;
2303 007630  103402                        BCS     3$
2304 007632  102401                        BVS     3$
2305 007634  100401                        BMI     4$              ;BRANCH IF GOOD
2306 007636  104001                3$:     ERROR   +1              ;CPU ERROR
2307                                                               ;MODE 2 ODD BYTE FAILED
```

BASE INSTRUCTION SET TESTS

```
2308 007640                          4$:
2309
2310                                 ;
2313 007640                          MSPAA:
2314
2315                                 ;       TEST TST MODE 3
2316 007640  005004                          CLR     R4
2317 007642  005014                          CLR     (R4)
2318 007644  105114                          COMB    (R4)
2319 007646  005214                          INC     (R4)            ;0=400
2320 007650  005034                          CLR     @(R4)+          ;400=0
2321 007652  005004                          CLR     R4              ;R4=0
2322 007654  000277                          SCC
2323 007656  000244                          CLZ                     ;CC=1011
2324 007660  005734                          TST     @(R4)+          ; TEST MODE 3
2325 007662  103403                          BCS     1$              ;BRANCH IF ERROR
2326 007664  102402                          BVS     1$              ;
2327 007666  100401                          BMI     1$
2328 007670  001401                          BEQ     2$              ;BRANCH IF GOOD
2329 007672  104001                  1$:     ERROR   +1              ;CPU ERROR
2330                                                                 ;MODE 3 FAILED
2331 007674  005304                  2$:     DEC     R4
2332 007676  005304                          DEC     R4              ;R4=0
2333 007700  005334                          DEC     @(R4)+          ;400=-1
2334 007702  005004                          CLR     R4
2335 007704  000277                          SCC
2336 007706  000250                          CLN                     ;CC=0111
2337 007710  005734                          TST     @(R4)+          ; TEST INSTRUCTION
2338 007712  103403                          BCS     3$              ;
2339 007714  001402                          BEQ     3$              ;BRANCH IF ERROR
2340 007716  102401                          BVS     3$              ;
2341 007720  100401                          BMI     4$              ;BRANCH IF GOOD
2342                                                                 ;ERROR MODE 3
2343 007722  104001                  3$:     ERROR   +1              ;CPU ERROR
2344 007724                          4$:
2345
2346                                 ;
2349 007724                          MSPBB:
2350
2351                                 ;       TEST TST MODE 3 AUTO-INC
2352 007724  005004                          CLR     R4
2353 007726  005014                          CLR     (R4)            ;0=0
2354 007730  105114                          COMB    (R4)            ;
2355 007732  005214                          INC     (R4)            ;0=400
2356 007734  005001                          CLR     R1
2357 007736  105101                          COMB    R1
2358 007740  005201                          INC     R1              ;R1=400
2359 007742  005011                          CLR     (R1)            ;400=0
2360 007744  000277                          SCC
2361 007746  005734                          TST     @(R4)+          ;400=0
2362 007750  103403                          BCS     1$              ;ERROR IF CARRY
2363 007752  102402                          BVS     1$              ;ERROR IF OVERFLOW
2364 007754  100401                          BMI     1$              ;ERROR IF MINUS
2365 007756  001401                          BEQ     2$              ;ERROR IF NOT EQUAL
2366 007760  104001                  1$:     ERROR   +1              ;CPU ERROR
2367                                                         ;CC SHOULD = 0100
2368 007762  005304                  2$:     DEC     R4
```

BASE INSTRUCTION SET TESTS

```
2369 007764  005304                    DEC     R4
2370 007766  005704                    TST     R4              ;SEE IF AUTO-INC WORKED
2371 007770  001401                    BEQ     4$              ;ERROR IF R4 NE 0
2372 007772  104001            3$:     ERROR   +1              ;CPU ERROR
2373                                                   ;AUTO INC FIALED
2374 007774                    4$:
2375
2376                           ;
2379 007774                    MSTB3:
2380
2381                           ;       TEST TST MODE 3 BYTE
2382 007774  005004                    CLR     R4
2383 007776  005014                    CLR     (R4)
2384 010000  105114                    COMB    (R4)
2385 010002  005214                    INC     (R4)
2386 010004  005214                    INC     (R4)            ;0=401
2387 010006  005001                    CLR     R1
2388 010010  105101                    COMB    R1
2389 010012  005201                    INC     R1              ;R1=400
2390 010014  005011                    CLR     (R1)
2391 010016  005111                    COM     (R1)
2392 010020  105011                    CLRB    (R1)            ;400=377 000
2393 010022  105734                    TSTB    @(R4)+          ;** TEST INSTRUCTION
2394 010024  001403                    BEQ     1$              ;ERROR IF EQUAL
2395 010026  103402                    BCS     1$              ;ERROR IF CARRY SET
2396 010030  102401                    BVS     1$              ;ERROR IR OVERFLOW
2397 010032  100401                    BMI     2$              ;BRANCH IF MINUS
2398 010034  104001            1$:     ERROR   +1              ;CPU ERROR
2399                                                   ;CC ERROR
2400 010036  005304            2$:     DEC     R4
2401 010040  005304                    DEC     R4
2402 010042  001401                    BEQ     4$                      ;BRANCH IF AUTO INC WORKED
2403 010044  104001            3$:     ERROR   +1              ;CPU ERROR
2404                                                   ;AUTO-INC FAILED
2405 010046                    4$:
2406
2407                           ;
2410 010046                    MST4:
2411
2412                           ;       TEST TST MODE 4
2413 010046  005004                    CLR     R4
2414 010050  005014                    CLR     (R4)            ;0=0
2415 010052  005204                    INC     R4
2416 010054  005204                    INC     R4              ;R4=2
2417 010056  000277                    SCC
2418 010060  000244                    CLZ                     ;CC=1011
2419 010062  005744                    TST     -(R4)           ;**TEST INTRUCTION
2420 010064  103403                    BCS     1$              ;ERROR IF CARRY
2421 010066  102402                    BVS     1$              ;ERROR IF OVERFLOW
2422 010070  100401                    BMI     1$              ;ERROR IF MINUS
2423 010072  001401                    BEQ     2$              ;BRANCH IF GOOD
2424 010074  104001            1$:     ERROR   +1              ;CPU ERROR
2425                                                   ;CC WRONG
2426 010076  005704            2$:     TST     R4              ;INSURE CORRECT AUTO-DEC
2427 010100  001401                    BEQ     4$              ;BRANCH IF GOOD AUTO-DEC
2428                                                           ;BAD AUTO-DEC
2429 010102  104001            3$:     ERROR   +1              ;CPU ERROR
```

BASE INSTRUCTION SET TESTS

```
2430 010104                          4$:
2431
2432                                 ;
2435 010104                          MST4B:
2436
2437                                 ;       TEST TST MODE 4 BYTE
2438 010104  005004                          CLR     R4
2439 010106  005014                          CLR     (R4)
2440 010110  005114                          COM     (R4)
2441 010112  105114                          COMB    (R4)            ;0=377 000
2442 010114  000277                          SCC
2443 010116  005204                          INC     R4
2444 010120  005204                          INC     R4              ;R4=2
2445 010122  105744                          TSTB    -(R4)           ;**TEST INSTRUCTION
2446 010124  001403                          BEQ     1$              ;ERROR IF EQUAL TO 0
2447 010126  103402                          BCS     1$              ;ERROR IF CARRY
2448 010130  102401                          BVS     1$              ;ERROR IF OVERFLOW
2449 010132  100401                          BMI     2$              ;BRANCH IF MINUS
2450 010134  104001                  1$:     ERROR   +1              ;CPU ERROR
2451                                                 ;CC SHOULD EQUAL 0100
2452 010136  105744                  2$:     TSTB    -(R4)           ;**TEST EVEN BYTE
2453 010140  001401                          BEQ     4$              ;BRANCH IF GOOD
2454 010142  104001                  3$:     ERROR   +1              ;CPU ERROR
2455                                                 ;CC SHOULD EQUAL 0100 AND R4=-1
2456 010144                          4$:
2457                                 ;
2460 010144                          MST5:
2461
2462                                 ;       TEST TST MODE 5
2463 010144  005004                          CLR     R4              ;
2464 010146  005024                          CLR     (R4)+           ;0=0, R4=2
2465 010150  000277                          SCC
2466 010152  000244                          CLZ                     ;CC=1011
2467 010154  005754                          TST     @-(R4)          ; TEST INSTRUCTION
2468 010156  103403                          BCS     1$               ERROR IF CARRY
2469 010160  102402                          BVS     1$              ;ERROR IF OVERFLOW
2470 010162  100401                          BMI     1$              ;ERROR IF MINUS
2471 010164  001401                          BEQ     2$              ;BRANCH IF GOOD
2472 010166  104001                  1$:     ERROR   +1              ;CPU ERROR
2473                                                 ;CC WRONG, SHOULD = 0100
2474 010170  005704                  2$:     TST     R4
2475 010172  001401                          BEQ     4$              ;BRANCH IF AUTO-DEC WORKED
2476 010174  104001                  3$:     ERROR   +1              ;CPU ERROR
2477                                                 ;AUTO-DEC FAILED
2478 010176                          4$:
2479
2480                                 ;
2483 010176                          MST5B:
2484
2485                                 ;       TEST TST MODE 5 BYTE
2486 010176  005004                          CLR     R4
2487 010200  005014                          CLR     (R4)
2488 010202  105114                          COMB    (R4)
2489 010204  005214                          INC     (R4)            ;0=400
2490 010206  005034                          CLR     @(R4)+          ;400=0, R4=2
2491 010210  005154                          COM     @-(R4)          ;
2492 010212  105134                          COMB    @(R4)+          ;400=377 000 R4=2
```

BASE INSTRUCTION SET TESTS

```
2493 010214  105754                        TSTB    @-(R4)          ;**TEST INSTRUCTION
2494 010216  103403                        BCS     1$              ;ERROR IF CARRY
2495 010220  100402                        BMI     1$              ;ERROR IF MINUS
2496 010222  102401                        BVS     1$              ;ERROR IF OVERFLOW
2497 010224  001401                        BEQ     2$              ;BRANCH IF GOOD
2498 010226  104001                 1$:    ERROR   +1              ;CPU ERROR
2499                                                    ;CC SHOULD = 0100
2500 010230  005224                 2$:    INC     (R4)+           ;0=401
2501 010232  105754                        TSTB    @ (R4)          ;**TEST INSTRUCTION
2502 010234  100401                        BMI     4$              ;BRANCH IF GOOD
2503                                                               ;EVEN BYTE FAILURE
2504 010236  104001                 3$:    ERROR   +1              ;CPU ERROR
2505 010240                         4$:
2506
2507                                ;
2510 010240                         MST6:
2511
2512                                ;       TEST TST MODE 6
2513 010240  005004                        CLR     R4
2514 010242  005014                        CLR     (R4)            ;0=0
2515 010244  105104                        COMB    R4
2516 010246  005204                        INC     R4              ;R4=400
2517 010250  005014                        CLR     (R4)
2518 010252  005114                        COM     (R4)            ;400=-1
2519 010254  005764  177400                TST     -400(R4)        ;**TEST LOCATION 0
2520 010260  103403                        BCS     1$              ;ERROR IF CARRY
2521 010262  102402                        BVS     1$              ;ERROR IF OVERFLOW
2522 010264  100401                        BMI     1$              ;ERROR IF MINUS
2523 010266  001401                        BEQ     2$              ;BRANCH IF ZERO
2524 010270  104001                 1$:    ERROR   +1              ;CPU ERROR
2525                                                    ;CC ARE WRONG
2526 010272  005004                 2$:    CLR     R4              ;
2527 010274  005764  000400                TST     400(R4)         ;TST LOCATION 400
2528 010300  001401                        BEQ     3$              ;ERROR IF EQUAL
2529 010302  100401                        BMI     4$              ;BRANCH IF MINUS
2530 010304  104001                 3$:    ERROR   +1              ;CPU ERROR
2531                                                    ;CC ERROR
2532 010306                         4$:
2533
2534                                ;
2537 010306                         MST7:
2538
2539                                ;       TEST TST MODE 7
2540 010306  005004                        CLR     R4
2541 010310  005014                        CLR     (R4)
2542 010312  005124                        COM     (R4)+           ;0= 1
2543 010314  005014                        CLR     (R4)            ;2=0
2544 010316  005002                        CLR     R2              ;R2=0
2545 010320  005004                        CLR     R4
2546 010322  105104                        COMB    R4
2547 010324  005204                        INC     R4              ;R4=400
2548 010326  005014                        CLR     (R4)            ;400=0
2549 010330  005774  177402                TST     @-376(R4)       ;**TEST LOCATION 0
2550 010334  103403                        BCS     1$              ;ERROR IF CARRY
2551 010336  102402                        BVS     1$              ;ERROR IF OVERFLOW
2552 010340  001401                        BEQ     1$              ;ERROR IF ZERO
2553 010342  100401                        BMI     2$              ;BRANCH IF GOOD
```

BASE INSTRUCTION SET TESTS

```
2554 010344  104001                  1$:     ERROR   +1              ;CPU ERROR
2555                                                         ;ERROR IN CC
2556 010346  005222                  2$:     INC     (R2)+           ;0=-2 R2=2
2557 010350  005774  177402                  TST     @ 376(R4)       ;** CHECK CONTENTS OF LOCATION 2
2558 010354  100401                          BMI     3$              ;ERROR IF MINUS
2559 010356  001401                          BEQ     4$              ;BRANCH IF GOOD
2560 010360  104001                  3$:     ERROR   +1              ;CPU ERROR
2561                                                         ;CC SHOULD = 0100
2562 010362                          4$:
2563
2564                                 ;
2566                                 ;
2567                                 ;
2568                                 ;
2569                                 ;
2570                                 .SBTTL ************ DOUBLE OPERAND TESTS *****************
2571                                 ;
2573 010362                          MDMO:
2574
2575                                 ;       TEST MOVE MODE 0
2576 010362  005004                          CLR     R4              ;R4=0
2577 010364  005001                          CLR     R1
2578 010366  005101                          COM     R1              ;R1=-1
2579 010370  010104                          MOV     R1,R4           ;**TEST MOVE INSTRUCTION
2580 010372  001402                          BEQ     1$              ;ERROR IF R4=0
2581 010374  102401                          BVS     1$              ;ERROR IF OVERFLOW
2582 010376  100401                          BMI     2$              ;BRANCH IF MINUS
2583 010400  104001                  1$:     ERROR   +1              ;CPU ERROR
2584                                                         ;CC SHOULD = 100-, R4 SHOULD= 1
2585 010402  005204                  2$:     INC     R4              ;R4 SHOULD =0
2586 010404  001375                          BNE     1$              ;ERROR IF R4 NE 0
2587
2588                                 ;
2591 010406                          MDAO:
2592
2593                                 ;       TEST ADD MODE 0
2594 010406  005004                          CLR     R4              ;R4=0
2595 010410  005001                          CLR     R1
2596 010412  005101                          COM     R1              ;R1=-1
2597 010414  060104                          ADD     R1,R4           ;** TEST ADD OF R1 TO R4
2598 010416  001403                          BEQ     1$              ;ERROR IF ZERO
2599 010420  103402                          BCS     1$              ;ERROR IF CARRY
2600 010422  102401                          BVS     1$              ;ERROR IF OVERFLOW
2601 010424  100401                          BMI     2$              ;BRANCH IF MINUS
2602 010426  104001                  1$:     ERROR   +1              ;CPU ERROR
2603                                                         ;CC SHOULD = 1000 , R4=-1
2604 010430  005204                  2$:     INC     R4              ;R4 SHOULD =0
2605 010432  001401                          BEQ     4$              ;BRANCH IF R4=0
2606 010434  104001                  3$:     ERROR   +1              ;CPU ERROR
2607                                                         ;R4 SHOULD = 0
2608 010436                          4$:
2609
2610                                 ;
2613 010436                          MDSO:
2614
2615                                 ;       TEST SUB MODE 0
2616 010436  005004                          CLR     R4
```

```
********** DOUBLE OPERAND TESTS *****************

2617 010440  005001                          CLR     R1
2618 010442  005201                          INC     R1                      ;R1=1 R4=0
2619 010444  160104                          SUB     R1,R4                   ;**TEST OF R4 R1, R4=-1
2620 010446  102403                          BVS     1$                      ;ERROR IF V SET
2621 010450  103002                          BCC     1$                      ;ERROR IF NO CARRY
2622 010452  001401                          BEQ     1$                      ;ERROR IF =0
2623 010454  100401                          BMI     2$                      ;BRANCH IF MINUS
2624 010456  104001                  1$:     ERROR   +1              ;CPU ERROR
2625                                                                 ;CC SHOULD = 1001
2626 010460  005101                  2$:     COM     R1                      ;R1=1
2627 010462  005201                          INC     R1                      ;GET TWO'S COMPLIMENT, R1= 1
2628 010464  160104                          SUB     R1,R4                   ;**TEST R4-R1 (1- 1= 0
2629 010466  001401                          BEQ     4$                      ;BRANCH IF ZERO
2630 010470  104001                  3$:     ERROR   +1              ;CPU ERROR
2631                                                                 ;CC SHOULD = 0100
2632 010472                          4$:
2633
2634
2637 010472                          MOM27:
2638
2639                                 ;       TEST MOV MODE 27,00
2640 010472  000257                          CCC                             ;CC=0000
2641 010474  012704  125252                  MOV     #125252,R4              ;**TEST MOVE
2642 010500  001401                          BEQ     1$                      ;ERROR IF = 0
2643 010502  100401                          BMI     2$                      ;BRANCH IF MINUS
2644 010504  104001                  1$:     ERROR   +1              ;CPU ERROR
2645                                                                 ;CC SHOULD = 1000
2646 010506  012701  052525          2$:     MOV     #052525,R1              ;**TEST MOVE
2647 010512  100401                          BMI     3$                      ;ERROR IF MINUS
2648 010514  001001                          BNE     4$                      ;BRANCH IF NE 0
2649 010516  104001                  3$:     ERROR   +1              ;CPU ERROR
2650                                                                 ;CC SHOULD = 0000
2651 010520  060104                  4$:     ADD     R1,R4                   ;R1+R4=-1
2652 01[illegible]  100401                   BMI     6$                      ;BRANCH IF MINUS
2653 01[illegible]  104001           5$:     ERROR   +1              ;CPU ERROR
2654                                                                 ;MOV FAILED
2655 010526  005204                  6$:     INC     R4                      ;R4+1=0
2656 010530  001375                          BNE     5$                      ;ERROR IF NOT ZERO
2657
2658
2661 010532                          MBI00:
2662
2663                                 ;       TEST BIC, BIS MODE 0,0
2664 010532  005004                          CLR     R4
2665 010534  005104                          COM     R4                      ;R4=-1
2666 010536  012701  125252                  MOV     #125252,R1              ;SETUP R1 TEST DATA
2667 010542  012702  052525                  MOV     #052525,R2              ;R2=COMPLIMENT OF R1
2668 010546  000261                          SEC
2669 010550  040104                          BIC     R1,R4                   ;**TEST BIC WITH CARRY SET
2670 010552  103003                          BCC     1$                      ;ERROR IF NO CARRY
2671 010554  102402                          BVS     1$                      ;ERROR IF OVERFLOW
2672 010556  001401                          BEQ     1$                      ;ERROR IF 0
2673 010560  100001                          BPL     2$                      ;BRANCH IF PLUS
2674 010562  104001                  1$:     ERROR   +1              ;CPU ERROR
2675                                                                 ;CC SHOULD = 0001
2676 010564  020402                  2$:     CMP     R4,R2                   ;COMPARE CONTENTS OF R4 AND R2
2677 010566  001401                          BEQ     4$                      ;BRANCH IF EQUAL
```

F5


```
********** DOUBLE OPERAND TESTS *****************

 2678 010570  104001                  3$:     ERROR   +1              ;CPU ERROR
 2679                                                                 ;R4 AND R2 SHOULD BE EQUAL
 2680 010572  005301                  4$:     DEC     R1                      ;R1=125251
 2681 010574  050201                          BIS     R2,R1                   ;BIS 052525 AND 125251=177775
 2682 010576  100401                          BMI     6$                      ;BRANCH IF MIUS VALUE
 2683 010600  104001                  5$:     ERROR   +1              ;CPU ERROR
 2684                                                                 ;BAD BIS OPERATION
 2685 010602  005201                  6$:     INC     R1
 2686 010604  005201                          INC     R1
 2687 010606  005201                          INC     R1                      ;R1=0
 2688 010610  001373                          BNE     5$                      ;ERROR IF NE 0
 2689
 2690                                  ;
 2693 010612                           MBC00:
 2694
 2695                                  ;       TEST BIT, CMP MODE 0,0
 2696 010612  012701  125252                    MOV     #125252,R1              ;R1=125252
 2697 010616  012704  100000                    MOV     #100000,R4              ;R4=100000
 2698 010622  012702  052525                    MOV     #052525,R2              ;R2=052525
 2699 010626  030401                            BIT     R4,R1                   ;**TEST OF BIT ,CC=1000
 2700 010630  001401                            BEQ     1$                      ;ERROR IF EQ 0
 2701 010632  100401                            BMI     2$                      ;BRANCH IF GOOD
 2702 010634  104001                  1$:     ERROR   +1              ;CPU ERROR
 2703                                                                 ;CC SHOULD = 1000
 2704 010636  020401                  2$:     CMP     R4,R1                   ;*TEST 100000-125252=25252
 2705 010640  001402                          BEQ     3$                      ;ERROR IF EQUAL 0
 2706 010642  103001                          BCC     3$                      ;ERROR IF CARRY CLEARED
 2707 010644  100401                          BMI     4$                      ;BRANCH IF GOOD
 2708 010646  104001                  3$:     ERROR   +1              ;CPU ERROR
 2709                                                                 ;CC SHOULD = 0010
 2710 010650  020104                  4$:     CMP     R1,R4                   ;125252 100000 = 25252
 2711 010652  001403                          BEQ     5$                      ;ERROR IF EQUAL
 2712 010654  103402                          BCS     5$                      ;ERROR IF CARRY
 2713 010656  102401                          BVS     5$                      ;ERROR IF OVERFLOW
 2714 010660  100001                          BPL     6$                      ;BRANCH IF GOOD
 2715 010662  104001                  5$:     ERROR   +1              ;CPU ERROR
 2716                                                                 ;CC SHOULD =0001
 2717 010664  005004                  6$:     CLR     R4
 2718 010666  005204                          INC     R4                      ;R4=1
 2719 010670  000277                          SCC
 2720 010672  030401                          BIT     R4,R1                   ;R4 ↑ R1 = 2
 2721 010674  001401                          BEQ     8$                      ;BRANCH IF GOO
 2722 010676  104001                  7$:     ERROR   +1              ;CPU ERROR
 2723                                                                 ;CC SHOULD = 0101
 2724 010700                           8$:
 2725
 2726                                  ;
 2729 010700                           MM11:
 2730
 2731                                  ;       TEST MOV, MOVB MODE 1,1 AND SIGN EXT ON MOVB TO GPR
 2732 010700  012704  000400                    MOV     #400,R4         ;R4=400
 2733 010704  012701  000402                    MOV     #402,R1         ;R1=402
 2734 010710  005014                            CLR     (R4)            ;
 2735 010712  005114                            COM     (R4)            ;400=-1
 2736 010714  005011                            CLR     (R1)                    ;
 2737 010716  105111                            COMB    (R1)            ;402=000 377
 2738 010720  005002                            CLR     R2              ;R2=0
```

```
********** DOUBLE OPERAND TESTS *****************

 2739 010722  012703  000405                MOV     #405,R3         ;R3=405
 2740 010726  000277                        SCC                     ;CC=1111
 2741 010730  011412                        MOV     (R4),(R2)       ;MOV    400 TO 0 ,0=-1
 2742 010732  001403                        BEQ     1$              ;ERROR IF 0
 2743 010734  102402                        BVS     1$              ;ERROR IF OVERFLOW
 2744 010736  103001                        BCC     1$              ;ERROR IF NO CARRY
 2745 010740  100401                        BMI     2$              ;BRANCH IF GOOD
 2746 010742  104001              1$:       ERROR   +1              ;CPU ERROR
 2747                                                  ;CC SHOULD =1001
 2748 010744  005212              2$:       INC     (R2)            ;0=0
 2749 010746  001004                        BNE     3$              ;ERROR IF NOT 0
 2750 010750  000257                        CCC                     ;CC=0000
 2751 010752  111113                        MOVB     R1),(R3)               ;405=377
 2752 010754  001401                        BEQ     3$              ;ERROR IF EQUAL
 2753 010756  100401                        BMI     4$              ;BRANCH IF GOOD
 2754 010760  104001              3$:       ERROR   +1              ;CPU ERROR
 2755                                                  ;
 2756 010762  105213              4$:       INCB    (R3)            ;405=0
 2757 010764  001375                        BNE     3$              ;ERROR IF 405 NOT 0
 2758                             ;CHECK THAT STGN EXTENSION OCCURS ON A MOVB TO GENERAL REGISTER.
 2759 010766  005002                        CLR     R2              ;INIT R2 TO ZERO.
 2760 010770  111102                        MOVB    (R1),R2         ;MOVE 377 TO R2
 2761 010772  100005                        BPL     5$              ;ERROR! BIT 15 SHOULD BE SET.
 2762 010774  102404                        BVS     5$              ;V BIT [illegible]LD BE CLEARED
 2763 010776  103403                        BCS     5$              ;CARR[illegible] SHOULD BE UNAFFECTED
 2764 011000  022702  177777                CMP     #177777,R2      ;TEST R2
 2765 011004  001401                        BEQ     6$              ;SIGN EXTENDED THROUGH UPPER BYTE
 2766                                                                ;ERROR! BYTE SHOULD HAVE
 2767                                                                ;SIGN EXTENDED THROUGH UPPER BYTE
 2768 011006  104001              5$:       ERROR   +1              ;CPU ERROR
 27[illegible] 011010              6$:
 2[illegible]                     ;
 2774 011010                      MA11:
 2775
 2776                             ;       TEST ADD MODE 1,1
 2777 011010  012704  000400                MOV     #400,R4         ;R4=400
 2778 011014  012701  000402                MOV     #402,R1         ;R1=402
 2779 011020  012714  177753                MOV     #-25,(R4)       ;400=-25
 2780 011024  012711  000024                MOV     #24,(R1)        ;402=24
 2781 011030  061114                        ADD     (R1),(R4)       ;-25+24=-1
 2782 011032  001404                        BEQ     1$              ;ERROR IF 0
 2783 011034  103403                        BCS     1$              ;ERROR IF CARRY
 2784 011036  100002                        BPL     1$              ;ERROR IF POSITIVE RESULT
 2785 011040  005214                        INC     (R4)            ;-1+1=0
 2786 011042  001401                        BEQ     2$              ;BRANCH IF GOOD
 2787 011044  104001              1$:       ERROR   +1              ;CPU ERROR
 2788                                                  ;CC SHOULD = 1000
 2789 011046                      2$:
 2790
 2791                             ;
 2794 011046                      MS11:
 2795
 2796                             ;       TEST SUB MODE 1,1
 2797 011046  012704  000400                MOV     #400,R4         ;R4=400
 2798 011052  012701  000404                MOV     #404,R1         ;R1=404
 2799 011056  012714  000003                MOV     #3,(R4)         ;400=3
```

```
********** DOUBLE OPERAND TESTS *****************

2800 011062  012711  000006                    MOV     #6,(R1)          ;406=6
2801 011066  000277                            SCC                      ;CC=1111
2802 011070  161411                            SUB     (R4),(R1)        ;6-3=3
2803 011072  001402                            BEQ     1$               ;ERROR IF 0
2804 011074  100401                            BMI     1$               ;ERROR IF MINUS
2805 011076  103001                            BCC     2$               ;BRANCH IF GOOD
2806 011100  104001                1$:         ERROR   +1               ;CPU ERROR
2807                                                           ;CC SHOULD = 0000
2808 011102  161411                2$:         SUB     (R4),(R1)        ;3-3=0
2809 011104  001375                            BNE     1$               ;ERROR IF NOT 0
2810
2811                               ;
2814 011106                        MBB11:
2815
2816                               ;           TEST BIC, BIS MODE 1,1
2817 011106  012704  000400                    MOV     #400,R4          ;R4=400
2818 011112  012701  000402                    MOV     #402,R1          ;R1=402
2819 011116  012714  052525                    MOV     #052525,(R4)     ;400=052525
2820 011122  012711  125252                    MOV     #125252,(R1)     ;402=125252
2821 011126  051411                            BIS     (R4),(R1)        ;R4 V R1 =  1
2822 011130  001401                            BEQ     1$               ;ERROR IF 0
2823 011132  100401                            BMI     2$               ;BRANCH IF GOOD
2824 011134  104001                1$:         ERROR   +1               ;CPU ERROR
2825                                                           ;CC SHOULD = 1000
2826 011136  005211                2$:         INC     (R1)             ;402=0
2827 011140  001401                            BEQ     4$               ;BRANCH IF GOOD
2828 011142  104001                3$:         ERROR   +1               ;CPU ERROR
2829                                                           ;CC SHOULD = 0100
2830 011144  005311                4$:         DEC     (R1)             ;402=-1
2831 011146  041411                            BIC     (R4),(R1)        ;R1=125252
2832 011150  001401                            BEQ     5$               ;ERROR IF 0
2833 011152  100401                            BMI     6$               ;BRANCH IF GOOD
2834 011154  104001                5$:         ERROR   +1               ;CPU ERROR
2835                                                           ;CC SHOULD = 1000
2836 011156  005111                6$:         COM     (R1)             ;402=052525
2837 011160  041114                            BIC     (R1),(R4)        ;400=0
2838 011162  001401                            BEQ     8$               ;BRANCH IF GOOD
2839 011164  104001                7$:         ERROR   +1               ;CPU ERROR
2840                                                           ;CC SHOULD = 0100
2841 011166                        8$:
2842
2843                               ;
2846 011166                        MBC11:
2847
2848                               ;           TEST BIT, CMP MODE 1,1
2849 011166  012704  000400                    MOV     #400,R4                  ;R4=400
2850 011172  012714  052525                    MOV     #052525,(R4)             ;400=052525
2851 011176  012701  000402                    MOV     #402,R1                  ;R1-402
2852 011202  012711  125252                    MOV     #125252,(R1)             ;402=125252
2853 011206  000241                            CLC                              ;CLEAR CARRY
2854 011210  031411                            BIT     (R4),(R1)                ;**052525↑125252=0
2855 011212  103401                            BCS     1$                       ;ERROR IF CARRY
2856 011214  001401                            BEQ     2$                       ;BRANCH IF GOOD
2857 011216  104001                1$:         ERROR   +1               ;CPU ERROR
2858                                                           ,CC SHOULD = 0100
2859 011220  021411                2$:         CMP     (R4),(R1)                ;400-402=125253
2860 011222  001403                            BEQ     3$                       ;ERROR IF ZERO
```

```
**********      ALE OPERAND TESTS *****************

 2861 011224  103002                        BCC     3$                      ;ERROR IF NO CARRY
 2862 011226  102001                        BVC     3$                      ;ERROR IF NO OVERFLOW
 2863 011230  100401                        BMI     4$                      ;BRANCH IF GOOD
 2864 011232  104001              3$:       ERROR   +1              ;CPU ERROR
 2865                                                               ;CC SHOULD = 1000
 2866 011234  005014              4$:       CLR     (R4)
 2867 011236  005214                        INC     (R4)                    ;400=1
 2868 011240  031114                        BIT     (R1),(R4)               ;125252+1=0
 2869 011242  001401                        BEQ     6$                      ;BRANCH IF GOOD
 2870 011244  104001              5$:       ERROR   +1              ;CPU ERROR
 2871                                                               ;CC SHOULD= 0100
 2872 011246                      6$:
 2873
 2874                             ;
 2877 011246                      MM22:
 2878
 2879                             ;       TEST MOV MODE 2,2
 2880 011246  012704  000400                MOV     #400,R4         ;R4=400
 2881 011252  012701  000402                MOV     #402,R1         ;R1=402
 2882 011256  012714  000005                MOV     #5,(R4)         ;400=5
 2883 011262  005021                        CLR     (R1)+           ;402=0
 2884 011264  005011                        CLR     (R1)            ;
 2885 011266  005111                        COM     (R1)            ;404=-1
 2886 011270  005741                        TST     -(R1)           ;R1=402
 2887 011272  000277                        SCC                     ;CC=1111
 2888 011274  012124                        MOV     (R1)+,(R4)+     ;400=0 R4=402 R1=404
 2889 011276  100403                        BMI     1$              ;ERROR IF MINUS
 2890 011300  103002                        BCC     1$              ;ERROR IF NO CARRY
 2891 011302  102401                        BVS     1$              ;ERROR IF OVERFLOW
 2892 011304  001401                        BEQ     2$              ;BRANCH IF GOOD
 2893 011306  104001              1$:       ERROR   +1              ;CPU ERROR
 2894                                                       ;CC SHOULD= 0101
 2895 011310  005244              2$:       INC     -(R4)           ;400=1 R4=400
 2896
 2897 011312  061411                        ADD     (R4),(R1)       ;1+ -1 =0
 2898 011314  001401                        BEQ     4$              ;BRANCH IF GOOD
 2899 011316  104001              3$:       ERROR   +1              ;CPU ERROR
 2900                                                       ;CC SHOULD = 0100
 2901 011320                      4$:
 2902
 2903                             ;
 2906 011320                      MS22:
 2907
 2908                             ;       TEST SUB MODE 2,2
 2909 011320  012704  000400                MOV     #400,R4                 ;R4=400
 2910 011324  012701  000402                MOV     #402,R1                 ;R1=402
 2911 011330  012714  177760                MOV     #177760,(R4)            ;400=177760
 2912 011334  012711  177750                MOV     #177750,(R1)            ;402=177750
 2913 011340  162421                        SUB     (R4)+,(R1)              ;R1=177770
 2914 011342  001403                        BEQ     1$                      ;ERROR IF ZERO
 2915 011344  102402                        BVS     1$                      ;ERROR IF OVERFLOW
 2916 011346  103001                        BCC     1$                      ;ERROR IF NO CARRY
 2917 011350  100401                        BMI     2$                      ;BRANCH IF GOOD
 2918 011352  104001              1$:       ERROR   +1              ;CPU ERROR
 2919                                                               ;CC SHOULD=1000
 2920 011354  005241              2$:       INC     -(R1)                   ;R1=1777771
 2921 011356  162721  177771              SUB     #177771,(R1)+           ;R1=0
```

```
********** DOUBLE OPERAND TESTS *****************

2922 011362  100401                          BMI     3$                      ;ERROR IF MINUS
2923 011364  001401                          BEQ     4$                      ;BRANCH IF GOOD
2924 011366  104001                  3$:     ERROR   -1              ;CPU ERROR
2925                                                                 ;CCSHOULD = 0100
2926 011370                          4$:
2927
2928                                 ;
2944 011370                          MBB22:
2945
2946                                 ;       TEST BIC, BICB, BIS, BISB MODE 2,2
2947 011370  012704  000400                  MOV     #400,R4         ;R4=400
2948 011374  012701  000402                  MOV     #402,R1         ;R1=402
2949 011400  012702  000404                  MOV     #404,R2         ;R2=404
2950 011404  012714  141401                  MOV     #141401,(R4)    ;400=303 001
2951 011410  012711  177405                  MOV     #177405,(R1)    ;402=377 005
2952 011414  012722  000070                  MOV     #70,(R2)+       ;404=2070
2953 011420  012722  177777                  MOV     #-1,(R2)+       ;406=-1
2954 011424  042421                          BIC     (R4)+,(R1)+     ;402=074004
2955 011426  001401                          BEQ     1$              ;ERROR IF ZERO
2956 011430  100001                          BPL     2$              ;BRANCH IF GOOD
2957                                                                 ;CC SHOULD = 1000
2958 011432  104001                  1$:     ERROR   +1              ;CPU ERROR
2959 011434  052421                  2$:     BIS     (R4)+,(R1)+     ;404=074074
2960 011436  142421                          BICB    (R4)+,(R1)+     ;406=074
2961 011440  005301                          DEC     R1              ;R4=405 R1=406
2962 011442  152421                          BISB    (R4)+,(R1)+     ;406=-1 R4=406 R1=407
2963 011444  100401                          BMI     4$              ;BRANCH IF GOOD
2964                                                                 ;406 SHOULD=-1
2965 011446  104001                  3$:     ERROR   +1              ;CPU ERROR
2966 011450  005214                  4$:     INC     (R4)            ;406 SHOULD=0
2967 011452  001401                          BEQ     6$              ;BRANCH IF GOOD
2968                                                                 ;ERROR! 406 NE 0
2969 011454  104001                  5$:     ERROR   +1              ;CPU ERROR
2970 011456                          6$:
2971
2972                                 ;
2975 011456                          MBC22:
2976
2977                                 ;       TEST BIT, CMP MODE 2,2
2978 011456  012704  000400                  MOV     #400,R4         ;R4=400
2979 011462  012701  000402                  MOV     #402,R1         ;R1=402
2980 011466  012714  125252                  MOV     #125252,(R4)    ;400=125252
2981 011472  012721  100001                  MOV     #100001,(R1)+   ;402=100001
2982 011476  012711  100002                  MOV     #100002,(R1)    ;404=100002
2983 011502  005741                          TST      (R1)           ;R1=402
2984 011504  132421                          BITB    (R4)+,(R1)+     ;**ANDED RESULT= 000
2985 011506  100401                          BMI     1$              ;ERROR IF MINUS
2986 011510  001401                          BEQ     2$              ;BRANCH IF GOOD
2987 011512  104001                  1$:     ERROR   +1              ;CPU ERROR
2988                                                         ;CC SHOULD = 0100
2989 011514  132124                  2$:     BITB    (R1)+,(R4)+     ;** ANDED RESULT = 200
2990 011516  001401                          BEQ     3$              ;ERROR IF EQUAL
2991 011520  100401                          BMI     4$              ;BRANCH IF GOOD
2992 011522  104001                  3$:     ERROR   +1              ;CPU ERROR
2993                                                         ;CC SHOULD= 1000 R4=402 R1=404
2994 011524  022421                  4$:     CMP     (R4)+,(R1)+     ;RESULT =+1
2995 011526  001402                          BEQ     5$              ;ERROR IF EQUAL
```

```
********** DOUBLE OPERAND TESTS *****************

 2996 011530  103001                    BCC     5$              ;ERROR IF NO CARRY
 2997 011532  100401                    BMI     6$              ;BRANCH IF GOOD
 2998 011534  104001              5$:   ERROR   ·1              ;CPU ERROR
 2999                                                 ;CC SHOULD = 0000
 3000 011536  005341              6$:   DEC     -(R1)           ;404-100001
 3001 011540  005741                    TST     (R1)            ;R4=404 R1=402
 3002 011542  022124                    CMP     (R1)·,(R4)·     ;RESULT =0
 3003 011544  001401                    BEQ     8$              ;BRANCH IF GOOD
 3004                                                           ;CC SHOULD = 0100 R1=404 R4=406
 3005 011546  104001              7$:   ERROR   ·1              ;CPU ERROR
 3006 01155(                      8$:
 3007
 3008                             ;
 3011 011550                      MS33:
 3012
 3013                             ;       TEST SUB MODE 3,3
 3014 011550  005004                    CLR     R4              ;R4=0
 3015 011552  012701  000002              MOV     #2,R1           ;R1=2
 3016 011556  012702  000400              MOV     #400,R2         ;R2=400
 3017 011562  012714  000400              MOV     #400,(R4)       ;0=400
 3018 011566  012711  000402              MOV     #402,(R1)       ;2=402
 3019 011572  012722  000200              MOV     #200,(R2)·      ;400=200
 3020 011576  012712  054320              MOV     #54320,(R2)     ;402=54320
 3021 011602  163431                    SUB     @(R4)·,@(R1)·   ;54320   200-54120
 3022 011604  001402                    BEQ     1$              ;ERROR IF ZERO
 3023 011606  103401                    BCS     1$              ;ERROR IF CARRY
 3024 011610  100001                    BPL     2$              ;BRANCH IF GOOD
 3025 011612  104001              1$:   ERROR   ·1              ;CPU ERROR
 3026                                                 ;CC SHOULD =0001
 3027 011614  022712  054120      2$:   C       #54120,(R2)     ; TEST R4 AUTO INC AND RESULT
 3028 011620  001401                    BEQ     4$              ;BRANCH IF GOOD
 3029 011622  104001              3$.   ERROR   ·1              ;CPU ERROR
 3030                                                 ;CC SHOULD = 0100 R4=2 R1=4
 3031 011624  005067  166150      4$:   CLR     0               ;RESTORE VECTORS
 3032 011630  005067  166146            CLR     2               ;       "
 3033
 3034
 3035                             ;
 3038 011634                      MCB44:
 3039
 3040                             ;       TEST CMP, BIT MODE 4,4
 3041 011634  012704  000400              MOV     #400,R4                 ;R4=400
 3042 011640  012701  000402              MOV     #402,R1                 ;R1=402
 3043 011644  012721  125366              MOV     #125366,(R1)·           ;402=125366 R1=404
 3044 011650  012724  173001              MOV     #173001,(R4)·           ;400=173001 R4=402
 3045 011654  024441                      CMP     -(R4),-(R1)             ;173001 - 125366=045603 CC=0000
 3046 011656  103401                      BCS     1$                      ;ERROR IF CARRY
 3047 011660  100001                      BPL     2$                      ;BRANCH IF GOOD
 3048                                                                     ;BAD COMPARE
 3049 011662  104001              1$:   ERROR   ·1              ;CPU ERROR
 3050 011664  005204              2$:   INC     R4
 3051 011666  005201                    INC     R1                      ;R1=403
 3052 011670  000261                    SEC                             ;SET CARRY
 3053 011672  134144                    BITB    -(R1),-(R4)             ;173↑1=0
 3054 011674  103001                    BCC     3$                      ;C SHOULD REMAIN SET
 3055 011676  001401                    BEQ     4$                      ;BRANCH IF GOOD
 3056                                                                     ;INCORRECT COMPARE R4=400 R1=402
```

```
********** DOUBLE OPERAND TESTS *****************

3057 011700  104001                  3$:    ERROR   ;1              ;CPU ERROR
3058 011702  005724                  4$:    TST     (R4)+                   ;R4=402
3059 011704  005201                         INC     R1                      ;R1=403
3060 011706  124441                         CMPB    -(R4),-(R1)             ;173   173=0
3061 011710  001401                         BEQ     6$                      ;BRANCH IF GOOD
3062                                                                         ;BAD COMPARE
3063 011712  104001                  5$:    ERROR   ;1              ;CPU ERROR
3064 011714                          6$:
3065                                 ;
3066 011714                          MA55:
3069                                 ;      TEST ADD MODE 5,5
3071 011714  012704  000400                 MOV     #400,R4
3072 011720  012724  000001                 MOV     #1,(R4)+                ;400=1
3073 011724  012724  177776                 MOV     #-2,(R4)+               ;402=-2
3074 011730  012724  000400                 MOV     #400,(R4)+              ;404=400
3075 011734  012714  000402                 MOV     #402,(R4)               ;406=402 R4=406
3076 011740  012701  000410                 MOV     #410,R1                 ;R1=410
3077 011744  065451                         ADD     @-(R4),@-(R1)           ;1+ -2= -1
3078 011746  001402                         BEQ     1$                      ;ERROR IF ZERO
3079 011750  100001                         BPL     1$                      ;ERROR IF PLUS
3080 011752  103001                         BCC     2$                      ;BRANCH IF GOOD
3081 011754  104001                  1$:    ERROR   ;1              ;CPU ERROR
3082                                                                         ;BAD ADD
3083 011756  062704  000004          2$:    ADD     #4,R4                   ;R4=410
3084 011762  065154                         ADD     @-(R1),@ (R4)           ;-1 + 1 = 0
3085 011764  001401                         BEQ     4$                      ;BRANCH IF GOOD
3086 011766  104001                  3$:    ERROR   ;1              ;CPU ERROR
3087                                                                         ;CC SHOULD= 0100 R4=406 R1=402
3088 011770                          4$:
3089
3090                                 ;
3093                                 ;
3094                                 ;
3095                                 ;-----------------------------------
3096                                 ;TEST DOP BIT(B) MODE 6,6
3097                                 ;
3098                                        ;
3099                                        ;
3100                                        ;
3101 011770                          MB66:
3102
3103                                 ;      TEST BIT, BITB MODE 6,6
3104 011770  005004                         CLR     R4                      ;R4=0
3105 011772  012701  000400                 MOV     #400,R1                 ;
3106 011776  012721  125252                 MOV     #125252,(R1)+           ;400=125252
3107 012002  012721  000001                 MOV     #1,(R1)+                ;402=1
3108 012006  012721  100000                 MOV     #100000,(R1)+           ;404=1000000  R1=406
3109 012012  036461  000400  177774         BIT     400(R4),-4(R1)          ;(400)&(402)=0
3110 012020  001401                         BEQ     2$                      ;BRANCH IF GOOD
3111                                                                         ;CC SHOULD = 0100
3112 012022  104001                  1$:    ERROR   ;1              ;CPU ERROR
3113 012024  136461  000405  177772  2$:    BITB    405(R4),-6(R1)          ;(405)&(400)=200
3114 012032  001401                         BEQ     3$                      ;ERROR IF ZERO
3115 012034  100401                         BMI     4$                      ;BRANCH IF GOOD
3116                                                                         ;CC SHOULD = 1000
3117 012036  104001                  3$:    ERROR   ;1              ;CPU ERROR
```

M5

********** DOUBLE OPERAND TESTS **********

```
3118 012040                                4$:
3119
3120                                       ;
3123 012040                                MS77:
3124
3125                                       ;       TEST SUB MODE 7,7
3126 012040  012704  000400                        MOV     #400,R4         ;
3127 012044  005001                                CLR     R1
3128 012046  012724  177776                        MOV     #-2,(R4)+               ;400= 2
3129 012052  012724  177777                        MOV     #-1,(R4)+               ;402=-1
3130 012056  012724  000400                        MOV     #400,(R4)+              ;404=400 R4=406
3131 012062  012711  000402                        MOV     #402,(R1)               ;0=402
3132 012066  005201                                INC     R1                      ;R1=1
3133 012070  167471  177372  000403                SUB     @-406(R4),@403(R1)      ;-2 -  1 =  1
3134 012076  001401                                BEQ     1$                      ;ERROR  IF ZERO
3135 012100  100401                                BMI     2$                      ;BRANCH IF GOOD
3136                                                                               ;CC SHOULD=1000
3137 012102  104001                        1$:     ERROR   +1              ;CPU ERROR
3138 012104  167174  177777  177776        2$:     SUB     @-1(R1),@-2(R4)         ;-1 -  1 = 0
3139 012112  001401                                BEQ     4$                      ;BRANCH IF GOOD
3140 012114  104001                        3$      ERROR   +1              ;CPU ERROR
                                                                           ;ERROR ON SUBTRACT 400-0
3141
3142 012116  005067  165656                4$:     CLR     0                       ;RESTORE VECTORS
3143 012122  005067  165654                        CLR     2                       ;       "
3144
3145
3146                                       ;
3149 012126                                MRL0:
3150
3151                                       ;       TEST ROL, ROLB MODE 0
3152 012126  012704  125252                        MOV     #125252,R4              ;R4=125252
3153 012132  000277                                SCC                             ;CC=1111
3154 012134  006104                                ROL     R4                      ;R4=052525 WITH CARRY SET
3155 012136  102004                                BVC     1$                      ;ERROR IF V CLEAR
3156 012140  103003                                BCC     1$                      ;ERROR IF CARRY CLEAR
3157 012142  022704  052525                        CMP     #052525,R4              ;SEE IF R0 = EXPECTED
3158 012146  001401                                BEQ     2$                      ;ERROR IF R4 NE EXPECTD
3159 012150  104001                        1$:     ERROR   +1              ;CPU ERROR
                                                                           ;ROL FAILED, CC SHOULD=0011
3160
3161 012152  012704  125252                2$:     MOV     #125252,R4              ;R4=125252
3162 012156  000257                                CCC                             ;CC=0000
3163 012160  106104                                ROLB    R4                      ;ROTATE EVEN BYTE
3164 012162  103005                                BCC     3$                      ;ERROR IF NO CARRY
3165 012164  102004                                BVC     3$                      ;ERROR IF NO OVERFLOW
3166 012166  100403                                BMI     3$                      ;ERROR IF MINUS
3167 012170  022704  125124                        CMP     #125124,R4              ;SEE IF R4 = EXPECTED
3168 012174  001401                                BEQ     4$                      ;BRANCH IF GOOD
3169 012176  104001                        3$:     ERROR   +1              ;CPU ERROR
                                                                           ;ROLB FAILED, CC SHOULD=1011, R4=125125
3170
3171 012200                                4$:
3172
3173                                       ;
3176 012200                                MRLB1:
3177
3178                                       ;       TEST ROL, ROLB MODE 1
3179 012200  005004                                CLR     R4                      ;R4=0
3180 012202  012714  052525                        MOV     #52525,(R4)             ;0=52525
```

```
********** DOUBLE OPERAND TESTS *****************

3181 012206  006114                          ROL     (R4)                ;**TEST INSTRUCTION, 0=125252
3182 012210  100005                          BPL     1$                  ;ERROR IF PLUS
3183 012212  102004                          BVC     1$                  ;ERROR IF NO OVERFLOW
3184 012214  103403                          BCS     1$                  ;ERROR IF CARRY
3185 012216  021427  125252                  CMP     (R4),#125252        ;SEE IF R4=EXPECTED
3186 012222  001401                          BEQ     2$                  ;BRANCH IF GOOD
3187                                                                     ;BAD ROL ,CC SHOULD=1010
3188 012224  104001                  1$:     ERROR   +1          ;CPU ERROR
3189 012226  012714  125252          2$:     MOV     #125252,(R4)        ;0=125252
3190 012232  005204                          INC     R4                  ;R4=1
3191 012234  000277                          SCC                         ;CC=1111
3192 012236  106114                          ROLB    (R4)                ;**TEST INSTRUCTION
3193 012240  100406                          BMI     3$                  ;ERROR IF RESULT IS POSITIVE
3194 012242  103005                          BCC     3$                  ;ERROR IF NO CARRY
3195 012244  102004                          BVC     3$                  ;ERROR IF V CLEAR
3196 012246  005304                          DEC     R4                  ;R4=0
3197 012250  022714  052652                  CMP     #52652,(R4)         ;ERROR IF 0 NE EXPECTED
3198 012254  001401                          BEQ     4$                  ;BRANCH IF GOOD
3199 012256  104001                  3$:     ERROR   +1          ;CPU ERROR
3200                                                             ;BAD ROLB ODD BYTE,CC SHOULD=1011
3201 012260                          4$:
3202
3203                                 ;
3206 012260                          MRL2:
3207
3208                                 ;       TEST ROL, ROLB MODE 2
3209 012260  005004                          CLR     R4                  ;R4=0
3210 012262  012714  100000                  MOV     #100000,(R4)        ;0=100000
3211 012266  000257                          CCC                         ;CC=0000
3212 012270  006124                          ROL     (R4)+               ;*TEST INSTRUCTION
3213 012272  103002                          BCC     1$                  ;ERROR IF NO CARRY
3214 012274  102001                          BVC     1$                  ;ERROR IF NO OVERFLOW
3215 012276  001401                          BEQ     2$                  ;BRANCH IF GOOD
3216                                                                     ;ROL FAILED ,CCSHOULD= 0100
3217 012300  104001                  1$:     ERROR   +1          ;CPU ERROR
3218 012302  005304                  2$:     DEC     R4
3219 012304  005304                          DEC     R4
3220 012306  001012                          BNE     3$                  ;ERROR IN AUTO-DEC
3221 012310  012714  004040                  MOV     #4040,(R4)          ;0=4040
3222 012314  000241                          CLC
3223 012316  106124                          ROLB    (R4)+               ;**TEST INSTURCTION
3224 012320  103405                          BCS     3$                  ;ERROR IF CARRY SET
3225 012322  102404                          BVS     3$                  ;ERROR IF V
3226 012324  005304                          DEC     R4
3227 012326  022714  004100                  CMP     #04100,(R4)         ;SEE IF 0= EXPECTED RESULT
3228 012332  001401                          BEQ     4$                  ;BRANCH IF GOOD
3229 012334  104001                  3$:     ERROR   +1          ;CPU ERROR
3230                                                             ;BAD ROL
3231 012336                          4$:
3232
3233                                 ;
3236 012336                          MRL3:
3237
3238                                 ;       TEST ROL, ROLB MODE 3
3239 012336  005004                          CLR     R4                  ;R4=0
3240 012340  012714  052525                  MOV     #052525,(R4)                ;0=52525
3241 012344  000277                          SCC                         ;CC=1111
```

B6


```
********** DOUBLE OPERAND TESTS *****************

3242 012346  006137  000000                ROL     @#0                     ;**TEST INSRUCTION MODE 3 WITH PC
3243 012352  100005                        BPL     1$                      ;ERROR IF PLUS
3244 012354  102004                        BVC     1$                      ;ERROR IF NO OVERFLOW
3245 012356  103403                        BCS     1$                      ;ERROR IF CARRY
3246 012360  022714  125253                CMP     #125253,(R4)            ;COMPARE RESULT WITH EXPECTED
3247 012364  001401                        BEQ     2$                      ;BRANCH IF GOOD
3248                                                                       ;BAD ROL CC SHOULD=1010
3249 012366  104001              1$:       ERROR   +1              ;CPU ERROR
3250 012370  012714  125252      2$:       MOV     #125252,(R4)            ;0=125252
3251 012374  000261                        SEC                             ;CC=----1
3252 012376  106137  000000                ROLB    @#0                     ;**TEST INSTRUCTION
3253 012402  100402                        BMI     3$                      ;ERROR IF MINUS
3254 012404  103001                        BCC     3$                      ;ERROR IF NO CARRY
3255 012406  102401                        BVS     4$                      ;BRANCH IF OVERFLOW
3256 012410  104001              3$:       ERROR   +1              ;CPU ERROR
3257                                                                       ;BAD ROL, CC SHOULD=1011
3258 012412                      4$:
3259
3260                                       ;
3263 012412                      MRL4:
3264
3265                                       ;       TEST ROL MODE 4
3266 012412  005001                                CLR     R1              ;R1=0
3267 012414  012704  000002                        MOV     #2,R4           ;R4=2
3268 012420  012711  054321                        MOV     #54321,(R1)     ;0=54321
3269 012424  000277                                SCC                     ;CC=1111
3270 012426  006144                                ROL     -(R4)           ;**TEST INSTRUCTION
3271 012430  100007                                BPL     1$              ;ERROR IF PLUS
3272 012432  102006                                BVC     1$              ;ERROR IF NO OVERFLOW
3273 012434  103405                                BCS     1$              ;ERROR IF CARRY
3274 012436  022711  130643                        CMP     #130643,(R1)    ;SEE IF EXPECTED RESULT
3275 012442  001002                                BNE     1$              ;BRANCH IF ROL FAILED
3276 012444  005704                                TST     R4              ;SEE IF AUTO-DEC WORKED
3277 012446  001401                                BEQ     2$              ;BRANCH IF GOOD
3278                                                                       ;ERROR! BAD ROL INST
3279 012450  104001              1$:       ERROR   +1              ;CPU ERROR
3280 012452                      2$:
3281
3282                                       ;
3285 012452                      MMRL5:
3286
3287                                       ;       TEST ROL MODE 5
3288 012452  005004                                CLR     R4                      ;R4=0
3289 012454  012714  000400                        MOV     #400,(R4)               ;0=400
3290 012460  012734  123456                        MOV     #123456,@(R4)+          ;400=123465, R4=2
3291 012464  000277                                SCC                             ;CC=1111
3292 012466  006154                                ROL     @-(R4)                  ;**TEST INSTRUCTION
3293 012470  100410                                BMI     1$                      ;ERROR IF RESULT IS MINUS
3294 012472  103007                                BCC     1$                      ;ERROR IF NO CARRY
3295 012474  102006                                BVC     1$                      ;ERROR IF NO OVERFLOW
3296 012476  005704                                TST     R4                      ;SEE IF AUTO-DEC WORKED
3297 012500  001004                                BNE     1$                      ;ERROR OF AUTO DEC
3298 012502  022737  047135  000400                CMP     #47135,@#400            ;SEE  IF CORRECT RESULT
3299 012510  001401                                BEQ     2$                      ;BRANCH IF GOOD
3300                                                                               ;BAD ROL MODE 5
3301 012512  104001              1$:       ERROR   +1              ;CPU ERROR
3302 012514                      2$:
```

C6

********** DOUBLE OPERAND TESTS **********

```
3303
3304                                    ;
3307 012514                             MRL6:
3308
3309                                    ;       TEST ROL MODE 6
3310 012514  012704  000400                        MOV     #400,R4                 ;R4=400
3311 012520  005001                                CLR     R1                      ;R1=0
3312 012522  012711  032525                        MOV     #32525,(R1)             ;0=32525
3313 012526  000277                                SCC
3314 012530  006164  177400                        ROL     -400(R4)                ;**TEST INSTRUCTION
3315 012534  100405                                BMI     1$                      ;ERROR IF MINUS
3316 012536  103404                                BCS     1$                      ;ERROR IF CARRY
3317 012540  102403                                BVS     1$                      ;ERROR IF OVERFLOW
3318 012542  022711  065253                        CMP     #65253,(R1)             ;SEE IF CORRECT RESULT
3319 012546  001401                                BEQ     2$                      ;BRANCH IF GOOD
3320                                                                               ;BAD ROL MODE 6
3321 012550  104001                     1$:        ERROR   +1              ;CPU ERROR
3322 012552                             2$:
3323
3324                                    ;
3327 012552                             MRL7:
3328
3329                                    ;       TEST ROL MODE 7
3330 012552  012704  000400                        MOV     #400,R4                 ;R4=400
3331 012556  005037  000402                        CLR     @#402                   ;402=0
3332 012562  012737  100000  000000                MOV     #100000,@#0             ;0=100000
3333 012570  006174  000002                        ROL     @2(R4)                  ;**TEST INSTRUCTION
3334 012574  100406                                BMI     1$                      ;ERROR IF MINUS
3335 012576  001005                                BNE     1$                      ;ERROR IF NOT ZERO
3336 012600  103004                                BCC     1$                      ;ERROR IF NO CARRY
3337 012602  102003                                BVC     1$                      ;ERROR IF NO OVERFLOW
3338 012604  005737  000000                        TST     @#0                     ;CHECK RESULT
3339 012610  001401                                BEQ     2$                      ;BRANCH IF GOOD
3340                                                                               ;BAD ROL MODE 7
3341 012612  104001                     1$:        ERROR   +1              ;CPU ERROR
3342 012614                             2$:
3343
3344                                    ;
3405 012614                             MSW37:
3406
3407                                    ;       TEST SWAB MODE 37
3408 012614  012704  000400                        MOV     #400,R4                 ;R4=400
3409 012620  012714  040700                        MOV     #40700,(R4)             ;400= 101 300
3410 012624  000337  000400                        SWAB    @#400                   ;400 SHOULD = 300 101
3411 012630  100406                                BMI     1$                      ;ERROR IF MINUS
3412 012632  022714  140101                        CMP     #140101,(R4)            ;SEE IF EXPECTED RESULT
3413 012636  001003                                BNE     1$                      ;BRANCH IF BAD
3414 012640  000337  000400                        SWAB    @#400                   ;400=101 300
3415 012644  100401                                BMI     2$                      ;BRANCH IF GOOD
3416                                                                               ;ERROR! BAD SWAB MODE 37
3417 012646  104001                     1$:        ERROR   +1              ;CPU ERROR
3418 012650                             2$:
3419
3420                                    ;
3423 012650                             MRR0:
3424
3425                                    ;       TEST ROR MODE 0
```

D6


```
********** DOUBLE OPERAND TESTS *****************

 3426 012650  012704  052525                    MOV     #52525,R4               ;R4=52525
 3427 012654  000257                            CCC                             ;CC=0000
 3428 012656  006004                            ROR     R4                      ;R4 SHOULD = 25252 WITH CARRY
 3429 012660  103003                            BCC     1$                      ;ERROR IF NO CARRY
 3430 012662  022704  025252                    CMP     #25252,R4               ;SEE IF R4= EXPECTED
 3431 012666  001401                            BEQ     2$                      ;BRANCH IF GOOD
 3432                                                                           ;ROR MODE 0 FAILED
 3433 012670  104001                  1$:       ERROR   +1              ;CPU ERROR
 3434 012672                          2$:
 3435
 3436                                 ;
 3439 012672                          MRRB1:
 3440
 3441                                 ;       TEST RORB MODE 1
 3442 012672  005004                            CLR     R4                      ;R4=0
 3443 012674  012714  000001                    MOV     #1,(R4)                 ;0=1
 3444 012700  000277                            SCC                             ;CC=1111
 3445 012702  106014                            RORB    (R4)                    ;0=000200, NO C
 3446 012704  103004                            BCC     1$                      ;ERROR IF NO CARRY
 3447 012706  100003                            BPL     1$                      ;ERROR IF PLUS
 3448 012710  022714  000200                    CMP     #200,(R4)               ;CHECK RESULT
 3449 012714  001401                            BEQ     2$                      ;BRANCH IF GOOD
 3450 012716  104001                  1$:       ERROR   +1              ;CPU ERROR
 3451                                                                           ;BAD RESULT
 3452 012720                          2$:
 3453
 3454                                 ;
 3456
 3458 012720                          MJ:
 3459
 3460                                 ;       TEST JMP - ALL MODES
 3461 012720  012737  000001  003072            MOV     #1,@#SEQ                ;SETUP TEST SEQUENCER
 3462 012726  012701  012772                    MOV     #MJU1,R1                ;SET MODE 1 JUMP ADDRESS
 3463 012732  000111                            JMP     (R1)                    ;*JMP MODE 1
 3464 012734  023727  003072  000002  MJU2:     CMP     @#SEQ,#2                ;CHECK FOR CORRECT SEQUENCE
 3465 012742  001401                            BEQ     MJU2A                   ;BRANCH IF GOOD
 3466                                                                           ;MODE 2 JUMP FAILED
 3467 012744  104001                  1$:       ERROR   +1              ;CPU ERROR
 3468 012746  020127  012736          MJU2A:    CMP     R1,#MJU2+2              ;CHECK FOR AUTO-INCREMENT
 3469 012752  001401                            BEQ     MJU2B                   ;BRANCH IF GOOD
 3470 012754  104001                  2$:       ERROR   +1              ;CPU ERROR
 3471                                                                           ;AUTO-INCR FAILED
 3472 012756  005237  003072          MJU2B:    INC     @#SEQ                   ;UPDATE TEST SEQUENCER
 3473 012762  012701  012770                    MOV     #MJ2,R1                 ;SETUP MODE 3 JUMP
 3474 012766  000131                            JMP     @(R1)+                  ;JUMP MODE 3
 3475 012770  013016                  MJ2:      .WORD   MJU3                    ;MODED 3 DETINATION
 3476 012772  023727  003072  000001  MJU1:     CMP     @#SEQ,#1                ; TEST FOR CORRECT SEQUENCE
 3477 013000  001401                            BEQ     MJU1A                   ;BRANCH IF GOOD
 3478 013002  104000                  3$:       ERROR   +1              ;CPU ERROR
 3479                                                                           ;JMP OUT OF SEQUENCE
 3480 013004  005237  003072          MJU1A:    INC     @#SEQ                   ;UPDATE SEQUENCE
 3481 013010  012701  012734                    MOV     #MJU2,R1                ;SETUP MODE 2 DESTINATION
Z 3482 013014  000121                            JMP     (R1)+                   ;JUMP MODE 2
 3483 013016  023727  003072  000003  MJU3:     CMP     @#SEQ,#3                ; TEST FOR CORRECT SEQUENCE
 3484 013024  001401                            BEQ     MJU3A                   ;BRANCH IF GOOD
 3485 013026  104001                  4$:       ERROR   +1              ;CPU ERROR
 3486                                                                           ;JMP OUT OF SEQUENCE
```

E6


```
********** DOUBLE OPERAND TESTS *****************

3487 013030  022701  012772          MJU3A:  CMP     #MJ2+2,R1       ; TEST AUTO-INCREMENT
3488 013034  001401                         BEQ     MJU3B           ;BRANCH IF GOOD
3489 013036  104001                 5$:     ERROR   +1              ;CPU ERROR
3490                                                                 ;AUTO INCREMENT FAILED MODE 3
3491 013040  005237  003072          MJU3B:  INC     @#SEQ           ;UPDATE SEQUENCER
3492 013044  012701  013114                 MOV     #MJU4+2,R1      ;SETUP DESTINATION MODE 4
3493 013050  000141                         JMP     (R1)            ;EXECUTE JUMP MODE 4
3494 013052  000000                 MJU5:   HALT
3495 013054  022701  013150                 CMP     #MJ5,R1         ;CHECK AUTO-DECREMENT
3496 013060  001401                         BEQ     MJU5A           ;BRANCH IF GOOD AUTO DEC
3497 013062  104001                 6$:     ERROR   +1              ;CPU ERROR
3498                                                                 ;AUTO DEC FAIELD MODE 5
3499 013064  023727  003072  000005 MJU5A:  CMP     @#SEQ,#5        ; TEST CORRECT SEQUENCE
3500 013072  001401                         BEQ     MJU5B           ;BRANCH IF GOOD SEQUENCE
3501 013074  104001                 7$:     ERROR   +1              ;CPU ERROR
3502                                                                 ;JMP OUT OF SEQUENCE
3503 013076  005237  003072          MJU5B:  INC     @#SEQ           ;UPDATE SEQUENCE COUNT
3504 013102  012701  013145                 MOV     #MJU6-5,R1      ;SETUP DESTINATION MODE6
3505 013106  000161  000005                 JMP     +5(R1)          ;JUMP MODE 6
3506                                         ;;
3507 013112  000240                 MJU4:   NOP
3508 013114  022701  013112                 CMP     #MJU4,R1        ; TEST AUTO-DECR
3509 013120  001401                         BEQ     MJU4A           ;BRANCH IF GOOD
3510 013122  104001                 8$:     ERROR   +1              ;CPU ERROR
3511                                                                 ;MODE 4 AUTO-DEC FAILED
3512 013124  023727  003072  000004 MJU4A:  CMP     @#SEQ,#4        ; TEST FOR CORRECT SEQUENCE
3513 013132  001401                         BEQ     MJU4B           ;BRANCH IF CORRECT SEQUENCE
3514 013134  104001                 9$:     ERROR   +1              ;CPU ERROR
3515                                                                 ;INCORRECT JMP SEQUFNCE
3516 013136  005237  003072          MJU4B:  INC     @#SEQ           ;UPDATE SEQUENCE
3517 013142  012701  013152                 MOV     #MJ5+2,R1       ;SETUP MODE 5 POINTER
3518 013146  000151                         JMP     @-(R1)          ;EXECUTE MODE 5 JMP
3519                                         ;
3520 013150  013054                 MJ5:    .WORD   MJU5+2          ;POINTER MODE 5
3521                                         ;
3522 013152  022737  000006  003072 MJU6:   CMP     #6,@#SEQ        ;CHECK FOR CORRECT SEQUENCE
3523 013160  001401                         BEQ     MJU6A           ;BRANCH IF GOOD
3524 013162  104001                 10$:    ERROR   +1              ;CPU ERROR
3525                                                                 ;INCORRECT SEQUENCE
3526 013164  005237  003072          MJU6A:  INC     @#SEQ           ;UPDATE SEQUENCER
3527 013170  012701  013210                 MOV     #MJ7+10,R1      ;SETUP INDEX
3528 013174  000171  177770                 JMP     @-10(R1)        ;EXECUTE MODE 7 JUMP
3529 013200  013204                 MJ7:    .WORD   MJU7            ;POINTER FOR MODE 7
3530 013202  000000                         HALT
3531 013204  022737  000007  003072 MJU7:   CMP     #7,@#SEQ        ; TEST FOR CORRECT SEQUENCE
3532 013212  001401                         BEQ     MJU7E           ;BRANCH IF GOOD SEQUENCE
3533 013214  104001                 11$:    ERROR   +1              ;CPU ERROR
3534                                                                 ; TESTING OUT OF SEQUENCE
3535 013216                         MJU7E:
3536                                         ;
3547
3548                                         ;       TEST THAT PRE-FETCH BUFFER CAN BE OVER WRITTEN
3549 013216  012701  013526                 MOV     #128$,R1        ;SET UP R1 WITH ADDRESS OF ERROR
3550                                                                 ;ROUTINE
3551         000040                 .REPT   32.
3553 013222  012717                         MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
     013224  000240                         .WORD   NOP             ;NOP INSTRUCTION
```

********** DOUBLE OPERAND TESTS **********

```
013226  000111  30008$: .WORD   111             ;JMP     (R1)
013230  012717          MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
013232  000240          .WORD   NOP             ;NOP INSTRUCTION
013234  000111  30009$: .WORD   111             ;JMP     (R1)
013236  012717          MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
013240  000240          .WORD   NOP             ;NOP INSTRUCTION
013242  000111  30010$: .WORD   111             ;JMP     (R1)
013244  012717          MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
013246  000240          .WORD   NOP             ;NOP INSTRUCTION
013250  000111  30011$: .WORD   111             ;JMP     (R1)
013252  012717          MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
013254  000240          .WORD   NOP             ;NOP INSTRUCTION
013256  000111  30012$: .WORD   111             ;JMP     (R1)
013260  012717          MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
013262  000240          .WORD   NOP             ;NOP INSTRUCTION
013264  000111  30013$: .WORD   111             ;JMP     (R1)
013266  012717          MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
013270  000240          .WORD   NOP             ;NOP INSTRUCTION
013272  000111  30014$: .WORD   111             ;JMP     (R1)
013274  012717          MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
013276  000240          .WORD   NOP             ;NOP INSTRUCTION
013300  000111  30015$: .WORD   111             ;JMP     (R1)
013302  012717          MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
013304  000240          .WORD   NOP             ;NOP INSTRUCTION
013306  000111  30016$: .WORD   111             ;JMP     (R1)
013310  012717          MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
013312  000240          .WORD   NOP             ;NOP INSTRUCTION
013314  000111  30017$: .WORD   111             ;JMP     (R1)
013316  012717          MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
013320  000240          .WORD   NOP             ;NOP INSTRUCTION
013322  000111  30018$: .WORD   111             ;JMP     (R1)
013324  012717          MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
013326  000240          .WORD   NOP             ;NOP INSTRUCTION
013330  000111  30019$: .WORD   111             ;JMP     (R1)
013332  012717          MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
013334  000240          .WORD   NOP             ;NOP INSTRUCTION
013336  000111  30020$: .WORD   111             ;JMP     (R1)
013340  012717          MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
013342  000240          .WORD   NOP             ;NOP INSTRUCTION
013344  000111  30021$: .WORD   111             ;JMP     (R1)
013346  012717          MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
013350  000240          .WORD   NOP             ;NOP INSTRUCTION
013352  000111  30022$: .WORD   111             ;JMP     (R1)
013354  012717          MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
013356  000240          .WORD   NOP             ;NOP INSTRUCTION
013360  000111  30023$: .WORD   111             ;JMP     (R1)
013362  012717          MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
013364  000240          .WORD   NOP             ;NOP INSTRUCTION
013366  000111  30024$: .WORD   111             ;JMP     (R1)
013370  012717          MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
013372  000240          .WORD   NOP             ;NOP INSTRUCTION
013374  000111  30025$: .WORD   111             ;JMP     (R1)
013376  012717          MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
013400  000240          .WORD   NOP             ;NOP INSTRUCTION
013402  000111  30026$: .WORD   111             ;JMP     (R1)
013404  012717          MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
013406  000240          .WORD   NOP             ;NOP INSTRUCTION
```

********** DOUBLE OPERAND TESTS **********

```
                013410  000111          30027$: .WORD   111             ;JMP    (R1)
                013412  012717                  MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
                013414  000240                  .WORD   NOP             ;NOP INSTRUCTION
                013416  000111          30028$: .WORD   111             ;JMP    (R1)
                013420  012717                  MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
                013422  000240                  .WORD   NOP             ;NOP INSTRUCTION
                013424  000111          30029$: .WORD   111             ;JMP    (R1)
                013426  012717                  MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
                013430  000240                  .WORD   NOP             ;NOP INSTRUCTION
                013432  000111          30030$: .WORD   111             ;JMP    (R1)
                013434  012717                  MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
                013436  000240                  .WORD   NOP             ;NOP INSTRUCTION
                013440  000111          30031$: .WORD   111             ;JMP    (R1)
                013442  012717                  MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
                013444  000240                  .WORD   NOP             ;NOP INSTRUCTION
                013446  000111          30032$: .WORD   111             ;JMP    (R1)
                013450  012717                  MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
                013452  000240                  .WORD   NOP             ;NOP INSTRUCTION
                013454  000111          30033$: .WORD   111             ;JMP    (R1)
                013456  012717                  MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
                013460  000240                  .WORD   NOP             ;NOP INSTRUCTION
                013462  000111          30034$: .WORD   111             ;JMP    (R1)
                013464  012717                  MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
                013466  000240                  .WORD   NOP             ;NOP INSTRUCTION
                013470  000111          30035$: .WORD   111             ;JMP    (R1)
                013472  012717                  MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
                013474  000240                  .WORD   NOP             ;NOP INSTRUCTION
                013476  000111          30036$: .WORD   111             ;JMP    (R1)
                013500  012717                  MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
                013502  000240                  .WORD   NOP             ;NOP INSTRUCTION
                013504  000111          30037$: .WORD   111             ;JMP    (R1)
                013506  012717                  MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
                013510  000240                  .WORD   NOP             ;NOP INSTRUCTION
                013512  000111          30038$: .WORD   111             ;JMP    (R1)
                013514  012717                  MOV     (PC)+,(PC)      ;WRITE THE NOP OVER THE JMP INSTRUCTION
                013516  000240                  .WORD   NOP             ;NOP INSTRUCTION
                013520  000111          30039$: .WORD   111             ;JMP    (R1)
 3554           013522  000137  013530          JMP     @#129$          ;JUMP OVER ERROR CALL
 3555           013526                  128$:                           ;ERROR! PRE-FETCH BUFFER WAS NOT
 3556                                                                   ;OVER WRITTEN
 3557           013526  104001                  ERROR   +1              ;CPU ERROR
 3558                                  ;
 3559                                  ;       NOW RESTORE THE OVER WRITTEN JMP INSTRUCTIONS FOR THE NEXT PASS.
 3560                                  ;
 3561           013530  012702  000040  129$:   MOV     #32.,R2                 ;SET UP R2 AS COUNTER
 3562           013534  012703  013216          MOV     #MJU7E,R3               ;SET UP R3 AS POINTER. $$$
 3563           013540  062703  000010          ADD     #10,R3          ;ADD OFFSET TO POINT TO 1st. JMP IN TABLE. $$$
 3564           013544  012713  000111  130$:   MOV     #111,(R3)               ;RESTORE OVER WRITTEN JUMPS
 3565           013550  062703  000006          ADD     #6,R3                   ;POINT TO NEXT OVER WRITTEN ADDR.
 3566           013554  077205                  SOB     R2,130$                 ;DO UNTIL R2=0
 3567
 3569                                  ;
 3571           013556                  MJP:
 3572
 3573                                  ;       TEST JMP MODES 17,27,37,67,77
 3574           013556  012737  000000  003072        MOV     #0,@#SEQ                ;SETUP TEST SEQUNCER
 3575           013564  000117                  JMP     (R7)                    ;JUMP MODE 17(SHOULD BE IN-LINE)
```

H6

********** DOUBLE OPERAND TESTS ********** **********

```
3576
3577 013566  005737  003072          MJP17:  TST     @#SEQ                   ;CHECK SEQUENCE
3578 013572  001401                         BEQ     2$                      ;BRANCH IF GOOD
3579                                                                        ;ERROR! BAD JUMP
3580 013574  104001                 1$:     ERROR   +1              ;CPU ERROR
3581 013576  005237  003072          2$:     INC     @#SEQ                   ;UPDATE SEQUENCE NUMBER
Z    3582 013602  000127  000401                 JMP     #401                    ;JUMP MODE 27
3583                                                                        ;(THE #401=BR UPDATED PC+2)
3584                                        ;
3585 013606  000000                         HALT                            ;
3586                                        ;
3587 013610  023727  003072  000001  MJP27:  CMP     @#SEQ,#1                ;CHECK IF CORRECT SEQUENCE
3588 013616  001401                         BEQ     MJP27A                  ;BRANCH IF IN SEQUENCE
3589 013620  104001                 1$:     ERROR   +1              ;CPU ERROR
3590                                                                ; TEST OUT OF SEQUENCE
3591 013622  005237  003072          MJP27A: INC     @#SEQ                   ;UPDATE SEQUENCER
3592 013626  000137  013674                 JMP     @#MJP37                 ;JUMP MODE 37
3593 013632  023727  003072  000003  MJP67:  CMP     @#SEQ,#3                ;CHECK FOR CORRECT SEQUENCE
3594 013640  001401                         BEQ     MJP67A                  ;BRANCH IF IN SEQUENCE
3595 013642  104001                 2$:     ERROR   +1              ;CPU ERROR
3596                                                                ; TEST OUT OF SEQUENCE
3597 013644  005237  003072          MJP67A: INC     @#SEQ                   ;UPDATE SEQUENCER
3598 013650  000257                         CCC                             ;INSURE ZBIT=0
3599 013652  000177  000002                 JMP     @MJP67B                 ;JUMP MODE 7
3600                                        ;
3601 013656  000000                         HALT                            ;
3602                                        ;
3603 013660  013662                 MJP67B: .WORD   MJP77                   ;
3604                                        ;
3605 013662  023727  003072  000004  MJP77:  CMP     @#SEQ,#4                ; TEST FOR CORRECT SEQUENCE
3606 013670  001412                         BEQ     MJP77E                  ;BRANCH IF IN SEQUENCE
3607 013672  104001                 3$:     ERROR   +1              ;CPU ERROR
3608                                                                ; TEST OUT OF SEQUENCE
3609 013674  023727  003072  000002  MJP37:  CMP     @#SEQ,#2                ; TEST FOR CORRECT SEQUENCE
3610 013702  001401                         BEQ     MJP37A                  ;BRANCH IF IN SEQUENCE
3611 013704  104001                 4$:     ERROR   +1              ;CPU ERROR
3612                                                                ; TEST OUT OF SEQUENCE
3613 013706  005237  003072          MJP37A: INC     @#SEQ                   ;UPDATE SEQUENCER
3614 013712  000167  177714                 JMP     MJP67                   ;JUMP MODE 6
3615                                        ;
3616                                        ;
3617 013716                         MJP77E:
3618                                        ;
3619                                        ;
3622 013716                         MJSR:
3623
3624                                        ;       TEST JSR ALL MODES
3625 013716  010637  003074                 MOV     R6,@#SPS                ;SAVE STACK POINTER LOCATION
3626 013722  010637  003076                 MOV     R6,@#SPSJ               ;
3627 013726  162737  000002  003076         SUB     #2,@#SPSJ               ;SPSJ = R6 AFTER DECRIMENT
3628 013734  012737  000001  003072         MOV     #1,@#SEQ                ;SETUP SEQUENCE COUNTER
3629 013742  012701  014036                 MOV     #MJSR1,R1               ;SETUP INITIAL JUMP IN MODE 1
3630 013746  005004                         CLR     R4                      ;
3631 013750  005104                         COM     R4                      ;R4=-1 TO BE SAVED ON STACK
3632 013752  004411                         JSR     R4,(R1)                 ;JSR MODE 1
3633                                        ;
3634 013754  022737  000002  003072  MJSR2:  CMP     #2,@#SEQ                ; TEST FOR CORRECT SEQUENCE
```

```
********** DOUBLE OPERAND TESTS *****************

 3635 013762  001401                        BEQ     MJSR2A          ;BRANCH IF GOOD
 3636 013764  104001                5$:     ERROR   +1              ;CPU ERROR
 3637                                                               ;MODE 2 JUMPED TO OUT OF SEQUENCE
 3638 013766  023706  003076        MJSR2A: CMP     @#SPSJ,R6       ;VERIFY STACK DECRIMENT
 3639 013772  001006                BNE     6$              ;BRANCH IF STACK INCORRECT
 3640 013774  021627  125252        CMP     (R6),#125252    ;VERIFY CONTENTS OF STACK
 3641 014000  001003                BNE     6$              ;BRANCH IF DATA ON STACK INCORRECT
 3642 014002  022704  014116        CMP     #MJSR4,R4       ;SEE IF CORRECT RETURN ADDRESS
 3643 014006  001401                BEQ     MJSR2B          ;BRANCH IF GOOD
 3644 014010  104001                6$:     ERROR   +1              ;CPU ERROR
 3645                                                               ;JSR MODE 2 FAILED
 3646 014012  005237  003072        MJSR2B: INC     @#SEQ           ;UPDATE SEQUENCE COUNTER
 3647 014016  013706  003074                MOV     @#SPS,R6        ;RELOAD STACK POINTER
 3648 014022  012701  014034                MOV     #MJSRA,R1       ;SETUP JSR MODE 3
 3649 014026  005004                        CLR     R4              ;DIFFERENT DATA TO R4
 3650 014030  004431                        JSR     R4,@(R1)+       ;JSR MODE 3
 3651 014032  000000                        HALT
 3652 014034  014202                MJSRA:  .WORD   MJSR3           ;LITERAL FOR JUMP MODE 3
 3653                                       ;
 3654 014036  022737  000001  003072 MJSR1:  CMP     #1,@#SEQ        ; TEST FOR CORRECT SEQUENCE
 3655 014044  001401                BEQ     MJSR1A          ;BRANCH IF GOOD
 3656 014046  104001                7$:     ERROR   +1              ;CPU ERROR
 3657                                                               ;MODE 1 JUMPED TO OUT OF SEQUENCE
 3658 014050  023706  003076        MJSR1A: CMP     @#SPSJ,R6       ;VERIFY STACK DECRIMENT
 3659 014054  001006                BNE     8$              ;BRANCH IF STACK INCORRECT
 3660 014056  021627  177777        CMP     (R6),#-1        ;VERIFY CONTENT OF STACK
 3661 014062  001003                BNE     8$              ;BRANCH IF DATA ON STACK INCORRECT
 3662 014064  022704  013754        CMP     #MJSR2,R4       ;SEE IF CORRECT RETURN ADDRESS
 3663 014070  001401                BEQ     MJSR1B          ;BRANCH IF GOOD
 3664 014072  104001                8$:     ERROR   +1              ;CPU ERROR
 3665                                                               ;JSR MODE 2 FAILED
 3666 014074  005237  003072        MJSR1B: INC     @#SEQ           ;UPDATE SEQUENCE COUNTER
 3667 014100  013706  003074                MOV     @#SPS,R6        ;RELOAD STACK POINTER
 3668 014104  012704  125252                MOV     #125252,R4      ;SETUP R4 DATA
 3669 014110  012701  013754                MOV     #MJSR2,R1       ;SETUP MODE 2 JUMP ADDRESS
Z 3670 014114  004421                      JSR     R4,(R1)+        ;*JUMP MODE 2
 3671                                       ;
 3672                                       ;
 3673 014116  022737  000004  003072 MJSR4:  CMP     #4,@#SEQ        ; TEST FOR CORRECT SEQUENCE
 3674 014124  001401                BEQ     MJSR4A          ;BRANCH IF GOOD
 3675 014126  104001                9$:     ERROR   +1              ;CPU ERROR
 3676                                                               ;MODE 4 JUMPED TO OUT OF SEQUENCE
 3677 014130  023706  003076        MJSR4A: CMP     @#SPSJ,R6       ;VERIFY STACK DECRIMENT
 3678 014134  001006                BNE     10$             ;BRANCH IF STACK INCORRECT
 3679 014136  021627  052525        CMP     (R6),#052525    ;VERIFY CONTENTS OF STAACK
 3680 014142  001003                BNE     10$             ;BRANCH IF DATA ON STACK INCORRECT
 3681 014144  022704  014264        CMP     #MJSR6,R4       ;SEE IF CORRECT RETURN ADDRESS
 3682 014150  001401                BEQ     MJSR4B          ;BRANCH IF GOOD
 3683 014152  104001                10$:    ERROR   +1              ;CPU ERROR
 3684                                                               ;JSR MODE 4 FAILED
 3685 014154  005237  003072        MJSR4B: INC     @#SEQ           ;UPDATE SEQUENCE COUNTER
 3686 014160  013706  003074                MOV     @#SPS,R6        ;RELOAD STACK POINTER
 3687 014164  012704  000377                MOV     #377,R4         ;SETUP R4 DATA
 3688 014170  012701  014202                MOV     #MJSRB+2,R1     ;SETUP JSR VECTOR
 3689 014174  004451                        JSR     R4,@-(R1)       ;JSR MODE 5
 3690 014176  000000                        HALT
 3691 014200  014350                MJSRB:  .WORD   MJSR5           ;MODE 5 VECTOR
```

********** DOUBLE OPERAND TESTS **********

```
3692                                    ;
3693                                    ;
3694 014202  022737  000003  003072  MJSR3:  CMP     #3,@#SEQ            ; TEST FOR CORRECT SEQUENCE
3695 014210  001401                          BEQ     MJSR3A              ;BRANCH IF GOOD
3696 014212  104001                  11$:    ERROR   +1          ;CPU ERROR
3697                                         ;MODE 3 JUMPED TO OUT OF SEQUENCE
3698 014214  023706  003076          MJSR3A: CMP     @#SPSJ,R6           ;VERIFY STACK DECRIMENT
3699 014220  001006                          BNE     12$                 ;BRANCH IF STACK INCORRECT
3700 014222  021627  000000                  CMP     (R6),#0             ;VERIFY CONTENTS OF STAACK
3701 014226  001003                          BNE     12$                 ;BRANCH IF DATA ON STACK INCORRECT
3702 014230  022704  014032                  CMP     #MJSRA 2,R4         ;SEE IF CORRECT RETURN ADDRESS
3703 014234  001401                          BEQ     MJSR3B              ;BRANCH IF GOOD
3704 014236  104001                  12$:    ERROR   +1          ;CPU ERROR
3705                                         ;JSR MODE 3 FAILED
3706 014240  005237  003072          MJSR3B: INC     @#SEQ               ;UPDATE SEQUENCE COUNTER
3707 014244  013706  003074                  MOV     @#SPS,R6            ;RELOAD STACK POINTER
3708 014250  012704  052525                  MOV     #052525,R4          ;SETUP R4 DATA
3709 014254  012701  014120                  MOV     #MJSR4+2,R1         ;SETUP JSR VECTOR
3710 014260  000257                          CCC                         ;CLEAR CONDITION CODES
3711 014262  004441                          JSR     R4, (R1)                    ;JSR MODE 4
3712                                    ;
3713                                    ;
3714 014264  022737  000006  003072  MJSR6:  CMP     #   ,@#SEQ          ; TEST FOR CORRECT SEQUENCE
3715 014272  001401                          BEQ     M   ,A              ;BRANCH IF GOOD
3716 014274  104001                  13$:    ERROR   +1          ;CPU ERROR
3717                                         ;MODE 6 JUMPED TO OUT OF SEQUENCE
3718 014276  023706  003076          MJSR6A: CMP     @#SPSJ,R6           ;VERIFY STACK DECRIMENT
3719 014302  001006                          BNE     14$                 ;BRANCH IF STACK INCORRECT
3720 014304  021627  123456                  CMP     (R6),#123456        ;VERIFY CONTENTS OF STACK
3721 014310  001003                          BNE     14$                 ;BRANCH IF DATA ON STACK INCORRECT
3722 014312  022704  014432                  CMP     #MJSR7,R4           ;SEE IF CORRECT RETURN ADDRESS
3723 014316  001401                          BEQ     MJSR6B              ;BRANCH IF GOOD
3724 014320  104001                  14$:    ERROR   +1          ;CPU ERROR
3725                                         ;JSR MODE 6 FAILED
3726 014322  005237  003072          MJSR6B: INC     @#SEQ               ;UPDATE SEQUENCE COUNTER
3727 014326  013706  003074                  MOV     @#SPS,R6            ;RELOAD STACK POINTER
3728 014332  012704  177773                  MOV     #-5,R4              ;SETUP R4 DATA
3729 014336  012701  014356                  MOV     #MJSRC+10,R1        ;SETUP JSR VECTOR
3730 014342  004471  177770                  JSR     R4,@-10(R1)         ;JSR MODE 7
3731 014346  014432                  MJSRC:  .WORD   MJSR7               ;JSR VECTOR
3732                                    ;
3733                                    ;
3734 014350  022737  000005  003072  MJSR5:  CMP     #5,@#SEQ            ; TEST FOR CORRECT SEQUENCE
3735 014356  001401                          BEQ     MJSR5A              ;BRANCH IF GOOD
3736 014360  104001                  15$:    ERROR   +1          ;CPU ERROR
3737                                         ;MODE 5 JUMPED TO OUT OF SEQUENCE
3738 014362  023706  003076          MJSR5A: CMP     @#SPSJ,R6           ;VERIFY STACK DECRIMENT
3739 014366  001006                          BNE     16$                 ;BRANCH IF STACK INCORRECT
3740 014370  021627  000377                  CMP     (R6),#377           ;VERIFY CONTENTS OF STACK
3741 014374  001003                          BNE     16$                 ;BRANCH IF DATA ON STACK INCORRECT
3742 014376  022704  014176                  CMP     #MJSRB-2,R4         ;SEE IF CORRECT RETURN ADDRESS
3743 014402  001401                          BEQ     MJSR5B              ;BRANCH IF GOOD
3744 014404  104001                  16$:    ERROR   +1          ;CPU ERROR
3745                                         ;JSR MODE 5 FAILED
3746 014406  005237  003072          MJSR5B: INC     @#SEQ               ;UPDATE SEQUENCE COUNTER
3747 014412  013706  003074                  MOV     @#SPS,R6            ;RELOAD STACK POINTER
3748 014416  012704  123456                  MOV     #123456,R4          ;SETUP DATA IN R4
```

```
********** DOUBLE OPERAND TESTS *****************

 3749 014422  012701  014274                    MOV     #MJSR6+10,R1            ;SETUP JSR VECTOR
 3750 014426  004461  177770                    JSR     R4, 10(R1)              ;JUMP MODE 6
 3751                                 ;
 3752                                 ;
 3753 014432  022737  000007  003072  MJSR7:    CMP     #7,@#SEQ                ; TEST FOR   9RECT SEQUENCE
 3754 014440  001401                            BEQ     MJSR7A                  ;BRANCH IF GOOD
 3755 014442  104001                  17$:      ERROR   +1                  ;CPU ERROR
 3756                                                                      ;MODE 7 JUMPED TO OUT OF SEQUENCE
 3757 014444  023706  003076          MJSR7A:   CMP     @#SPSJ,R6               ;VERIFY STACK DECRIMENT
 3758 014450  001006                            BNE     18$                     ;BRANCH IF STACK INCORRECT
 3759 014452  021627  177773                    CMP     (R6),# 5                ;VERIFY CONTENTS OF STAACK
 3760 014456  001003                            BNE     18$                     ;BRANCH IF DATA ON STACK INCORRECT
 3761 014460  022704  014346                    CMP     #MJSR5 2,R4             ;SEE IF CORRECT RETURN ADDRESS
 3762 014464  001401                            BEQ     MJSR7E                  ;BRANCH IF GOOD
 3763 014466  104001                  18$:      ERROR   +1                  ;CPU ERROR
 3764                                                                      ;JSR MODE 7 FAILED
 3765 014470                          MJSR7E:
 3766 014470  013706  0C3074                    MOV     @#SPS,R6                ;REPLACE STACK
 3767
 3768                                 ;
 3771 014474                          MJRA:
 3772
 3773                                 ;       TEST JSR MODES 27, 37, 67, 77
 3774 014474  012737  000001  003072            MOV     #1,@#SEQ                ;SETUP SEQUENCER
 3775 014502  010637  003074                    MOV     R6,@#SPS                ;SAVE STACK ADDRES.
 3776 014506  010637  003076                    MOV     R6,@#SPSJ               ;SAVE STACK DECRIMENT ADDRESS
 3777 014512  162737  000002  003076            SUB     #2,@#SPSJ               ;
 3778 014520  012704  177777                    MOV     #-1,R4                  ;SETUP R4 DATA
Z 3779 014524  004427  000240                   JSR     R4,#240                 ;EXECUTE A JSR MODE 27
 3780                                 ;
 3781 014530  022737  000001  003072  MJR27:    CMP     #1,@#SEQ                ;VERIFY COERRECT TEST SEQUENCE
 3782 014536  001011                            BNE     1$                      ;INCORRECT TEST SEQUENCE
 3783 014540  023706  003076                    CMP     @#SPSJ,R6               ;VERIFY STACK POINTER
 3784 014544  001006                            BNE     1$
 3785 014546  021627  177777                    CMP     (R6),#-1                ;VERIFY R4 GOT LOADED ON THE STACK
 3786 014552  001003                            BNE     1$                      ;BRANCH IF INCORRECT STACK CONTENTS
 3787 014554  020427  014530                    CMP     R4,#MJR27               ;VERIFY CORRECT RETURN ADDRESS
 3788 014560  001401                            BEQ     MJR27A                  ;BRANCH IF GOOD RETURN ADDRESS ON STACK
 3789 014562  104001                  1$:       ERROR   +1                  ;CPU ERROR
 3790                                                                      ;MODE 27 FAILED
 3791 014564  005237  003072          MJR27A:   INC     @#SEQ                   ;UPDATE SEQUENCER
 3792 014570  012704  152525                    MOV     #152525,R4              ;SETUP R4 TEST DATA
 3793 014574  013706  003074                    MOV     @#SPS,R6                ;RESET STACK POINTER
 3794 014600  004437  014666                    JSR     R4,@#MJR37              ;JSR MODE 37
 3795 014604  000000                  MJR27B:   HALT
 3796                                 ;
 3797 014606  023727  003072  000003  MJR67:    CMP     @#SEQ,#3                ;VERIFY TEST SEQUENCE
 3798 014614  001011                            BNE     2$                      ;INCORRECT TEST SEQUENCE
 3799 014616  023706  003076                    CMP     @#SPSJ,R6               ;VERIFY STACK DECRIMENT
 3800 014622  001006                            BNE     2$                      ;INCORRECT STACK DECRIMENT
 3801 014624  021627  000125                    CMP     (R6),#125               ;VERFIFY STACK WAS LOADED
 3802 014630  001003                            BNE     2$
 3803 014632  020427  014742                    CMP     R4,#MJR77               ;VERIFY RETURN ADDRESS
 3804 014636  001401                            BEQ     MJR67A                  ;BRANCH IF GOOD
 3805 014640  104001                  2$:       ERROR   +1                  ;CPU ERROR
 3806                                                                      ;MODE 67 FAILED
 3807 014642  005237  003072          MJR67A:   INC     @#SEQ                   ;UPDATE SEQUENCER
```

```
********** DOUBLE OPERAND TESTS *****************

 3808 014646  013706  003074                    MOV     @#SPS,R6                ;RESET STACK
 3809 014652  012704  000001                    MOV     #1,R4                   ;SETUP R4 DATA
 3810 014656  004477  000002                    JSR     R4,@MJR6B               ;JSR MODE 77
 3811 014662  000000                    MJR6A:  HALT
 3812 014664  014742                    MJR6B:   WORD   MJR77                   ;DATA FOR MODE 77 JUMP
 3813                                           ;
 3814 014666  023727  003072  000002    MJR37:  CMP     @#SEQ,#2                ;VERIFY TES   SEQUENCE
 3815 014674  001011                            BNE     2$                      ;INCORRECT    T SEQUENCE
 3816 014676  023706  003076                    CMP     @#SPSJ,R6               ;VERIFY ST    DECRIMENT
 3817 014702  001006                            BNE     2$                      ;INCORRECT    ACK DECRIMENT
 3818 014704  021627  152525                    CMP     (R6),#152525            ;VERFIFY STACK WAS LOADED
 3819 014710  001003                            BNE     2$
 3820 014712  020427  014604                    CMP     R4,#MJR27B              ;VERIFY RETURN ADDRESS
 3821 014716  001401                            BEQ     MJR37A                  ;BRANCH IF GOOD
 3822 014720  104001                    2$:     ERROR   +1              ;CPU ERROR
                                                                        ;MODE 37 FAILED
 3823
 3824 014722  005237  003072            MJR37A: INC     @#SEQ                   ;UPDATE SEQUENCER
 3825 014726  013706  003074                    MOV     @#SPS,R6                ;RELOAD STACK
 3826 014732  012704  000125                    MOV     #125,R4                 ;SETUP R4 TEST DAT-
 3827 014736  004467  177644                    JSR     R4,MJR67                ;JSR MODE 6
 3828                                           ;
 3829 014742  023727  003072  000004    MJR77:  CMP     @#SEQ,#4                ;VERIFY TEST SEQUENCE
 3830 014750  001011                            BNE     2$                      ;INCORRECT TEST SEQUENCE
 3831 014752  023706  003076                    CMP     @#SPSJ,R6               ;VERIFY STACK DECRIMENT
 3832 014756  001006                            BNE     2$                      ;INCORRECT STACK DECRIMENT
 3833 014760  021627  000001                    CMP     (R6),#1                 ;VERFIFY STACK WAS LOADED
 3834 014764  001003                            BNE     2$
 3835 014766  020427  014662                    CMP     R4,#MJR6A               ;VERIFY RETURN ADDRESS
 3836 014772  001401                            BEQ     MJR77A                  ;BRANCH IF GOOD
 3837 014774  104001                    2$:     ERROR   +1              ;CPU ERROR
                                                                        ;MODE 77 FAILED
 3838
 3839 014776                            MJR77A:
 3840                                           ;
 3841 014776  013706  003074                    MOV     @#SPS,R6                ;RESET STACK
 3842
 3843                                           ;
 3846 015002                            MRTS:
 3847
 3848                                           ;       TEST RTS AND RTS R6
 3849 015002  012706  001000                    MOV     #STBOT,R6               ;INSURE VALID STACK
 3850 015006  012746  123456                    MOV     #123456,-(R6)           ;SETUP TEST REGISTER
 3851 015012  012703  015022                    MOV     #RTS1,R3                ;SETUP TEST PC
 3852 015016  000203                            RTS     R3                      ;**TEST INSTRUCTION
 3853 015020  104001                            ERROR   +1              ;CPU ERROR
                                                                        ;INCORRECT PC ON RTS
 3854
 3855 015022  022703  123456            RTS1:   CMP     #123456,R3
 3856 015026  001401                            BEQ     RTS6                    ;BRANCH IF GOOD
 3857 015030  104001                            ERROR   +1              ;CPU ERROR
                                                                        ;REGISTER CONTENTS INCORRECT
 3858
 3859                                           ;
 3860                                           ;THIS TEST CHECKS AN UN-TESTED PLA TERM
 3861                                           ;
 3862 015032  010601            RTS6:   MOV     R6,R1                           ;SAVE STACK IN R1
 3863 015034  012705  015046                    MOV     #1$,R5                  ;MOVE EXPECTED RETURN ADDR TO R5
 3864 015040  010506                            MOV     R5,R6                   ;MOVE RETURN ADDR TO R6
 3865 015042  000206                            RTS     R6                      ;
 3866 015044  104001                            ERROR   +1              ;CPU ERROR
```

********** DOUBLE OPERAND TESTS **********

```
3867
3868 015046  021506                    1$:     CMP     (R5),R6         ;ERROR! RTS NOT EXECUTED
3869 015050  001401                            BEQ     RTSE            ;IS R6=31506?
3870 015052  104001                            ERROR   +1              ;IF IT IS THEN GO TO END OF TEST
3871                                                                   ;CPU ERROR
3872 015054  01010f                    RTSE:   MOV     R1,R6           ;ERROR! WRONG ADDR IN R6
3873                                                                   ;RESTORE STACK
3876 015056                            TSMMU0:
3877                                   ;       SETUP AND TEST KERNEL, SUPERVISOR AND USER STACKS
3878 015056  012737  040000  177776             MOV     #40000,@#177776 ;SET PS TO SUP MODE
3879 015064  012706  177777                    MOV     #177777,R6      ;INIT SUP STACK TO ALL ONES
3880 015070  022706  177777                    CMP     #177777,R6      ;ARE ALL BITS SET
3881 015074  001401                            BEQ     1$              ;YES GO ON
3882                                                                   ;NO GO TO ERROR
3883 015076  104001                            ERROR   +1              ;CPU ERROR
3884 015100  005006                    1$:     CLR     R6              ;SET SUP STACK TO ALL ZEROES
3885 015102  022706  000000                    CMP     #0,R6           ;ARE ALL BITS CLEARED
3886 015106  001401                            BEQ     2$              ;YES GO ON
3887                                                                   ;NO GO TO ERROR
3888 015110  104001                            ERROR   +1              ;CPU ERROR
3889 015112  012706  125252            2$:     MOV     #125252,R6      ;SET SUP STACK TO ALTERNATING PATTERN
3890 015116  022706  125252                    CMP     #125252,R6      ;IS SUP SP CORRECT
3891 015122  001401                            BEQ     3$              ;YES GO ON
3892                                                                   ;NO GO TO ERROR
3893 015124  104001                            ERROR   +1              ;CPU ERROR
3894 015126  012706  052525            3$:     MOV     #52525,R6       ;SET SUP STACK TO ALTERNATING PATTERN
3895 015132  022706  052525                    CMP     #52525,R6       ;IS SUP SP CORRECT
3896 015136  001401                            BEQ     4$              ;YES GO ON
3897                                                                   ;NO GO TO ERROR
3898 015140  104001                            ERROR   +1              ;CPU ERROR
3899 015142  012706  000700            4$:     MOV     #700,R6         ;SETUP SUP SP
3900 015146  012737  140000  177776             MOV     #140000,@#177776 ;SET PS TO USER MODE
3901 015154  012706  177777                    MOV     #177777,R6      ;INIT USER STACK TO ALL ONES
3902 015160  022706  177777                    CMP     #177777,R6      ;ARE ALL BITS SET
3903 015164  001401                            BEQ     5$              ;YES GO ON
3904                                                                   ;NO GO TO ERROR
3905 015166  104001                            ERROR   +1              ;CPU ERROR
3906 015170  005006                    5$:     CLR     R6              ;SET USER STACK TO ALL ZEROES
3907 015172  022706  000000                    CMP     #0,R6           ;ARE ALL BITS CLEARED
3908 015176  001401                            BEQ     6$              ;YES GO ON
3909                                                                   ;NO GO TO ERROR
3910 015200  104001                            ERROR   +1              ;CPU ERROR
3911 015202  012706  125252            6$:     MOV     #125252,R6      ;SET USER STACK TO ALTERNATING PATTERN
3912 015206  022706  125252                    CMP     #125252,R6      ;IS USER SP CORRECT
3913 015212  001401                            BEQ     7$              ;YES GO ON
3914                                                                   ;NO GO TO ERROR
3915 015214  104001                            ERROR   +1              ;CPU ERROR
3916 015216  012706  052525            7$:     MOV     #52525,R6       ;SET USER STACK TO ALTERNATING PATTERN
3917 015222  022706  052525                    CMP     #52525,R6       ;IS USER SP CORRECT
3918 015226  001401                            BEQ     8$              ;YES GO ON
3919                                                                   ;NO GO TO ERROR
3920 015230  104001                            ERROR   +1              ;CPU ERROR
3921 015232  012706  000600            8$:     MOV     #600,R6         ;SETUP USER SP
3922 015236  005037  177776                    CLR     @#177776        ;SET PS TO KER MODE
3923 015242  012706  001000                    MOV     #STBOT,R6       ;SETUP KERNEL SP
3924 015246  005037  000700                    CLR     @#700           ;CLEAR FIRST WORDS OF SUP, KER, AND USE STACKS
3925 015252  005037  000600                    CLR     @#600           ;
```

```
********** DOUBLE OPERAND TESTS *****************

3926 015276 005037 001000                  CLR     @#STBOT                 ;
3927 015302 004767 000054                  JSR     PC,CHECK                ; TEST KER, SUP, AND USE STACKS
3928 015306 012737 040000 177776 RET1:     MOV     #40000,@#177776         ;SET PSW TO SUP MODE
3929 015314 022706 000700                  CMP     #700,R6                 ;IS SUP SP CORRECT
3930 015320 001401                         BEQ     1$                      ;YES GO ON
3931                                                                       ;NO GO TO ERROR
3932 015322 104001                         ERROR   .+1                 ;CPU ERROR
3933 015324 012737 140000 177776 1$:       MOV     #140000,@#177776        ;SET PSW TO USE MODE
3934 015332 022706 000600                  CMP     #600,R6                 ;IS USE SP CORRECT
3935 015336 001401                         BEQ     2$                      ;YES GO ON
3936                                                                       ;NO GO TO ERROR
3937 015340 104001                         ERROR   .+1                 ;CPU ERROR
3938 015342 005037 177776          2$:     CLR     @#177776                ;SET PSW TO KER MODE
3939 015346 022706 001000                  CMP     #STBOT,R6               ;IS KER SP CORRECT
3940 015352 001401                         BEQ     3$                      ;YES GO ON
3941                                                                       ;NO GO TO ERROR
3942 015354 104001                         ERROR   .+1                 ;CPU ERROR
3943 015356                        3$:
3944 015356 000167 000206                  JMP     MTS0
3945                                       ;
3946                                       ;ROUTINE TO CHECK STACKS AFTER TWO RTS STATEMENTS
3947                                       ;
3948 015362 012737 040000 177776 CHECK:    MOV     #40000,@#177776         ;SET PSW TO SUP MODE
3949 015370 004767 000044                  JSR     PC,CHECK1               ; TEST SUP, KER, AND USE STACKS
3950 015374 022716 000000          RET2:   CMP     #0,(SP)                 ;IS SUP STACK CLEARED
3951 015400 001401                         BEQ     1$                      ;YES GO ON
3952                                                                       ;NO GO TO ERROR
3953 015402 104001                         ERROR   .+1                 ;CPU ERROR
3954 015404 012737 140000 177776 1$:       MOV     #140000,@#177776        ;SET PSW TO USE MODE
3955 015412 022716 000000                  CMP     #0,(SP)                 ;IS USE STACK CLEARED
3956 015416 001401                         BEQ     2$                      ;YES GO ON
3957                                                                       ;NO GO TO ERROR
3958 015420 104001                         ERROR   .+1                 ;CPU ERROR
3959 015422 005037 177776          2$:     CLR     @#177776                ;SET PSW TO KER MODE
3960 015426 022716 015266                  CMP     #RET1,(SP)              ;DOES KER STACK HAVE CORRECT DATA
3961 015432 001401                         BEQ     3$                      ;YES GO ON
3962                                                                       ;NO GO TO ERROR
3963 015434 104001                         ERROR   .+1                 ;CPU ERROR
3964 015436 000207                 3$:     RTS     PC                      ;RETURN
3965                                       ;
3966                                       ;ROUTINE TO CHECK STACKS AFTER ONE RTS
3967                                       ;
3968 015440 012737 140000 177776 CHECK1:   MOV     #140000,@#177776        ;SET PSW TO USE MODE
3969 015446 004767 000044                  JSR     PC,CHECK2               ; TEST KER, SUP, AND USE STACKS
3970 015452 022716 000000          RET3:   CMP     #0,(SP)                 ;IS USE STACK CLEARED
3971 015456 001401                         BEQ     1$                      ;YES GO ON
3972                                                                       ;NO GO TO ERROR
3973 015460 104001                         ERROR   .+1                 ;CPU ERROR
3974 015462 005037 177776          1$:     CLR     @#177776                ;SET PSW TO KER MODE
3975 015466 022716 015266                  CMP     #RET1,(SP)              ;IS KER STACK CORRECT
3976 015472 001401                         BEQ     2$                      ;YES GO ON
3977                                                                       ;NO GO TO ERROR
3978 015474 104001                         ERROR   .+1                 ;CPU ERROR
3979 015476 012737 040000 177776 2$:       MOV     #40000,@#177776         ;SET PSW TO SUP MODE
3980 015504 022716 015354                  CMP     #RET2,(SP)              ;IS SUP STACK CORRECT
3981 015510 001401                         BEQ     3$                      ;YES GO ON
3982                                                                       ;NO GO TO ERROR
```

```
********** DOUBLE OPERAND TESTS *****************

3983 015472  104001                        ERROR   +1              ;CPU ERROR
3984 015474  000207              3$:       RTS     PC                      ;RETURN
3985                             ;
3986                             ;ROUTINE TO CHECK STACKS AFTER ZERO RTS
3987                             ;
3988 015476  022716  015432      CHECK2:   CMP     #RET3,(SP)              ;IS USE STACK CORRECT
3989 015502  001401                        BEQ     1$                      ;YES GO ON
3990                                                                       ;NO GO TO ERROR
3991 015504  104001                        ERROR   +1              ;CPU ERROR
3992 015506  012737  040000  177776  1$:   MOV     #40000,@#177776         ;SET PSW TO SUP MODE
3993 015514  022716  015354              CMP     #RET2,(SP)              ;IS SUP STACK CORRECT
3994 015520  001401                        BEQ     2$                      ;YES GO ON
3995                                                                       ;NO GO TO ERROR
3996 015522  104001                        ERROR   +1              ;CPU ERROR
3997 015524  005037  177776      2$:       CLR     @#177776                ;SET PSW TO KER MODE
3998 015530  022716  015266              CMP     #RET1,(SP)              ;IS KER STACK CORRECT
3999 015534  001401                        BEQ     3$                      ;YES GO ON
4000                                                                       ;NO GO TO ERROR
4001 015536  104001                        ERROR   +1              ;CPU ERROR
4002 015540  012737  140000  177776  3$:   MOV     #140000,@#177776        ;SET PSW TO USE MODE
4003 015546  000207                        RTS     PC                      ;RETURN
4004                             ;
4005 015550                      MTSO:
4008 015550                      MMVCC:
4009
4010                             ;       TEST MOV CONDITION CODES   **0
4011 015550  000277                        SCC                             ;
4012 015552  000244                        CLZ                             ;CC=1011
4013 015554  012704  000000                MOV     #0,R4                   ;CC=0101, R4=0
4014 015560  100403                        BMI     1$                      ;ERROR IF N FLAG
4015 015562  102402                        BVS     1$                      ;ERROR IF V FLAG SET
4016 015564  103001                        BCC     1$                      ;ERROR IF C FLAG CLEAR
4017 015566  001401                        BEQ     2$                      ;SKIP IF Z FLAG SET
4018                                                                       ;CC SHOULD=0101
4019 015570  104001              1$:       ERROR   +1              ;CPU ERROR
4020 015572  000277              2$:       SCC
4021 015574  000251                        .WORD   251                     ;CC=0110
4022 015576  012704  100000                MOV     #100000,R4              ;R4=100000, CC=1000
4023 015602  001403                        BEQ     3$                      ;ERROR IF Z SET
4024 015604  102402                        BVS     3$                      ;ERROR IF V SET
4025 015606  103401                        BCS     3$                      ;ERROR IF C SET
4026 015610  100401                        BMI     4$                      ;EXIT IF N SET
4027 015612  104001              3$:       ERROR   +1              ;CPU ERROR
4028                                                                       ;CC SHOULD= 1000
4029 015614                      4$:
4030
4031
4034 015614                      MBTCC:
4035
4036                             ;       TEST BIT CONDITION CODES - **0-
4037 015614  005004                        CLR     R4
4038 015616  005104                        COM     R4                      ;R4=-1
4039 015620  000277                        SCC                             ;
4040 015622  000244                        CLZ                             ;CC=1011
4041 015624  032704  000000                BIT     #0,R4                   ;CC=0101
4042 015630  100403                        BMI     1$                      ;ERROR IF N FLAG
4043 015632  102402                        BVS     1$                      ;ERROR IF V FLAG SET
```

```
********** DOUBLE OPERAND TESTS *****************

4044 015634  103001                    BCC     1$                      ;ERROR IF C FLAG CLEAR
4045 015636  001401                    BEQ     2$                      ;SKIP IF Z FLAG SET
4046 015640  104001            1$:     ERROR   +1              ;CPU ERROR
4047                                                           ;CC SHOULD=0101
4048 015642  000277            2$:     SCC
4049 015644  000251                    .WORD   251                     ;CC=0110
4050 015646  032704  100000            BIT     #100000,R4              ;CC=1000
4051 015652  001403                    BEQ     3$                      ;ERROR IF Z SET
4052 015654  102402                    BVS     3$                      ;ERROR IF V SET
4053 015656  103401                    BCS     3$                      ;ERROR IF C SET
4054 015660  100401                    BMI     4$                      ;EXIT IF N SET
4055 015662  104001            3$:     ERROR   +1              ;CPU ERROR
4056                                                           ;CC SHOULD= 1000
4057 015664                    4$:
4058
4059
4062 015664                    MBCCC:
4063
4064                           ;       TEST BIC CONDITION CODES - **0-
4065 015664  005004                    CLR     R4
4066 015666  005104                    COM     R4                      ;R4= 1
4067 015670  000277                    SCC                             ;
4068 015672  000244                    CLZ                             ;CC=1011
4069 015674  042704  177777            BIC     #177777,R4              ;CC=0101
4070 015700  100403                    BMI     1$                      ;ERROR IF N FLAG
4071 015702  102402                    BVS     1$                      ;ERROR IF V FLAG SET
4072 015704  103001                    BCC     1$                      ;ERROR IF C FLAG CLEAR
4073 015706  001401                    BEQ     2$                      ;SKIP IF Z FLAG SET
4074 015710  104001            1$:     ERROR   +1              ;CPU ERROR
4075                                                           ;CC SHOULD=0101
4076 015712  005104            2$:     COM     R4                      ;R4=-1
4077 015714  000277                    SCC
4078 015716  000251                    .WORD   251                     ;CC=0110
4079 015720  042704  077777            BIC     #77777,R4               ;CC=1000
4080 015724  001403                    BEQ     3$                      ;ERROR IF Z SET
4081 015726  102402                    BVS     3$                      ;ERROR IF V SET
4082 015730  103401                    BCS     3$                      ;ERROR IF C SET
4083 015732  100401                    BMI     4$                      ;EXIT IF N SET
4084 015734  104001            3$:     ERROR   +1              ;CPU ERROR
4085                                                           ;CC SHOULD= 1000
4086 015736                    4$:
4087
4088
4091 015736                    MBSCC:
4092
4093                           ;       TEST BIS CONDITION CODES
4094 015736  005004                    CLR     R4                      ;R4=0
4095 015740  000277                    SCC
4096 015742  000246                    .WORD   246                     ;CC=1001
4097 015744  052704  000000            BIS     #0,R4                   ;R4=0, CC=0101
4098 015750  100403                    BMI     1$                      ;ERROR IF MINUS
4099 015752  102402                    BVS     1$                      ;ERROR IF V SET
4100 015754  103001                    BCC     1$                      ;ERROR IF C CLEAR
4101 015756  001401                    BEQ     2$                      ;BRANCH IF GOOD
4102 015760  104001            1$:     ERROR   +1              ;CPU ERROR
4103                                                           ;BIS CC FAILED
4104 015762  000277            2$:     SCC
```

```
********** DOUBLE OPERAND TESTS *****************

4105 015764  000241                    CLC                             ;CC=1110
4106 015766  052704  100076             BIS     #100076,R4              ;R4=100076, CC=1000
4107 015772  001403                     BEQ     3$                      ;ERROR IF Z SET
4108 015774  102402                     BVS     3$                      ;ERROR IF V SET
4109 015776  103401                     BCS     3$                      ;ERROR IF C SET
4110 016000  100401                     BMI     4$                      ;BRANCH IF GOOD
4111 016002  104001             3$:     ERROR   +1              ;CPU ERROR
4112                                                             ;BAD BIS CC
4113 016004                     4$:
4114
4115
4118 016004                     MDCCC:
4119
4120                            ;       TEST DEC, INC CONDITION CODES
4121 016004  012704  077777             MOV     #77777,R4               ;R4=77777
4122 016010  000257                     CCC
4123 016012  000261                     SEC                             ;CC=0001
4124 016014  005204                     INC     R4                      ;R4=100000, CC=0011
4125 016016  001403                     BEQ     1$                      ;ERROR IF ZERO
4126 016020  100002                     BPL     1$                      ;ERROR IF POSITIVE
4127 016022  102001                     BVC     1$                      ;ERROR IF V CLEAR
4128 016024  103401                     BCS     2$                      ;BRANCH IF GOOD
4129 016026  104001             1$:     ERROR   +1              ;CPU ERROR
4130                                                             ;INC FAILED
4131 016030  000257             2$:     CCC
4132 016032  005204                     INC     R4                      ;R4=100001, CC=1000
4133 016034  103413                     BCS     3$                      ;ERROR IF C SET
4134 016036  102412                     BVS     3$                      ;ERROR IF V SET
4135 016040  005304                     DEC     R4                      ;R4=100000, CC=1000
4136 016042  102410                     BVS     3$                      ;ERROR IF V SET
4137 016044  103407                     BCS     3$                      ;ERROR IF C SET
4138 016046  000277                     SCC                             ;
4139 016050  000252                     .WORD   252                     ;CC=0101
4140 016052  005304                     DEC     R4                      ;R4=77777, CC=1011
4141 016054  001403                     BEQ     3$                      ;ERROR IF Z SET
4142 016056  102002                     BVC     3$                      ;ERROR IF V CLEAR
4143 016060  103001                     BCC     3$                      ;ERROR IF C CLEAR
4144 016062  100001                     BPL     4$                      ;BRANCH IF GOOD
4145 016064  104001             3$:     ERROR   +1              ;CPU ERROR
4146                                                             ;BAD CC
4147 016066                     4$:
4148
4149
4159 016066                     MCTSCC:
4160
4161                            ;       TEST CLR, TST, SWAB CONDITION CODES
4162                            ;;**************.**************************************************
                                ;0100 - **00    **00
4163 016066  000277                     SCC
4164 016070  000244                     CLZ                             ;CC=1011
4165 016072  005004                     CLR     R4                      ;R4=0, CC=0100
4166 016074  100403                     BMI     1$                      ;ERROR IF MINUS
4167 016076  102402                     BVS     1$                              ;ERROR IF V SET
4168 016100  103401                     BCS     1$                      ;ERROR IF C SET
4169 016102  001401                     BEQ     2$                      ;BRANCH IF GOOD
4170 016104  104001             1$:     ERROR   +1              ;CPU ERROR
4171                                                             ;BAD CC ON CLR
```

```
********** DOUBLE OPERAND TESTS *****************

4172 016106  005104                  2$:     COM     R4                      ;R4=-1
4173 016110  000277                          SCC                             ;CC=1111
4174 016112  005704                          TST     R4                      ;CC=1000
4175 016114  001403                          BEQ     3$                      ;ERROR IF Z SET
4176 016116  102402                          BVS     3$                      ;ERROR IF V SET
4177 016120  103401                          BCS     3$                      ;ERROR IF C SET
4178 016122  100401                          BMI     4$                      ;BRANCH IF GOOD
4179 016124  104001                  3$:     ERROR   +1              ;CPU ERROR
4180                                                                 ;BAD TST CC
4181 016126  000277                  4$:     SCC                             ;CC=1111
4182 016130  000304                          SWAB    R4                      ;CC=1000
4183 016132  102402                          BVS     5$                      ;ERROR IF V SET
4184 016134  103401                          BCS     5$                      ;ERROR IF C SET
4185 016136  100401                          BMI     6$                      ;BRANCH IF GOOD
4186 016140  104001                  5$:     ERROR   +1              ;CPU ERROR
4187                                                                 ;BAD SWAB CC
4188 016142                          6$:
4189
4190
4193 016142                          MADCC:
4194
4195                                 ;       TEST ADD CONDITION CODES
4196 016142  012704  077777                  MOV     #77777,R4               ;R4=77777
4197 016146  012701  000001                  MOV     #1,R1                   ;R1=1
4198 016152  000257                          CCC                             ;CC=0000
4199 016154  060401                          ADD     R4,R1                   ;77777 + 1 = 100000 IN R1
4200 016156  102003                          BVC     1$                      ;ERROR IF V CLEAR
4201 016160  103402                          BCS     1$                      ;ERROR IF CARRY
4202 016162  001401                          BEQ     1$                      ;ERROR IF Z SET
4203 016164  100401                          BMI     2$                      ;BRANCH IF GOOD
4204 016166  104001                  1$:     ERROR   +1              ;CPU ERROR
4205                                                                 ;CC SHOULD =1010
4206 016170  005204                  2$:     INC     R4                      ;R4=100000
4207 016172  060401                          ADD     R4,R1                   ;100000 + 100000 = 0 IN R1
4208 016174  102002                          BVC     3$                      ;ERROR IF V CLEAR
4209 016176  103001                          BCC     3$                      ;ERROR IF CARRY  CLEAR
4210 016200  001401                          BEQ     4$                      ;BRANCH IF GOOD
4211 016202  104001                  3$:     ERROR   +1              ;CPU ERROR
4212                                                                 ;CC SHOULD = 0111
4213 016204  060401                  4$:     ADD     R4,R1                   ;0 + 100000 = 100000
4214 016206  102402                          BVS     5$                      ;ERROR IF V SET
4215 016210  103401                          BCS     5$                      ;ERROR IF  SET
4216 016212  100401                          BMI     6$                      ;BRANCH IF GOOD
4217 016214  104001                  5$:     ERROR   +1              ;CPU ERROR
4218                                                                 ;CC SHOULD = 1000
4219 016216                          6$:
4220
4221
4224 016216                          MACCC:
4225
4226                                 ;       TEST ADC CONDITION CODES
4227 016216  012704  177777                  MOV     #177777,R4              ;R4=177777
4228 016222  000277                          SCC                             ;CC=1111
4229 016224  005504                          ADC     R4                      ;R4=0 CC=0101
4230 016226  100403                          BMI     1$                      ;ERROR IF MINUS
4231 016230  102402                          BVS     1$                      ;ERROR IF V SET
4232 016232  103001                          BCC     1$                      ;ERROR IF C SET
```

F7


```
********** DOUBLE OPERAND TESTS *****************

4233 016234  001401                        BEQ     2$                      ;BRANCH IF GOOD
4234 016236  104001              1$:       ERROR   +1              ;CPU ERROR
4235                                                             ;BAD ADC
4236 016240  012704  077777      2$:       MOV     #077777,R4              ;R4=077777
4237 016244  000277                        SCC                             ;
4238 016246  000242                        CLV                             ;CC=1101
4239 016250  005504                        ADC     R4                      ;R4=100000 CC=1010
4240 016252  100003                        BPL     3$                      ;ERROR IF PLUS
4241 016254  103402                        BCS     3$                      ;ERROR IF C SET
4242 016256  001401                        BEQ     3$                      ;ERROR IF ZERO
4243 016260  102401                        BVS     4$                      ;BRANCH IF GOOD
4244 016262  104001              3$:       ERROR   +1              ;CPU ERROR
4245                                                             ;BAD ADC
4246 016264  000277              4$:       SCC                             ;CC=1111
4247 016266  005504                        ADC     R4                      ;R4=100000 CC=1000
4248 016270  102402                        BVS     5$                      ;ERROR IF V SET
4249 016272  103401                        BCS     5$                      ;ERROR IF C SET
4250 016274  100401                        BMI     6$                      ;BRANCH IF GOOD
4251 016276  104001              5$:       ERROR   +1              ;CPU ERROR
4252                                                             ;BAD ADC CC SHOULD= 1000
4253 016300                      6$:
4254
4255
4258 016300                      MNCCCC:
4259
4260                             ;         TEST NEG, CMP, COM CONDITION CODES
4261 016300  012704  077777                MOV     #077777,R4              ;R4=77777
4262 016304  000257                        CCC                             ;CC=0000
4263 016310  005404                        NEG     R4                      ;R4=100001      CC=1001
4264 016312  102403                        BVS     1$                      ;ERROR IF V SET
4265 016314  103002                        BCC     1$                      ;ERROR IF C CLEAR
4266 016316  001401                        BEQ     1$                      ;ERROR IF Z SET
4267 016320  100401                        BMI     2$                      ;BRANCH IF GOOD
4268 016322  104001              1$:       ERROR   +1              ;CPU ERROR
4269                                                             ;BAD NEGATE
4270 016324  005004              2$:       CLR     R4                      ;R4=0
4271 016326  000257                        CCC                             ;CC=0000
4272 016330  005404                        NEG     R4                      ;CC=0101
4273 016332  100403                        BMI     3$                      ;ERROR IF N SET
4274 016334  103402                        BCS     3$                      ;ERROR IF C SET
4275 016336  102401                        BVS     3$                      ;ERROR IF V SET
4276 016340  001401                        BEQ     4$                      ;BRANCH IF GOOD
4277 016342  104001              3$:       ERROR   +1              ;CPU ERROR
4278                                                             ;BAD NEG
4279 016344  012704  077777      4$:       MOV     #77777,R4               ;R4=77777
4280 016350  012701  170000                MOV     #170000,R1              ;R1=170000
4281 016354  000257                        CCC                             ;CC=0000
4282 016356  020401                        CMP     R4,R1                   ;77777 - 170000 = 107777 CC= 1011
4283 016360  102003                        BVC     5$                      ;ERROR IF V CLEAR
4284 016362  103002                        BCC     5$                      ;ERROR IF C CLEAR
4285 016364  001401                        BEQ     5$                      ;ERROR IF ZERO
4286 016366  100401                        BMI     6$                      ;BRANCH IF GOOD
4287 016370  104001              5$:       ERROR   +1              ;CPU ERROR
4288                                                             ;BAD CMP
4289 016372  000257              6$:       CCC
4290 016374  005101                        COM     R1                      ;R1=7777
4291 016376  100403                        BMI     7$                      ;ERROR IF MINUS
```

```
********** DOUBLE OPERAND TESTS *****************

4292 016376  001402                        BEQ     7$                      ;ERROR IF ZERO
4293 016400  103001                        BCC     7$                      ;ERROR IF CARRY
4294 016402  102001                        BVC     8$                      ;BRANCH IF
4295 016404  104001                7$:     ERROR   +1              ;CPU ERROR
4296                                                               ;BAD COM
4297 016406  000277                8$:     SCC
4298 016410  005101                        COM     R1
4299 016412  100401                        BMI     10$                     ;BRANCH IF GOOD
4300 016414  104001                9$:     ERROR   +1              ;CPU ERROR
4301                                                               ;BAD COM
4302 016416                        10$:
4303
4304
4307 016416                        MSBCC:
4308
4309                               ;       TEST SUB CONDITION CODES
4310 016416  012704  077775                MOV     #77775,R4               ;R4=77775
4311 016422  000257                        CCC                             ;CC=0000
4312 016424  162704  137757                SUB     #137757,R4              ;77775 - 137757
4313                                                                       ;TRY TO CAUSE AN ARITHMETIC OVERFLOW
4314 016430  102003                        BVC     .                       ;ERROR IF V CLEAR
4315 016432  100002                        BPL     .                       ;ERROR IF RESULT IS POSITIVE
4316 016434  001401                        BEQ     1$                      ;ERROR IF  Z SET
4317 016436  103401                        BCS     2$                      ;BRANCH IF GOOD
4318 016440  104001                1$:     ERROR   +1              ;CPU ERROR
4319                                                               ;BAD SUBTRACT
4320 016442  012704  000005        2$:     MOV     #5,R4                   ;R4=5
4321 016446  000257                        CCC                             ;CC=0000
4322 016450  162704  000012                SUB     #12,R4                  ;5-12=-5 AND SETS CARRY
4323 016454  103003                        BCC     3$                      ;ERROR IF CARRY CLEAR
4324 016456  102402                        BVS     3$                      ;ERROR IF OVERFLOW
4325 016460  001401                        BEQ     3$                      ;ERROR IF ZERO
4326 016462  100401                        BMI     4$                      ;BRANCH IF GOOD
4327 016464  104001                3$:     ERROR   +1              ;CPU ERROR
4328                                                               ;SUBTRACT FAILED
4329 016466                        4$:
4330
4331
4334 016466                        MSBCCC:
4335
4336                               ;       TEST SBC CONDITION CODES
4337 016466  012704  100000                MOV     #100000,R4              ;R4=100000
4338 016472  000257                        CCC                             ;C=0000
4339 016474  005604                        SBC     R4                      ;TRY TO SET V
4340 016476  100006                        BPL     1$                      ;ERROR IF N CLEAR
4341 016500  102405                        BVS     1$                      ;ERROR IF V SET (HAVENT SET C YET)
4342 016502  000261                        SEC                             ;CC SHOULD = 1001
4343 016504  005604                        SBC     R4                      ;TRY AGAIN TO SET V
4344 016506  102002                        BVC     1$                      ;ERROR IF V CLEAR
4345 016510  103401                        BCS     1$                      ;ERROR IF C SET
4346 016512  100001                        BPL     2$                      ;BRANCH IF GOOD
4347 016514  104001                1$:     ERROR   +1              ;CPU ERROR
4348                                                               ;SBC FAILED
4349 016516  005004                2$:     CLR     R4                      ;R4=0
4350 016520  000277                        SCC
4351 016522  000241                        CLC                             ;CC=1110
4352 016524  005604                        SBC     R4                      ;TRY TO CAUSE C FLAG FAILURE
```

```
********** DOUBLE OPERAND TESTS *****************

4353 016526  103410                    BCS     3$                      ;ERROR IF C SET
4354 016530  102407                    BVS     3$                      ;ERROR IF V SET
4355 016532  001006                    BNE     3$                      ;ERROR IF  NOT ZERO
4356 016534  000261                    SEC                             ;SET CARRY
4357 016536  005604                    SBC     R4                      ;NOW, 0 - CARRY = 177777
4358 016540  103003                    BCC     3$                      ;ERROR IF CARRY CLEAR
4359 016542  102402                    BVS     3$                      ;ERROR IF V SET
4360 016544  001401                    BEQ     3$                      ;ERROR IF ZERO
4361 016546  100401                    BMI     4$                      ;BRANCH IF GOOD
4362 016550  104001            3$:     ERROR   +1              ;CPU ERROR
4363                                                            ;SBC FAILED
4364 016552                     4$:
4365
4366
4369 016552                     MRLCC:
4370
4371                            ;       TEST ROL CONDITION CODES
4372 016552  012704  060000              MOV     #60000,R4               ;R4= 0110000000000000
4373 016556  000257                    CCC                             ;CC=0000
4374 016560  006104                    ROL     R4                      ;R4= 1100000000000000
4375 016562  103402                    BCS     1$                      ;ERROR IF CARRY
4376 016564  102001                    BVC     1$                      ;ERROR IF V CLEAR
4377 016566  100401                    BMI     2$                      ;BRANCH IF GOOD
4378 016570  104001            1$:     ERROR   +1              ;CPU ERROR
4379                                                            ;ROL  FAILED
4380 016572  006104            2$:     ROL     R4                      ;R4= 1000000000000000
4381 016574  103002                    BCC     3$                      ;ERROR IF CARRY CLEAR
4382 016576  102401                    BVS     3$                      ;ERROR IF V SET
4383 016600  100401                    BMI     4$                      ;BRANCH IF GOOD
4384 016602  104001            3$:     ERROR   +1              ;CPU ERROR
4385                                                            ;BAD ROL
4386 016604  006104            4$:     ROL     R4                      ;R4 = 0000000000000001
4387 016606  102003                    BVC     5$                      ;ERROR IF V CLEAR
4388 016610  103002                    BCC     5$                      ;ERROR IF C CLEAR
4389 016612  100401                    BMI     5$                      ;ERROR IF MINUS
4390 016614  001001                    BNE     6$                      ;BRANCH IF GOOD
4391 016616  104001            5$:     ERROR   +1              ;CPU ERROR
4392                                                            ;BAD ROL
4393 016620  006104            6$:     ROL     R4                      ;R4=0000000000000011
4394 016622  102402                    BVS     7$                      ;ERROR IF V SET
4395 016624  103401                    BCS     7$                      ;ERROR IF C SET
4396 016626  100001                    BPL     8$                      ;BRANCH IF GOOD
4397 016630  104001            7$:     ERROR   +1              ;CPU ERROR
4398                                                            ;BAD ROL
4399 016632                     8$:
4400
4401
4404 016632                     MRRCC:
4405
4406                            ;       TEST ROR CONDITION CODES
4407 016632  012704  000003              MOV     #3,R4                   ;R4= 0000000000000011
4408 016636  000257                    CCC                             ;CC= 0000
4409 016640  006004                    ROR     R4                      ;R4= 0000000000000001
4410 016642  103002                    BCC     1$                      ;ERROR IF NO CARRY
4411 016644  102001                    BVC     1$                      ;ERROR IF V CLEAR
4412 016646  100001                    BPL     2$                      ;BRANCH IF GOOD
4413 016650  104001            1$:     ERROR   +1              ;CPU ERROR
```

```
********** DOUBLE OPERAND TESTS *****************

                                                                ;ROR  FAILED
4414
4415 016652  006004                2$:     ROR     R4           ;R4= 1000000000000000
4416 016654  103002                        BCC     3$           ;ERROR IF CARRY CLEAR
4417 016656  102401                        BVS     3$           ;ERROR IF V SET
4418 016660  100401                        BMI     4$           ;BRANCH IF GOOD
4419 016662  104001                3$:     ERROR   +1           ;CPU ERROR
4420                                                            ;BAD ROR
4421 016664  006004                4$:     ROR     R4           ;R4 = 1100000000000000
4422 016666  102002                        BVC     5$           ;ERROR IF V
4423 016670  103401                        BCS     5$           ;ERROR IF C SET
4424 016672  100401                        BMI     6$           ;BRANCH IF GOOD
4425 016674  104001                5$:     ERROR   +1           ;CPU ERROR
4426                                                            ;BAD ROR
4427 016676  006004                6$:     ROR     R4           ;R4= 0110000000000000
4428 016700  102402                        BVS     7$           ;ERROR IF V SET
4429 016702  103401                        BCS     7$           ;ERROR IF C SET
4430 016704  100001                        BPL     8$           ;BRANCH IF GOOD
4431 016706  104001                7$:     ERROR   +1           ;CPU ERROR
4432                                                            ;BAD ROR
4433 016710                        8$:
4434
4435
4438 016710                        XCBIT:
4447
4448                               ;       TEST C BIT WITH ROR/ROL
4449                               ;;*****************************************************************
                                   ;THIS TEST IS TO CHECK FOR A SLOW C BIT PATH INTERNAL TO THE J11 DATA CHIP
                                   ;PROBLEM IS ONLY EXHIBITED ON EARLY MASK SETS (1590 AND 1593)
4450 016710  012701  052525                MOV     #52525,R1    ;INIT R1 WITH DATA
4451 016714  000241                        CLC                  ;CLEAR THE C BIT
4452 016716  006001                        ROR     R1           ;R1=025252, C BIT =1
4453 016720  006001                        ROR     R1           ;R1=112525, C BIT =0
4454 016722  006001                        ROR     R1           ;R1=045252, C BIT =1
4455 016724  103401                        BCS     1$           ;BRANCH IF CARRY BIT SET
4456 016726  104001                        ERROR   +1           ;CPU ERROR
4457 016730  022701  045252        1$:     CMP     #45252,R1    ;IS DATA IN R1 = TO EXPECTED DATA?
4458 016734  001401                        BEQ     2$           ;BRANCH IF YES
4459 016736  104001                        ERROR   +1           ;CPU ERROR
4460 016740  012701  125252        2$:     MOV     #125252,R1   ;SET UP R1
4461 016744  000241                        CLC                  ;CLEAR THE CARRY BIT
4462 016746  006101                        ROL     R1           ;R1=052524, C BIT =1
4463 016750  006101                        ROL     R1           ;R1=125251, C BIT =0
4464 016752  006101                        ROL     R1           ;R1=052522, C BIT =1
4465 016754  103401                        BCS     3$           ;BRANCH IF CARRY SET
4466 016756  104001                        ERROR   +1           ;CPU ERROR
4467 016760  022701  052522        3$:     CMP     #052522, R1  ;IS DATA IN R1 = TO EXPECTED DATA?
4468 016764  001401                        BEQ     4$
4469 016766  104001                        ERROR   +1           ;CPU ERROR
4470 016770                        4$:
4472
4474 016770                        MALCC:
4475
4476                               ;       TEST ASL CONDITION CODES
4477 016770  012704  060000                MOV     #60000,R4    ;R4= 0110000000000000
4478 016774  000257                        CCC                  ;CC=0000
4479 016776  006304                        ASL     R4           ;C=0 R4= 1100000...  0000
4480 017000  103402                        BCS     1$           ;ERROR IF CARRY
```

```
********** DOUBLE OPERAND TESTS *****************

4481 017002  102001                          BVC     1$              ;ERROR IF V CLEAR
4482 017004  100401                          BMI     2$              ;BRANCH IF GOOD
4483 017006  104001                  1$:     ERROR   +1          ;CPU ERROR
4484                                                             ;ASL  FAILED
4485 017010  006304                  2$:     ASL     R4              ;C=1 R4= 1000000000000000
4486 017012  103002                          BCC     3$              ;ERROR IF CARRY CLEAR
4487 017014  102401                          BVS     3$              ;ERROR IF V SET
4488 017016  100401                          BMI     4$              ;BRANCH IF GOOD
4489 017020  104001                  3$:     ERROR   +1          ;CPU ERROR
4490                                                             ;BAD ASL
4491 017022  006304                  4$:     ASL     R4              ;C=1 R4= 0000000000000000
4492 017024  102003                          BVC     5$              ;ERROR IF V CLEAR
4493 017026  103002                          BCC     5$              ;ERROR IF C CLEAR
4494 017030  100401                          BMI     5$              ;ERROR IF MINUS
4495 017032  001401                          BEQ     6$              ;BRANCH IF GOOD
4496 017034  104001                  5$:     ERROR   +1          ;CPU ERROR
4497                                                             ;BAD ASL
4498 017036  006304                  6$:     ASL     R4              ;C=0 R4= 0000000000000000
4499 017040  102402                          BVS     7$              ;ERROR IF V SET
4500 017042  103401                          BCS     7$              ;ERROR IF C SET
4501 017044  100001                          BPL     8$              ;BRANCH IF GOOD
4502 017046  104001                  7$:     ERROR   +1          ;CPU ERROR
4503                                                             ;BAD ASL
4504 017050                          8$:
4505
4508 017050                          MARCC:
4509
4510                                 ;       TEST ASR CONDITION CODES
4511 017050  012704  000341                  MOV     #341,R4         ;R4= 0000000011100001
4512 017054  000257                          CCC                     ;CC=0000
4513 017056  006204                          ASR     R4              ;R4= 0000000001110000
4514 017060  103002                          BCC     1$              ;ERROR IF NO CARRY
4515 017062  102001                          BVC     1$              ;ERROR IF V CLEAR
4516 017064  100001                          BPL     2$              ;BRANCH IF GOOD
4517 017066  104001                  1$:     ERROR   +1          ;CPU ERROR
4518                                                             ;ASR  FAILED
4519 017070  052704  100001          2$:     BIS     #100001,R4      ;R4= 1000000001110001
4520 017074  006204                          ASR     R4              ;R4= 1100000000111000
4521 017076  103002                          BCC     3$              ;ERROR IF CARRY CLEAR
4522 017100  102401                          BVS     3$              ;ERROR IF V SET
4523 017102  100401                          BMI     4$              ;BRANCH IF GOOD
4524 017104  104001                  3$:     ERROR   +1          ;CPU ERROR
4525                                                             ;BAD ASR
4526 017106  006204                  4$:     ASR     R4              ;R4= 1110000000011100
4527 017110  102002                          BVC     5$              ;ERROR IF V
4528 017112  103401                          BCS     5$              ;ERROR IF C SET
4529 017114  100401                          BMI     6$              ;BRANCH IF GOOD
4530 017116  104001                  5$:     ERROR   +1          ;CPU ERROR
4531                                                             ;BAD ASR
4532 017120  006204                  6$:     ASR     R4              ;R4= 1111000000001110
4533 017122  102005                          BVC     7$              ;ERROR IF V CLEAR
4534 017124  103404                          BCS     7$              ;ERROR IF C SET
4535 017126  100003                          BPL     7$              ;ERROR IF PLUS
4536 017130  022704  170016                  CMP     #170016,R4      ;SEE IF EXPECTED RESULT
4537 017134  001401                          BEQ     8$              ;BRANCH IF GOOD
4538 017136  104001                  7$:     ERROR   +1          ;CPU ERROR
4539                                                             ;BAD ASR
```

K7


```
********** DOUBLE OPERAND TESTS ********.*********

4540 017140                            8$:
4541
4542
4545 017140                            MSXTCC:
4546
4547                                   ;       TEST SXT CONDITION CODES /  *0-
4548 017140  012704  123456                    MOV     #123456,R4                      ;R4=123456
4549 017144  010401                            MOV     R4,R1                   ;SAVE  CONTENTS
4550 017146  000257                            CCC                             ;CC=0000
4551 017150  006704                            SXT     R4                      ;R4=0 CC=0100
4552 017152  103403                            BCS     1$                      ;ERROR IF CARRY
4553 017154  100402                            BMI     1$                      ;ERROR IF MINUS
4554 017156  102401                            BVS     1$                      ;ERROR IF OVERFLOW
4555 017160  001401                            BEQ     2$                      ;BRANCH IF GOOD
4556 017162  104001                    1$:     ERROR   +1              ;CPU ERROR
4557                                                                   ;BAD SXT
4558 017164  010104                    2$:     MOV     R1,R4                   ;RESTORE R4
4559 017166  000277                            SCC                             ;CC=1111
4560 017170  006704                            SXT     R4                      ;R4=-1 CC=1001
4561 017172  001405                            BEQ     3$                      ;ERROR IF ZERO
4562 017174  100004                            BPL     3$                      ;ERROR IF PLUS
4563 017176  103003                            BCC     3$                      ;ERROR IF NO CARRY
4564 017200  102402                            BVS     3$                      ;ERROR IF OVERFLOW
4565 017202  005104                            COM     R4                      ;R4=0
4566 017204  001401                            BEQ     4$                      ;BRANCH IF GOOD
4567 017206  104001                    3$:     ERROR   +1              ;CPU ERROR
4568                                                                   ;BAD SXT
4569 017210                            4$:
4570
4571
4574 017210                            MXRCC:
4575
4576                                   ;       TEST XOR CONDITION CODES / **0-
4577 017210  012704  123456                    MOV     #123456,R4                      ;R4=123456
4578 017214  012701  052525                    MOV     #52525,R1                       ;R1=52525
4579 017220  000257                            CCC                             ;CC=0000
4580 017222  074104                            XOR     R1,R4                   ;*TI* R4=171173
4581 017224  102403                            BVS     1$                      ;;ERROR IF OVERFLOW
4582 017226  001402                            BEQ     1$                      ;ERROR IF ZERO
4583 017230  103401                            BCS     1$                      ;ERROR IF CARRY
4584 017232  100401                            BMI     2$                      ;BRANCH IF GOOD
4585 017234  104001                    1$:     ERROR   +1              ;CPU ERROR
4586                                                                   ;BAD XOR
4587 017236  012701  125252            2$:     MOV     #125252,R1                      ;R1=125252
4588 017242  000277                            SCC                             ;CC=1111
4589 017244  074104                            XOR     R1,R4                   ;R4=054321
4590 017246  100403                            BMI     3$                      ;ERROR IF MINUS
4591 017250  001402                            BEQ     3$                      ;ERROR IF ZERO
4592 017252  103001                            BCC     3$                      ;ERROR IF CARRY CLEAR
4593 017254  102001                            BVC     4$                      ;BRANCH IF GOOD
4594 017256  104001                    3$:     ERROR   +1              ;CPU ERROR
4595                                                                   ;BAD XOR
4596 017260  074404                    4$:     XOR     R4,R4                   ;R4=0
4597 017262  102406                            BVS     5$                      ;ERROR IF OVREFLOW
4598 017264  100405                            BMI     5$                      ;ERROR IF MINUS
4599 017266  103004                            BCC     5$                      ;ERROR IF NO CARRY
4600 017270  001003                            BNE     5$                      ;ERROR IF NOT ZERO
```

```
********** DOUBLE OPERAND TESTS *****************

4601 017272  022704  000000                CMP     #0,R4                   ;SEE IF EXPECTED RESULT
4602 017276  001401                        BEQ     6$                      ;BRANCH IF GOOD
4603 017300  104001                5$:     ERROR   ·1              ;CPU ERROR
4604                                                               ;BAD XOR
4605 017302                        6$:
4606
4607
4608                               ;
4620 017302                        MSXT:
4621
4622                               ;       TEST SXT (SIGN EXTEND INSTRUCTION)
4623                               ;;*****************************************************************
                                   ;AN ADDITIONAL TEST IS INCLUDED TO CHECK FOR A SLOW N BIT PATH
                                   ;ON A TRANSITION FROM ZERO TO ONE INTERNAL TO THE J11 DATA CHIP
                                   ;THE PROBLEM IS ONLY EXHIBITED ON EARLY MASK SETS (1590 OR 1593)
4624 017302  005004                        CLR     R4                      ;TRASH R4
4625 017304  000257                        CCC                             ;CC=0000
4626 017306  000271                        .WORD   271                     ;CC=1001
4627 017310  006704                        SXT     R4                      ;*TEST INSTRUCTION
4628 017312  102405                        BVS     1$                      ;BRANCH IF OVERFLOW IS NOT CLEARED
4629 017314  100004                        BPL     1$                      ;BRANCH IF N BIT EFFECTED
4630 017316  001403                        BEQ     1$                      ;BRANCH IF Z BIT EFFECTED
4631 017320  103002                        BCC     1$                      ;BRANCH IF C BIT EFFECTED
4632 017322  005204                        INC     R4
4633 017324  001401                        BEQ     2$                      ;BRANCH IF R4 =0
4634 017326  104001                1$:     ERROR   ·1              ;CPU ERROR
4635                                                       ;CC SHOULD HAVE = 1101
4636 017330  000277                2$:     SCC
4637 017332  000250                        CLN                             ;CC=0111
4638 017334  005004                        CLR     R4
4639 017336  012714  000055                MOV     #55,(R4)                ;TRASH R4
4640 017342  006714                        SXT     (R4)                    ;*TEST INSTRUCTION
4641 017344  001005                        BNE     3$                      ;BRANCH IF BIT EFFECTED
4642 017346  102404                        BVS     3$                      ;BRANCH IF OVERFLOW
4643 017350  103403                        BCS     3$                      ;
4644 017352  100402                        BMI     3$                      ;BRANCH IF N IS SET
4645 017354  005714                        TST     (R4)                    ;VERIFY INSTRUCTION WORKED
4646 017356  001401                        BEQ     4$                      ;BRANCH IF R4=0
4647 017360  104001                3$:     ERROR   ·1              ;CPU ERROR
4648                                                       ;SXT FAILED
4649
4650                               ;       NOW TEST FOR SLOW N BIT IN J11 DATA CHIP
4651
4652 017362  012700  177777        4$:     MOV     #-1,R0                  ;R0=177777, N BIT = 1
4653 017366  005004                        CLR     R4                      ;CLEAT THE N BIT
4654 017370  006700                        SXT     R0                      ;****TEST INSTRUCTION***
4655                                                                       ;TEST N BIT TRANSITION 1 TO 0
4656 017372  005700                        TST     R0                      ;R0 SHOULD = 0
4657 017374  001401                        BEQ     5$                      ;BRANCH IF OK
4658 017376  104001                        ERROR   ·1              ;CPU ERROR
4659 017400  005000                5$:     CLR     R0                      ;CLEAR R0, N BIT = 0
4660 017402  012704  177777                MOV     #-1,R4                  ;SET N BIT
4661 017406  006700                        SXT     R0                      ;***TEST INSTRUCTION***
4662                                                                       ;TEST N BIT TRANSITION 0 TO 1
4663 017410  022700  177777                CMP     # 1,R0                  ;R0 SHOULD = 177777
4664 017414  001401                        BEQ     6$                      ;BRANCH IF OK
4665 017416  104001                        ERROR   ·1              ;CPU ERROR
```

********** DOUBLE OPERAND TESTS **********

```
4666 017420                          6$:
4668                                 ;
4670 017420                          MXOR:
4671
4672                                 ;       TEST XOR
4673 017420  012701  007643                     MOV     #7643,R1            ;SETUP DATA
4674 017424  012704  133333                     MOV     #133333,R4          ;SETUP DATA
4675 017430  000277                             SCC
4676 017432  074401                             XOR     R4,R1               ;*TEST INSTRUCTION
4677 017434  100006                             BPL     1$                  ;BRANCH IF PLUS TO ERROR
4678 017436  001405                             BEQ     1$                  ;ERROR IF ZERO
4679 017440  103004                             BCC     1$                  ;ERROR IF CARRY CLEAR
4680 017442  102403                             BVS     1$                  ;ERROR IF V SET
4681 017444  020127  134570                     CMP     R1,#134570          ;VERIFY CORRECT RESULT
4682 017450  001401                             BEQ     2$                  ;BRANCH IF GOOD
4683 017452  104001                 1$:         ERROR   +1              ;CPU ERROR
4684                                                                     ;BAD XOR
4685 017454  010102                 2$:         MOV     R1,R2
4686 017456  000257                             CCC
4687 017460  074402                             XOR     R4,R2               ;*TEST INSTRUCTION
4688 017462  100405                             BMI     3$                  ;ERROR IF MINUS
4689 017464  102404                             BVS     3$                  ;ERROR IF OVERFLOW
4690 017466  103403                             BCS     3$                  ;ERROR IF CARRY
4691 017470  020227  007643                     CMP     R2,#7643
4692 017474  001401                             BEQ     4$                  ;BRANCH IF GOOD
4693 017476  104001                 3$:         ERROR   +1              ;CPU ERROR
4694                                                                     ;BAD XOR
4695 017500                          4$:
4696
4698                                 ;
4700 017500                          MSOB:
4701
4702                                 ;       TEST SOB
4703 017500  012704  000555                     MOV     #555,R4             ;SETUP TEST COUNTER
4704 017504  000277                             SCC                         ;CC=17
4705 017506  103015                 1$:         BCC     2$                  ;ERROR IF CARRY CLEAR
4706 017510  102014                             BVC     2$                  ;ERROR IF NO OVERFLOW
4707 017512  100013                             BPL     2$                  ;ERROR IF PLUS
4708 017514  001012                             BNE     2$                  ;ERROR IF ZERO
4709 017516  077405                             SOB     R4,1$               ;*TEST INSTRUCTION
4710 017520  103005                             BCC     3$                  ;ERROR IF CARRY CLEAR
4711 017522  102004                             BVC     3$                  ;ERROR IF NO OVERFLOW
4712 017524  100003                             BPL     3$                  ;ERROR IF PLUS
4713 017526  001002                             BNE     3$                  ;ERROR IF ZERO
4714 017530  000167  000010                     JMP     4$
4715 017534  104001                 3$:         ERROR   +1                  ;CPU ERROR
4716                                                                 ;CC EFFECTED DURING TEST
4717 017536  000167  000002                     JMP     4$
4718 017542  104001                 2$:         ERROR   +1                  ;CPU ERROR
4719                                                                 ;CC EFFECTED AFTER TEST
4720 017544  020427  000000         4$:         CMP     R4,#0               ;IS R4 CORRECT
4721 017550  001401                             BEQ     5$                  ;YES GO ON
4722 017552  104001                             ERROR   +1                  ;CPU ERROR
4723                                                                 ;NO GO TO ERROR
4724 017554                          5$:
4725
4727
```

```
********** DOUBLE OPERAND TESTS *****************

4729 017554                          MMARK:
4730
4731                            ;       MARK INSTRUCTION TEST
4732 017554  012706  000700             MOV     #STBOT-100,SP           ;SETUP TEST STACK = 700
4733 017560  012737  125252  000776     MOV     #125252,@#STBOT-2       ;SET UP NEW R5 VALUE ON STACK
4734 017566  012705  017604             MOV     #1$,R5                  ;PUT NEW PC IN R5
4735 017572  012746  006437             MOV     #MARK+37, (SP)          ;INSERT MARK 37 INSTRUCTION ONTO STACK
4736 017576  000277                     SCC
4737 017600  000116                     JMP     (SP)                    ;* TEST INSTRUCTION
                                                                        ;MARK INSTRUCTION SHOULD HAVE GONE TO 1$
4738
4739 017602  104001                     ERROR   +1              ;CPU ERROR
4740                                    ;
4741 017604  101002             1$:     BHI     2$                      ;ERROR IF C OR Z BIT CLEAR
4742 017606  100001                     BPL     2$                      ;ERROR IF N BIT CLEAR
4743 017610  102401                     BVS     3$                      ;BRANCH IF V BIT SET
                                                                        ;BAD CONDITION CODES ON MARK
4744
4745 017612  104001             2$:     ERROR   +1              ;CPU ERROR
4746 017614  022705  125252     3$:     CMP     #125252,R5              ;VERIFY R5
4747 017620  001401                     BEQ     4$                      ;BRANCH IF GOOD
4748 017622  104001                     ERROR   +1              ;CPU ERROR
4749
4750 017624  020627  001000     4$:     CMP     SP,#STBOT               ;VERIFY THAT STACK IS CORRECT
4751 017630  001401                     BEQ     15$                     ;BRANCH IF OK
                                                                        ;ERROR! STACK WAS NOT CORRECT AFTER MARK
4752
4753 017632  104001                     ERROR   +1              ;CPU ERROR
4754
4755 017634  012746  052525     15$:    MOV     #52525,-(SP)            ;SETUP EXPECTED R5
4756 017640  012746  006400             MOV     #6400,-(SP)             ;MOVE MARK 0 INSTRUCTION ON STACK
4757 017644  010605                     MOV     SP,R5                   ;R5=ADDRESS OF INSTRUCTION
4758 017646  004767  000004             JSR     PC,5$                   ;LEAVE 6$ ON STACK
4759 017652  000167  000006     6$:     JMP     16$                     ;MARK RETURNED CORRECTLY
4760
4761 017656  000257             5$:     CCC                             ;CLEAR THE CONDITION CODES
4762 017660  000205                     RTS     R5                      ;RETURN TO MARK INSTRUCTION
                                                                        ;NEXT INSTRUCTION ON STACK IS THE RETURN
4763                                                                    ;FROM THE JSR
4764                                                                    ;ERROR! BAD MARK SEQUENCE
4765
4766 017662  104001                     ERROR   +1              ;CPU ERROR
4767
4768 017664  101402             16$:    BLOS    7$                      ;IS C OR Z BIT SET?
4769 017666  100401                     BMI     7$                      ;IS N BIT SET?
4770 017670  102001                     BVC     8$                      ;IS V BIT SET?
                                                                        ;ERROR! CONDITIONS CODES INCORRECT
4771
4772 017672  104001             7$:     ERROR   +1              ;CPU ERROR
4773
4774 017674  020627  001000     8$:     CMP     R6,#STBOT               ;IS THE STACK CORRECT?
4775 017700  001401                     BEQ     9$                      ;BRANCH IF YES
                                                                        ;ERROR! BAD STACK CLEANUP
4776
4777 017702  104001                     ERROR   +1              ;CPU ERROR
4778 017704  022705  052525     9$:     CMP     #52525,R5               ;VERIFY THAT R5 WAS LOADED PROPERLY.
4779 017710  001401                     BEQ     10$                     ;IF OK, GO TO NEXT TEST.
                                                                        ;ERROR! R5 WAS NOT CORRECT AFTER MARK.
4780
4781 017712  104001                     ERROR   +1              ;CPU ERROR
4782 017714                     10$:
4784 017714                     XME100:
4786
4787                            ;       TEST CLEAR CONDITION CODES INSTRUCTION
```

```
********** DOUBLE OPERAND TESTS *****************

4788 017714  012737  030017  177776          MOV     #30017,@#177776         ;SETUP PSW
4789 017722  000257                          CCC                             ; TEST INSTRUCTION
4790 017724  022737  030000  177776          CMP     #30000,@#177776         ;DID IT CLEAR ALL CONDITION CODE BITS
4791 017732  001401                          BEQ     1$                      ;YES GO ON
4792 017734  104001                          ERROR   +1              ;CPU ERROR
4793                                                                 ;NO GO TO ERROR
4794 017736                          1$:
4795                                 ;
4797 017736                          XME101:
4799
4800                                 ;       TEST CLEAR C BIT INSTRUCTION
4801 017736  012737  030017  177776          MOV     #30017,@#177776         ;SETUP PSW
4802 017744  000241                          CLC                             ; TEST INSTRUCTION
4803 017746  022737  030016  177776          CMP     #30016,@#177776         ;DID IT CLEAR CARRY BIT
4804 017754  001401                          BEQ     1$                      ;YES GO ON
4805                                                                 ;C BIT NOT CLEAR GO TO ERROR
4806 017756  104001                          ERROR   +1              ;CPU ERROR
4807 017760                          1$:
4808                                 ;
4811 017760                          TE102:
4812                                 ;       TEST CLEAR N BIT INST (CLN)
4813 017760  012737  030017  177776          MOV     #30017,@#177776         ;SETUP PSW
4814 017766  000250                          CLN                             ; TEST INSTRUCTION
4815 017770  022737  030007  177776          CMP     #30007,@#177776         ;DID IT CLEAR NEGATIVE BIT
4816 017776  001401                          BEQ     1$                      ;YES GO ON
4817 020000  104001                          ERROR   +1              ;CPU ERROR
4818                                                                 ;NO GO TO ERROR
4819 020002                          1$:
4820                                 ;
4823 020002                          TE103:
4824                                 ;       TEST CLEAR V BIT INST (CLV)
4825 020002  012737  030017  177776          MOV     #30017,@#177776         ;SETUP PSW
4826 020010  000242                          CLV                             ; TEST INSTRUCTION
4827 020012  022737  030015  177776          CMP     #30015,@#177776         ;DID IT CLEAR OVERFLOW BIT
4828 020020  001401                          BEQ     1$                      ;YES GO ON
4829 020022  104001                          ERROR   +1              ;CPU ERROR
4830                                                                 ;NO GO TO ERROR
4831 020024                          1$:
4832                                 ;
4835 020024                          TE104:
4836                                 ;       TEST CLEAR Z BIT INST (CLZ)
4837 020024  012737  030017  177776          MOV     #30017,@#177776         ;SETUP PSW
4838 020032  000244                          CLZ                             ; TEST INSTRUCTION
4839 020034  022737  030013  177776          CMP     #30013,@#177776         ;DID IT CLEAR ZERO BIT
4840 020042  001401                          BEQ     1$                      ;YES GO ON
4841 020044  104001                          ERROR   +1              ;CPU ERROR
4842                                                                 ;NO GO TO ERROR
4843 020046                          1$:
4844                                 ;
4847 020046                          TE105:
4848                                 ;       TEST SET CONDITION CODES INST (SCC)
4849 020046  012737  030000  177776          MOV     #30000,@#177776         ;SETUP PSW
4850 020054  000277                          SCC                             ; TEST INSTRUCTION
4851 020056  022737  030017  177776          CMP     #30017,@#177776         ;DID IT SET ALL CONDITION CODE BITS
4852 020064  001401                          BEQ     1$                      ;YES GO ON
4853 020066  104001                          ERROR   +1              ;CPU ERROR
4854                                                                 ;NO GO TO ERROR
```

```
********** DOUBLE OPERAND TESTS *****************

4855 020070                                1$:
4856                                       ;
4859 020070                                TE106:
4860                                       ;       TEST SET C BIT INST (SEC)
4861 020070  012737  030000  177776                MOV     #30000,@#177776         ;SETUP PSW
4862 020076  000261                                SEC                             ; TEST INSTRUCTION
4863 020100  022737  030001  177776                CMP     #30001,@#177776         ;DID IT SET THE CARRY BIT
4864 020106  001401                                BEQ     1$                      ;YES GO ON
4865 020110  104001                                ERROR   +1              ;CPU ERROR
4866                                                                       ;NO GO TO ERROR
4867 020112                                1$:
4868                                       ;
4871 020112                                TE107:
4872                                       ;       TEST SET N BIT INST (SEN)
4873 020112  012737  030000  177776                MOV     #30000,@#177776         ;SETUP PSW
4874 020120  000270                                SEN                             ; TEST INSTRUCTION
4875 020122  022737  030010  177776                CMP     #30010,@#177776         ;DID IT SET THE NEGATIVE BIT
4876 020130  001401                                BEQ     1$                      ;YES GO ON
4877 020132  104001                                ERROR   +1              ;CPU ERROR
4878                                                                       ;NO GO TO ERROR
4879 020134                                1$:
4880                                       ;
4883 020134                                TE110:
4884                                       ;       TEST SET V BIT INST (SEV)
4885 020134  012737  030000  177776                MOV     #30000,@#177776         ;SETUP PSW
4886 020142  000262                                SEV                             ; TEST INSTRUCTION
4887 020144  022737  030002  177776                CMP     #30002,@#177776         ;DID IT SET THE OVERFLOW BIT
4888 020152  001401                                BEQ     1$                      ;YES GO ON
4889 020154  104001                                ERROR   +1              ;CPU ERROR
4890                                                                       ;NO GO TO ERROR
4891 020156                                1$:
4892                                       ;
4895 020156                                TE111:
4896                                       ;       TEST SET Z BIT INST (SEZ)
4897 020156  012737  030000  177776                MOV     #30000,@#177776         ;SETUP PSW
4898 020164  000264                                SEZ                             ; TEST INSTRUCTION
4899 020166  022737  030004  177776                CMP     #30004,@#177776         ;DID IT SET THE ZERO BIT
4900 020174  001401                                BEQ     1$                      ;YES GO ON
4901 020176  104001                                ERROR   +1              ;CPU ERROR
4902                                                                       ;NO GO TO ERROR
4903 020200                                1$:
4904                                       ;
4907 020200                                TE112:
4908                                       ;       TEST MULTIPLE CLEARS OF CC BITS
4909 020200  012737  030000  177776                MOV     #30000,@#177776         ;INIT PSW
4910 020206  000277                                SCC                             ;SETUP PSW
4911 020210  000243                                .WORD   243                     ; TEST CLC CLV
4912 020212  022737  030014  177776                CMP     #30014,@#177776         ;PSW CORRECT?
4913 020220  001401                                BEQ     1$                      ;YES GO ON
4914 020222  104001                                ERROR   +1              ;CPU ERROR
4915                                                                       ;NO GO TO ERROR
4916 020224  000277                        1$:     SCC                             ;SETUP PSW
4917 020226  000245                                .WORD   245                     ; TEST CLC CLZ
4918 020230  022737  030012  177776                CMP     #30012,@#177776         ;PSW CORRECT?
4919 020236  001401                                BEQ     2$                      ;YES GO ON
4920 020240  104001                                ERROR   +1              ;CPU ERROR
4921                                                                       ;NO GO TO ERROR
```

```
********** DOUBLE OPERAND TESTS *****************

 4922 020242  000277                    2$:     SCC                             ;SETUP PSW
 4923 020244  000246                            .WORD   246                     ; TEST CLV CLZ
 4924 020246  022737  030011  177776             CMP     #30011,@#177776         ;PSW CORRECT?
 4925 020254  001401                            BEQ     3$                      ;YES GO ON
 4926 020256  104001                            ERROR   .1              ;CPU ERROR
                                                                        ;NO GO TO ERROR
 4927
 4928 020260  000277                    3$:     SCC                             ;SETUP PSW
 4929 020262  000247                            .WORD   247                     ; TEST CLC CLV CLZ
 4930 020264  022737  030010  177776             CMP     #30010,@#177776         ;PSW CORRECT?
 4931 020272  001401                            BEQ     4$                      ;YES GO ON
 4932 020274  104001                            ERROR   .1              ;CPU ERROR
                                                                        ;NO GO TO ERROR
 4933
 4934 020276  000277                    4$:     SCC                             ;SETUP PSW
 4935 020300  000251                            .WORD   251                     ; TEST CLN CLC
 4936 020302  022737  030006  177776             CMP     #30006,@#177776         ;PSW CORRECT?
 4937 020310  001401                            BEQ     5$                      ;YES GO ON
 4938 020312  104001                            ERROR   .1              ;CPU ERROR
                                                                        ;NO GO TO ERROR
 4939
 4940 020314  000277                    5$:     SCC                             ;SETUP PSW
 4941 020316  000252                            .WORD   252                     ; TEST CLN CLV
 4942 020320  022737  030005  177776             CMP     #30005,@#177776         ;PSW CORRECT?
 4943 020326  001401                            BEQ     6$                      ;YES GO ON
 4944 020330  104001                            ERROR   .1              ;CPU ERROR
                                                                        ;NO GO TO ERROR
 4945
 4946 020332  000277                    6$:     SCC                             ;SETUP PSW
 4947 020334  000253                            .WORD   253                     ; TEST CLN CLC CLV
 4948 020336  022737  030004  177776             CMP     #30004,@#177776         ;PSW CORRECT?
 4949 020344  001401                            BEQ     7$                      ;YES GO ON
 4950 020346  104001                            ERROR   .1              ;CPU ERROR
                                                                        ;NO GO TO ERROR
 4951
 4952 020350  000277                    7$:     SCC                             ;SETUP PSW
 4953 020352  000254                            .WORD   254                     ; TEST CLN CLZ
 4954 020354  022737  030003  177776             CMP     #30003,@#177776         ;PSW CORRECT?
 4955 020362  001401                            BEQ     8$                      ;YES GO ON
 4956 020364  104001                            ERROR   .1              ;CPU ERROR
                                                                        ;NO GO TO ERROR
 4957
 4958 020366  000277                    8$:     SCC                             ;SETUP PSW
 4959 020370  000255                            .WORD   255                     ; TEST CLN CLC CLZ
 4960 020372  022737  030002  177776             CMP     #30002,@#177776         ;PSW CORRECT?
 4961 020400  001401                            BEQ     9$                      ;YES GO ON
 4962 020402  104001                            ERROR   .1              ;CPU ERROR
                                                                        ;NO GO TO ERROR
 4963
 4964 020404  000277                    9$:     SCC                             ;SETUP PSW
 4965 020406  000256                            .WORD   256                     ; TEST CLN CLV CLZ
 4966 020410  022737  030001  177776             CMP     #30001,@#177776         ;SETUP PSW
 4967 020416  001401                            BEQ     10$                     ;YES GO ON
 4968 020420  104001                            ERROR   .1              ;CPU ERROR
                                                                        ;NO GO TO ERROR
 4969
 4970 020422                            10$:
 4971                                   ;
 4974 020422                            TE113:
 4975                                   ;       TEST MULTIPLE SETS OF CC BITS
 4976 020422  012737  030000  177776             MOV     #30000,@#177776         ;INIT PSW
 4977 020430  000263                            .WORD   263                     ; TEST SEC SEV
 4978 020432  022737  030003  177776             CMP     #30003,@#177776         ;PSW CORRECT?
 4979 020440  001401                            BEQ     1$                      ;YES GO ON
 4980 020442  104001                            ERROR   .1              ;CPU ERROR
```

```
********** DOUBLE OPERAND TESTS *****************

4981
4982 020444  000257                    1$:     CCC                             ;NO GO TO ERROR
4983 020446  000265                            .WORD   265                             ;SETUP PSW
4984 020450  022737  030005  177776            CMP     #30005,@#177776         ; TEST SEC SEZ
4985 020456  001401                            BEQ     2$                      ;PSW CORRECT?
4986 020460  104001                            ERROR   +1                      ;YES GO ON
4987                                                                    ;CPU ERROR
4988 020462  000257                    2$:     CCC                             ;NO GO TO ERROR
4989 020464  000266                            .WORD   266                             ;SETUP PSW
4990 020466  022737  030006  177776            CMP     #30006,@#177776         ; TEST SEV SEZ
4991 020474  001401                            BEQ     3$                      ;PSW CORRECT?
4992 020476  104001                            ERROR   +1                      ;YES GO ON
4993                                                                    ;CPU ERROR
4994 020500  000257                    3$:     CCC                             ;NO GO TO ERROR
4995 020502  000267                            .WORD   267                             ;SETUP PSW
4996 020504  022737  030007  177776            CMP     #30007,@#177776         ; TEST SEC SEV SEZ
4997 020512  001401                            BEQ     4$                      ;PSW CORRECT?
4998 020514  104001                            ERROR   +1                      ;YES GO ON
4999                                                                    ;CPU ERROR
5000 020516  000257                    4$:     CCC                             ;NO GO TO ERROR
5001 020520  000271                            .WORD   271                             ;SETUP PSW
5002 020522  022737  030011  177776            CMP     #30011,@#177776         ; TEST SEN SEC
5003 020530  001401                            BEQ     5$                      ;PSW CORRECT?
5004 020532  104001                            ERROR   +1                      ;YES GO ON
5005                                                                    ;CPU ERROR
5006 020534  000257                    5$:     CCC                             ;NO GO TO ERROR
5007 020536  000272                            .WORD   272                             ;SETUP PSW
5008 020540  022737  030012  177776            CMP     #30012,@#177776         ; TEST SEN SEV
5009 020546  001401                            BEQ     6$                      ;PSW CORRECT?
5010 020550  104001                            ERROR   +1                      ;YES GO ON
5011                                                                    ;CPU ERROR
5012 020552  000257                    6$:     CCC                             ;NO GO TO ERROR
5013 020554  000273                            .WORD   273                             ;SETUP PSW
5014 020556  022737  030013  177776            CMP     #30013,@#177776         ; TEST SEN SEC SEV
5015 020564  001401                            BEQ     7$                      ;PSW CORRECT?
5016 020566  104001                            ERROR   +1                      ;YES GO ON
5017                                                                    ;CPU ERROR
5018 020570  000257                    7$:     CCC                             ;NO GO TO ERROR
5019 020572  000274                            .WORD   274                             ;SETUP PSW
5020 020574  022737  030014  177776            CMP     #30014,@#177776         ; TEST SEN SEZ
5021 020602  001401                            BEQ     8$                      ;PSW CORRECT?
5022 020604  104001                            ERROR   +1                      ;YES GO ON
5023                                                                    ;CPU ERROR
5024 020606  000257                    8$:     CCC                             ;NO GO TO ERROR
5025 020610  000275                            .WORD   275                             ;SETUP PSW
5026 020612  022737  030015  177776            CMP     #30015,@#177776         ; TEST SEN SEC SEZ
5027 020620  001401                            BEQ     9$                      ;PSW CORRECT?
5028 020622  104001                            ERROR   +1                      ;YES GO ON
5029                                                                    ;CPU ERROR
5030 020624  000257                    9$:     CCC                             ;NO GO TO ERROR
5031 020626  000276                            .WORD   276                             ;SETUP PSW
5032 020630  022737  030016  177776            CMP     #30016,@#177776         ; TEST SEN SEV SEZ
5033 020636  001401                            BEQ     10$                     ;PSW CORRECT?
5034 020640  104001                            ERROR   +1                      ;YES GO ON
5035                                                                    ;CPU ERROR
5036 020642                            10$:                            ;NO GO TO ERROR
5037                                   ;
```

********** DOUBLE OPERAND TESTS **********

```
5040 020642                    TE113A:
5041                           ;       TEST SIGNED AND CONDITIONAL BRANCHES
5042 020642  000257                    CCC                             ;CLE,  ALL CC BITS IN PSW
5043 020644  002001                    BGE     1$                      ;BGE SHOULD BRANCH
5044 020646  104001                    ERROR   +1              ;CPU ERROR
                                                               ;ERROR; DIDN'T BRANCH
5045
5046 020650  003001            1$:     BGT     2$                      ;BGT SHOULD BRANCH
5047 020652  104001                    ERROR   +1              ;CPU ERROR
                                                               ;ERROR;  DIDN'T BRANCH
5048
5049 020654  003401            2$:     BLE     3$                      ;BLE SHOULDN'T BRANCH
5050 020656  000401                    BR      4$                      ;BRANCH TO NEXT TEST
5051 020660  104001            3$:     ERROR   +1              ;CPU ERROR
                                                               ;ERROR; BLE SHOULD NOT HAVE BRANCHED
5052
5053 020662  002401            4$:     BLT     5$                      ;BLT SHOULD NOT BRANCH
5054 020664  000401                    BR      6$                      ;BRANCH TO NEXT TEST
5055 020566  104001            5$:     ERROR   +1              ;CPU ERROR
                                                               ;ERROR; BLT SHOULD NOT HAVE BRANCHED
5056
5057 020670  000264            6$:     SEZ                             ;SET THE Z BIT IN PSW
5058 020672  003401                    BLE     7$                      ;BLE SHOULD BRANCH
5059 020674  104001                    ERROR   +1              ;CPU ERROR
                                                               ;ERROR; BLE DIDN'T BRANCH
5060
5061 020676  003001            7$:     BGT     8$                      ;BGT SHOULD NOT BRANCH
5062 020700  000401                    BR      9$                      ;BRANCH TO NEXT TEST
5063 020702  104001            8$:     ERROR   +1              ;CPU ERROR
                                                               ;ERROR; BGT SHOULD NOT HAVE BRANCHED
5064
5065 020704  000257            9$:     CCC                             ;CLEAR ALL CC BITS IN PSW
5066 020706  000270                    SEN                             ;SET N BIT IN PSW
5067 020710  002401                    BLT     10$                     ;SHOULD BRANCH TO NEXT TEST
5068 020712  104001                    ERROR   +1              ;CPU ERROR
                                                               ;ERROR; BLT SHOULD HAVE BRANCHED
5069
5070 020714  003401            10$:    BLE     11$                     ;SHOULD BRANCH TO NEXT TEST
5071 020716  104001                    ERROR   +1              ;CPU ERROR
                                                               ;ERROR; BLE SHOULD HAVE BRANCHED
5072
5073 020720  002001            11$:    BGE     12$                     ;BGE SHOULD NOT BRANCH
5074 020722  000[illegible]             BR      13$                     ;BRANCH TO NEXT TEST
5075 020724  104[illegible]    12$:    ERROR   +1              ;CPU ERROR
                                                               ;ERROR; BGE SHOULD NOT HAVE BRANCHED
5076
5077 020726  003001            13$:    BGT     14$                     ;BGT SHOULD NOT BRANCH
5078 020730  000401                    BR      15$                     ;BRANCH TO NEXT TEST
5079 020732  104001            14$:    ERROR   +1              ;CPU ERROR
                                                               ;ERROR; BGT SHOULD NOT HAVE BRANCHED
5080
5081 020734  000257            15$:    CCC                             ;CLEAR ALL CC BITS
5082 020736  000262                    SEV                             ;SET V BIT IN PSW
5083 020740  003401                    BLE     16$                     ;BLE SHOULD BRANCH
5084 020742  104001                    ERROR   +1              ;CPU ERROR
                                                               ;ERROR; BLE DIDN'T BRANCH
5085
5086 020744  002401            16$:    BLT     17$                     ;BLT SHOULD BRANCH
5087 020746  104001                    ERROR   +1              ;CPU ERROR
                                                               ;ERROR; BLT DIDN'T BRANCH
5088
5089 020750  002001            17$:    BGE     18$                     ;BGE SHOULDN'T BRANCH
5090 020752  000401                    BR      19$                     ;BRANCH TO NEXT TEST
5091 020754  104001            18$:    ERROR   +1              ;CPU ERROR
                                                               ;ERROR; BGE SHOULD NOT HAVE BRANCHED
5092
5093 020756  003001            19$:    BGT     20$                     ;BGT SHOULD NOT BRANCH
5094 020760  000401                    BR      21$                     ;BRANCH TO NEXT TEST
5095 020762  104001            20$:    ERROR   +1              ;CPU ERROR
                                                               ;ERROR; BGT SHOULD NOT HAVE BRANCHED
5096
```

```
********** DOUBLE OPERAND TESTS *****************

5097 020764  000257                  21$:    CCC                     ;CLEAR ALL CC BITS
5098 020766  000272                          .WORD   272             ;SET N AND V BITS IN PSW
5099 020770  002001                          BGE     22$             ;BGE SHOULD BRANCH
5100 020772  104001                          ERROR   +1      ;CPU ERROR
                                             ;ERROR; BGE DIDN T BRANCH
5101
5102 020774  003001                  22$:    BGT     23$             ;BGT SHOULD BRANCH
5103 020776  104001                          ERROR   +1      ;CPU ERROR
                                             ;ERROR; BGT DIDN'T BRANCH
5104
5105 021000  003401                  23$:    BLE     24$             ;BLE SHOULDN'T BRANCH
5106 021002  000401                          BR      25$             ;BRANCH TO NEXT TEST
5107 021004  104001                  24$:    ERROR   +1      ;CPU ERROR
                                             ;ERROR; BLE SHOULD NOT HAVE BRANCHED
5108
5109 021006  002401                  25$:    BLT     26$             ;BLT SHOULD NOT BRANCH
5110 021010  000401                          BR      27$             ;BRANCH TO NEXT TEST
5111 021012  104001                  26$:    ERROR   +1      ;CPU ERROR
                                             ;ERROR; BLT SHOULD NOT HAVE BRANCHED
5112
5113 021014                          27$:
5116 021014                          TE114:
5117                                 ;       TEST NOP INST
5118 021014  012737  030000  177776          MOV     #30000,@#177776 ;INIT PSW
5119 021022  012700  000001                  MOV     #1,R0           ;INIT R0
5120 021026  012701  000002                  MOV     #2,R1           ;INIT R1
5121 021032  012702  000003                  MOV     #3,R2           ;INIT R2
5122 021036  012703  000004                  MOV     #4,R3           ;INIT R3
5123 021042  012704  000005                  MOV     #5,R4           ;INIT R4
5124 021046  012705  000006                  MOV     #6,R5           ;INIT R5
5125 021052  010637  003012                  MOV     R6,@#SLOC00     ;SAVE SP
5126 021056  012737  030017  114146          MOV     #30017,@#EXPDAT ;SETUP PSW
5127 021064  000277                          SCC                     ;SET ALL CONDITION CODE BITS
5128 021066  000240                          NOP                     ; TEST INSTRUCTION
5129 021070  000240                          NOP                     ; TEST INSTRUCTION
5130 021072  000240                          NOP                     ; TEST INSTRUCTION
5131 021074  004767  000046                  JSR     PC,T114         ;CHECK PSW, AND GPR'S
5132 021100  020637  003012                  CMP     R6,@#SLOC00     ;CHECK SP
5133 021104  001401                          BEQ     1$              ;OK GO ON
5134 021106  104001                          ERROR   +1      ;CPU ERROR
                                             ;NO GO TO ERROR
5135
5136 021110  012737  030000  114146  1$:     MOV     #30000,@#EXPDAT ;SETUP PSW
5137 021116  000257                          CCC                     ;CLEAR ALL CONDITION CODE BITS
5138 021120  000260                          .WORD   260             ; TEST FOR NOP OPERATION
5139 021122  000260                          .WORD   260             ; TEST FOR NOP OPERATION
5140 021124  000260                          .WORD   260             ; TEST FOR NOP OPERATION
5141 021126  004767  000014                  JSR     PC,T114         ;CHECK PSW, AND GPR'S
5142 021132  020637  003012                  CMP     R6,@#SLOC00     ;CHECK SP
5143 021136  001401                          BEQ     2$              ;OK GO ON
5144 021140  104001                          ERROR   +1      ;CPU ERROR
                                             ;NO GO TO ERROR
5145
5146 021142                          2$:
5147 021142  000167  000104                  JMP     FINNOP
5148                                 ;
5149 021146  023737  114146  177776  T114:   CMP     @#EXPDAT,@#177776       ;CHECK PSW
5150 021154  001405                          BEQ     TA114           ;OK GO ON
5151 021156  010067  157216                  MOV     R0,400          ;SAVE R0
5152 021162  104001                          ERROR   +1      ;CPU ERROR
                                             ;NO GO TO ERROR
5153
5154 021164  016700  157210                  MOV     400,R0          ;RESTORE R0
5155 021170  022700  000001          TA114:  CMP     #1,R0           ;CHECK R0
```

```
********** DOUBLE OPERAND TESTS *****************

5156 021174  001401                        BEQ     TB114                   ;OK GO ON
5157 021176  104001                        ERROR   +1              ;CPU ERROR
5158                                                               ;NO GO TO ERROR
5159 021200  022701  000002          TB114: CMP     #2,R1                   ;CHECK R1
5160 021204  001401                        BEQ     TC114                   ;OK GO ON
5161 021206  104001                        ERROR   +1              ;CPU ERROR
5162                                                               ;NO GO TO ERROR
5163 021210  022702  000003          TC114: CMP     #3,R2                   ;CHECK R2
5164 021214  001401                        BEQ     TD114                   ;OK GO ON
5165 021216  104001                        ERROR   +1              ;CPU ERROR
5166                                                               ;NO GO TO ERROR
5167 021220  022703  000004          TD114: CMP     #4,R3                   ;CHECK R3
5168 021224  001401                        BEQ     TF114                   ;OK GO ON
5169 021226  104001                        ERROR   +1              ;CPU ERROR
5170                                                               ;NO GO TO ERROR
5171 021230  022704  000005          TF114: CMP     #5,R4                   ;CHECK R4
5172 021234  001401                        BEQ     TG114                   ;OK GO ON
5173 021236  104001                        ERROR   +1              ;CPU ERROR
5174                                                               ;NO GO TO ERROR
5175 021240  022705  000006          TG114: CMP     #6,R5                   ;CHECK R5
5176 021244  001401                        BEQ     TH114                   ;OK GO ON
5177 021246  104001                        ERROR   +1              ;CPU ERROR
5178                                                               ;NO GO TO ERROR
5179 021250  000207                  TH114: RTS     PC                      ;RETURN
5180                                 ;
5181 021252  000240                  FINNOP: NOP
5182                                 ;
5185                                 ;-  ---          ------ -  ---
5186                                 ;TEST ATERNATE REGISTER SET
5187                                 ;
5188
5189 021254  005000                        CLR     R0                      ;-  ---CLEAR  --
5190 021256  005001                        CLR     R1                      ;---  PRIMARY- --
5191 021260  005002                        CLR     R2                      ;-----GENERAL-----
5192 021262  005003                        CLR     R3                      ;-----PURPOSE-----
5193 021264  005004                        CLR     R4                      ;---- REGISTER- --
5194 021266  005005                        CLR     R5                      ;-------SET-------
5195 021270  012767  004000  156500        MOV     #4000,PS                ;SELECT ALTERNATE REGISTER SET
5196 021276  032767  004000  156472        BIT     #BIT11,PS               ;TEST TO SEE THAT BIT IS SET IN PS
5197 021304  001001                        BNE     1$                      ;IF IT'S SET GO TESTS REGISTERS
5198 021306  104001                        ERROR   +1              ;CPU ERROR
5199                                                               ;ERROR! ALTERNATE REGISTER SELECT
5200                                                                       ;BIT IN PS IS NOT SET
5201 021310  012700  177777          1$:    MOV     #177777,R0              ;WRITE ALL ONES TO ALTERNATE REG SET
5202 021314  010001                        MOV     R0,R1                   ;
5203 021316  010102                        MOV     R1,R2                   ;
5204 021320  010203                        MOV     R2,R3                   ;
5205 021322  010304                        MOV     R3,R4                   ;
5206 021324  010405                        MOV     R4,R5                   ;
5207 021326  042767  004000  156442        BIC     #BIT11,PS               ;CLEAR BIT 11 IN PS
5208 021334  032767  004000  156434        BIT     #BIT11,PS               ;IS BIT 11 = 0?
5209 021342  001401                        BEQ     2$                      ;IF IT'S NOT ZERO
5210 021344  104001                        ERROR   +1              ;CPU ERROR
5211                                                               ;ERROR! BIT 11 DID NOT CLEAR
5212 021346  005700                  2$:   TST     R0                      ;MAKE SURE THAT WE REALLY
5213 021350  001401                        BEQ     3$                      ;WERE TALKING TO ALTERNATE REGISTERS.
5214 021352  104001                        ERROR   +1              ;CPU ERROR
```

```
                                                                ;ERROR!
021354  005701                      3$:     TST     R1                  ;
021356  001401                              BEQ     4$                  ;
021360  104001                              ERROR   +1                  ;CPU ERROR
                                                                ;ERROR!
021362  005702                      4$:     TST     R2                  ;
021364  001401                              BEQ     5$                  ;
021366  104001                              ERROR   +1                  ;CPU ERROR
                                                                ;ERROR!
021370  005703                      5$:     TST     R3                  ;
021372  001401                              BEQ     6$                  ;
021374  104001                              ERROR   +1                  ;CPU ERROR
                                                                ;
021376  005704                      6$:     TST     R4                  ;
021400  001401                              BEQ     7$                  ;
021402  104001                              ERROR   +1                  ;CPU ERROR
                                                                ;
021404  005705                      7$:     TST     R5                  ;
021406  001401                              BEQ     8$                  ;
021410  104001                              ERROR   +1                  ;CPU ERROR
                                                                ;ERROR!
021412  012767  004000  156356      8$:     MOV     #BIT11,PS           ;ENABLE ALTERNATE REGISTER SET
021420  005000                              CLR     R0                  ;-----CLEAR-----
021422  005001                              CLR     R1                  ;- -ALTERNATE---
021424  005002                              CLR     R2                  ;----REGISTER---
021426  005003                              CLR     R3                  ;---- -SET----
021430  005004                              CLR     R4                  ;
021432  005005                              CLR     R5                  ;
021434  042767  004000  156334              BIC     #BIT11,PS           ;BACK TO PRIMARY GPRS
021442  012700  177777                      MOV     #177777,R0          ;ENSURE THAT WRITES TO PRIMARY GPRS
021446  010001                              MOV     R0,R1               ;DO NOT EFFECT ALTERNATE GPRS
021450  010102                              MOV     R1,R2               ;
021452  010203                              MOV     R2,R3               ;
021454  010304                              MOV     R3,R4               ;
021456  010405                              MOV     R4,R5               ;
021460  052767  004000  156310              BIS     #BIT11,PS           ;RETURN TO ALTERNATE REGISTERS
021466  005700                              TST     R0                  ;SHOULD BE ALL 0'S
021470  001401                              BEQ     9$                  ;
021472  104001                              ERROR   +1                  ;CPU ERROR
                                                                ;ERROR!
021474  005701                      9$:     TST     R1                  ;
021476  001401                              BEQ     10$                 ;
021500  104001                              ERROR   +1                  ;CPU ERROR
                                                                ;
021502  005702                      10$:    TST     R2                  ;
021504  001401                              BEQ     11$                 ;
021506  104001                              ERROR   +1                  ;CPU ERROR
                                                                ;
021510  005703                      11$:    TST     R3                  ;
021512  001401                              BEQ     12$                 ;
021514  104001                              ERROR   +1                  ;CPU ERROR
                                                                ;
021516  005704                      12$:    TST     R4                  ;
021520  001401                              BEQ     13$                 ;
021522  104001                              ERROR   +1                  ;CPU ERROR
                                                                ;
021524  005705                      13$:    TST     R5                  ;
```

```
********** DOUBLE OPERAND TESTS *****************

5272 021526  001401                        BEQ     14$
5273 021530  104001                        ERROR   +1              ;CPU ERROR
5274 021532                         14$:
5275
5276 021532                         ALROTS:
5277                                ;       ALTERNATE REGISTER SET R0 BIT TESTS
5278 021532  012700  177777                MOV     #177777,R0              ;R0=177777
5279 021536  020027  177777                CMP     R0,#177777              ;DOES R0=177777
5280 021542  001401                        BEQ     1$                      ;YES GO ON
5281 021544  104001                        ERROR   +1              ;CPU ERROR
5282                                                                        ;NO GO TO ERROR
5283 021546  005000                 1$:    CLR     R0                      ;R0=0
5284 021550  020027  000000                CMP     R0,#0                   ;DOES R0=0
5285 021554  001401                        BEQ     2$                      ;YES GO ON
5286 021556  104001                        ERROR   +1              ;CPU ERROR
5287                                                                        ;NO GO TO ERROR
5288 021560  012700  125252           2$:    MOV     #125252,R0              ;R0=125252
5289 021564  020027  125252                CMP     R0,#125252              ;DOES R0=125252
5290 021570  001401                        BEQ     3$                      ;YES GO ON
5291 021572  104001                        ERROR   +1              ;CPU ERROR
5292                                                                        ;NO GO TO ERROR
5293 021574  012700  052525           3$:    MOV     #52525,R0               ;R0=52525
5294 021600  020027  052525                CMP     R0,#52525               ;DOES R0=52525
5295 021604  001401                        BEQ     4$                      ;YES GO ON
5296 021606  104001                        ERROR   +1              ;CPU ERROR
5297                                                                        ;NO GO TO ERROR
5298 021610                         4$:
5299                                ;
5302 021610                         ALR1TS:
5303                                ;       ALTERNATE REGISTER SET R1 BIT TESTS
5304 021610  012701  177777                MOV     #177777,R1              ;R1=177777
5305 021614  020127  177777                CMP     R1,#177777              ;DOES R1=177777
5306 021620  001401                        BEQ     1$                      ;YES GO ON
5307 021622  104001                        ERROR   +1              ;CPU ERROR
5308                                                                        ;NO GO TO ERROR
5309 021624  005001                 1$:    CLR     R1                      ;R1=0
5310 021626  020127  000000                CMP     R1,#0                   ;DOES R1=0
5311 021632  001401                        BEQ     2$                      ;YES GO ON
5312 021634  104001                        ERROR   +1              ;CPU ERROR
5313                                                                        ;NO GO TO ERROR
5314 021636  012701  125252           2$:    MOV     #125252,R1              ;R1=125252
5315 021642  020127  125252                CMP     R1,#125252              ;DOES R1=125252
5316 021646  001401                        BEQ     3$                      ;YES GO ON
5317 021650  104001                        ERROR   +1              ;CPU ERROR
5318                                                                        ;NO GO TO ERROR
5319 021652  012701  052525           3$:    MOV     #52525,R1               ;R1=52525
5320 021656  020127  052525                CMP     R1,#52525               ;DOES R1=52525
5321 021662  001401                        BEQ     4$                      ;YES GO ON
5322 021664  104001                        ERROR   +1              ;CPU ERROR
5323                                                                        ;NO GO TO ERROR
5324 021666                         4$:
5325
5328 021666                         ALP2TS:
5329                                ;       ALTERNATE REGISTER SET R2 BIT TESTS
5330 021666  012702  177777                MOV     #177777,R2              ;R2=177777
5331 021672  020227  177777                CMP     R2,#177777              ;DOES R2=177777
5332 021676  001401                        BEQ     1$                      ;YES GO ON
```

```
********** DOUBLE OPERAND TESTS *****************

5333 021700  104001                          ERROR   +1              ;CPU ERROR
                                                                     ;NO GO TO ERROR
5334
5335 021702  005002                  1$:     CLR     R2              ;R2=0
5336 021704  020227  000000                  CMP     R2,#0           ;DOES R2=0
5337 021710  001401                          BEQ     2$              ;YES GO ON
5338 021712  104001                          ERROR   +1              ;CPU ERROR
                                                                     ;NO GO TO ERROR
5339
5340 021714  012702  125252          2$:     MOV     #125252,R2      ;R2=125252
5341 021720  020227  125252                  CMP     R2,#125252      ;DOES R2=125252
5342 021724  001401                          BEQ     3$              ;YES GO ON
5343 021726  104001                          ERROR   +1              ;CPU ERROR
                                                                     ;NO GO TO ERROR
5344
5345 021730  012702  052525          3$:     MOV     #52525,R2       ;R2=52525
5346 021734  020227  052525                  CMP     R2,#52525       ;DOES R2=52525
5347 021740  001401                          BEQ     4$              ;YES GO ON
5348 021742  104001                          ERROR   +1              ;CPU ERROR
                                                                     ;NO GO TO ERROR
5349
5350 021744                          4$:
5351                                 ;
5354 021744                          ALR3TS:
5355                                 ;       ALTERNATE REGISTER SET R3 BIT TESTS
5356 021744  012703  177777                  MOV     #177777,R3      ;R3=177777
5357 021750  020327  177777                  CMP     R3,#177777      ;DOES R3=177777
5358 021754  001401                          BEQ     1$              ;YES GO ON
5359 021756  104001                          ERROR   +1              ;CPU ERROR
                                                                     ;NO GO TO ERROR
5360
5361 021760  005003                  1$:     CLR     R3              ;R3=0
5362 021762  020327  000000                  CMP     R3,#0           ;DOES R3=0
5363 021766  001401                          BEQ     2$              ;YES GO ON
5364 021770  104001                          ERROR   +1              ;CPU ERROR
                                                                     ;NO GO TO ERROR
5365
5366 021772  012703  125252          2$:     MOV     #125252,R3      ;R3=125252
5367 021776  020327  125252                  CMP     R3,#125252      ;DOES R3=125252
5368 022002  001401                          BEQ     3$              ;YES GO ON
5369 022004  104001                          ERROR   +1              ;CPU ERROR
                                                                     ;NO GO TO ERROR
5370
5371 022006  012703  052525          3$:     MOV     #52525,R3       ;R3=52525
5372 022012  020327  052525                  CMP     R3,#52525       ;DOES R3=52525
5373 022016  001401                          BEQ     4$              ;YES GO ON
5374 022020  104001                          ERROR   +1              ;CPU ERROR
                                                                     ;NO GO TO ERROR
5375
5376 022022                          4$:
5377                                 ;
5380 022022                          ALR4TS:
5381                                 ;       ALTERNATE REGISTER SET R4 BIT TESTS
5382 022022  012704  177777                  MOV     #177777,R4      ;R4=177777
5383 022026  020427  177777                  CMP     R4,#177777      ;DOES R4=177777
5384 022032  001401                          BEQ     1$              ;YES GO ON
5385 022034  104001                          ERROR   +1              ;CPU ERROR
                                                                     ;NO GO TO ERROR
5386
5387 022036  005004                  1$:     CLR     R4              ;R4=0
5388 022040  020427  000000                  CMP     R4,#0           ;DOES R4=0
5389 022044  001401                          BEQ     2$              ;YES GO ON
5390 022046  104001                          ERROR   +1              ;CPU ERROR
                                                                     ;NO GO TO ERROR
5391
5392 022050  012704  125252          2$:     MOV     #125252,R4      ;R4=125252
5393 022054  020427  125252                  CMP     R4,#125252      ;DOES R4=125252
```

```
********** DOUBLE OPERAND TESTS *****************

5394 022060  001401                        BEQ     3$                      ;YES CO ON
5395 022062  104001                        ERROR   +1              ;CPU ERROR
5396                                                                       ;NO GO TO ERROR
5397 022064  012704  052525        3$:     MOV     #52525,R4               ;R4=52525
5398 022070  020427  052525                CMP     R4,#52525               ;DOES R4=52525
5399 022074  001401                        BEQ     4$                      ;YES GO ON
5400 022076  104001                        ERROR   +1              ;CPU ERROR
5401                                                                       ;NO GO TO ERROR
5402 022100                        4$:
5403                                ;
5406 022100                        ALR5TS:
5407                                ;       ALTERNATE REGISTER SET R5 BIT TESTS
5408 022100  012705  177777                MOV     #177777,R5              ;R5=177777
5409 022104  020527  177777                CMP     R5,#177777              ;DOES R5=177777
5410 022110  001401                        BEQ     1$                      ;YES GO ON
5411 022112  104001                        ERROR   +1              ;CPU ERROR
5412                                                                       ;NO GO TO ERROR
5413 022114  005005                1$:     CLR     R5                      ;R5=0
5414 022116  020527  000000                CMP     R5,#0                   ;DOES R5=0
5415 022122  001401                        BEQ     2$                      ;YES GO ON
5416 022124  104001                        ERROR   +1              ;CPU ERROR
5417                                                                       ;NO GO TO ERROR
5418 022126  012705  125252        2$:     MOV     #125252,R5              ;R5=125252
5419 022132  020527  125252                CMP     R5,#125252              ;DOES R5=125252
5420 022136  001401                        BEQ     3$                      ;YES GO ON
5421 022140  104001                        ERROR   +1              ;CPU ERROR
5422                                                                       ;NO GO TO ERROR
5423 022142  012705  052525        3$:     MOV     #52525,R5               ;R5=52525
5424 022146  020527  052525                CMP     R5,#52525               ;DOES R5=52525
5425 022152  001401                        BEQ     4$                      ;YES GO ON
5426 022154  104001                        ERROR   +1              ;CPU ERROR
5427                                                                       ;NO GO TO ERROR
5428 022156  042767  004000  155612  4$:     BIC     #BIT11,PS               ;RETURN TO PRIMARY GEN PURPOSE REGS
5429
5430                                ;
5433 022164                        TE115:
5434                                ;       TEST MOVE FROM PROCCESSOR STATUS INST (MFPS)
5435 022164  005004                        CLR     R4                      ;SETUP DESTINATION R4
5436
5437 022166  012701  022374                MOV     #TE115A,R1              ;SETUP POINTERS TO TABLES
5438 022172  012702  022404                MOV     #TE115B,R2              ;
5439 022176  012703  022412                MOV     #TE115C,R3              ;
5440 022202  012137  177776        1$:     MOV     (R1)+,@#177776          ;SETUP PSW
5441 022206  106704                        MFPS    R4                      ; TEST INSTRUCTION
5442 022210  023722  177776                CMP     @#177776,(R2)+          ;CHECK PSW
5443 022214  001401                        BEQ     2$                      ;OK GO ON
5444 022216  104001                        ERROR   +1              ;CPU ERROR
5445                                                                ;NO GO TO ERROR
5446 022220  020423                2$:     CMP     R4,(R3)+                ;CHECK R4
5447 022222  001401                        BEQ     3$                      ;OK GO ON
5448 022224  104001                        ERROR   +1              ;CPU ERROR
5449                                                                ;NO GO TO ERROR
5450 022226  021127  177777        3$:     CMP     (R1),#177777            ;ARE WE DONE
5451 022232  001363                        BNE     1$                      ;NO GO TO 1$
5452
5453
5454 022234  012701  022374                MOV     #TE115A,R1              ;SETUP POINTERS TO TABLES
```

```
********** DOUBLE OPERAND TESTS *****************

5455 022240  012702  022404                MOV     #TE115B,R2              ;
5456 022244  012703  022412                MOV     #TE115C,R3              ;
5457 022250  010605                        MOV     R6,R5                   ;SAVE STACK IN R5
5458 022252  011137  177776          101$: MOV     (R1),@#177776           ;SETUP PSW
5459 022256  106706                        MFPS    R6                      ; TEST INSTRUCTION
5460 022260  023712  177776                CMP     @#177776,(R2)           ;CHECK PSW
5461 022264  001401                        BEQ     102$                    ;OK GO ON
5462 022266  104001                        ERROR   +1              ;CPU ERROR
                                                                   ;NO GO TO ERROR
5463
5464 022270  020613                  102$: CMP     R6,(R3)                 ;CHECK R6
5465 022272  001401                        BEQ     103$                    ;OK GO ON
5466 022274  104001                        ERROR   +1              ;CPU ERROR
                                                                   ;NO GO TO ERROR
5467
5468 022276  010506                  103$: MOV     R5,R6                   ;RESTORE STACK
5469
5470
5471 022300  012701  022374                MOV     #TE115A,R1              ;SETUP POINTERS TO TABLES
5472 022304  012702  022404                MOV     #TE115B,R2              ;
5473 022310  012703  022420                MOV     #TE115D,R3              ;
5474 022314  005037  114146                CLR     @#EXPDAT                ;INIT EXPECTED DATA HOLDER.
5475 022320  012704  114146          4$:   MOV     #EXPDAT,R4              ;SETUP POINTER TO TEST LOCATION
5476 022324  012137  177776                MOV     (R1)+,@#177776          ;SETUP PSW
5477 022330  106724                        MFPS    (R4)                    ; TEST INSTRUCTION
5478 022332  023722  177776                CMP     @#177776,(R2)+          ;CHECK PSW
5479 022336  001401                        BEQ     5$                      ;OK GO ON
5480 022340  104001                        ERROR   +1              ;CPU ERROR
                                                                   ;NO GO TO ERROR
5481
5482 022342  020427  114147          5$:   CMP     R4,#EXPDAT+1            ;CHECK R4
5483 022346  001401                        BEQ     6$                      ;OK GO ON
5484 022350  104001                        ERROR   +1              ;CPU ERROR
                                                                   ;NO GO TO ERROR
5485
5486 022352  023723  114146          6$:   CMP     @#EXPDAT,(R3)+          ;CHECK TEST LOCATION
5487 022356  001401                        BEQ     7$                      ;OK GO ON
5488 022360  104001                        ERROR   +1              ;CPU ERROR
                                                                   ;NO GO TO ERROR
5489
5490 022362  021127  177777          7$:   CMP     (R1),#177777            ;ARE WE DONE
5491 022366  001354                        BNE     4$                      ;NO GO TO 4$
5492
5493 022370  000167  000032                JMP     TE115F
5494
5495                                  ;
5496                                  ;
5497 022374  030207                TE115A: .WORD   30207
5498 022376  030000                        .WORD   30000
5499 022400  030057                        .WORD   30057
5500 022402  177777                        .WORD   177777
5501 022404  030211                TE115B: .WORD   30211
5502 022406  030004                        .WORD   30004
5503 022410  030041                        .WORD   30041
5504 022412  177607                TE115C: .WORD   177607
5505 022414  000000                        .WORD   0
5506 022416  000057                        .WORD   57
5507 022420  000207                TE115D: .WORD   207
5508 022422  000000                        .WORD   0
5509 022424  000057                        .WORD   57
5510 022426  000240                TE115F: NOP
5511                                  ;
```

N8


```
********** DOUBLE OPERAND TESTS *****************

5514 022430                            TE116:
5515                                   ;       TEST MOVE TO PROCESSOR STATUS INST (MTPS)
5516
5517 022430  012737  030000  177776            MOV     #30000,@#177776         ;SET PSW TO KERNEL MODE
5518 022436  012701  022732                      MOV     #TE116D,R1              ;SETUP POINTERS TO TABLES
5519 022442  012702  022710                      MOV     #TE116B,R2              ;
5520 022446  010103                              MOV     R1,R3                   ;
5521 022450  004767  000136                      JSR     PC,T116                 ; TEST INSTRUCTION AND CHECK PSW
5522                                                                             ;AND SOURCE OPERAND
5523
5524
5525 022454  012737  140000  177776            MOV     #140000,@#177776        ;SET PSW TO USER MODE
5526 022462  012706  000600                      MOV     #600,R6                 ;SETUP USER STACK
5527 022466  012701  022732                      MOV     #TE116D,R1              ;SETUP POINTERS TO TABLES
5528 022472  012702  022722                      MOV     #TE116C,R2              ;
5529 022476  010103                              MOV     R1,R3                   ;
5530 022500  004767  000106                      JSR     PC,T116                 ; TEST INSTRUCTION AND CHECK PSW
5531                                                                             ;AND SOURCE OPERAND
5532
5533
5534 022504  012737  140000  177776            MOV     #140000,@#177776        ;SET PSW TO USER MODE AND CLEAR CC BITS
5535 022512  012701  022704                      MOV     #TE116A,R1              ;SETUP POINTERS TO TABLES
5536 022516  012702  022722                      MOV     #TE116C,R2              ;
5537 022522  012703  022732                      MOV     #TE116D,R3              ;
5538 022526  010104                              MOV     R1,R4                   ;SAVE A COPY OF R1 INTO R4
5539 022530  004767  000110                      JSR     PC,TA116                ; TEST INSTRUCTION AND CHECK PSW
5540                                                                             ;AND SOURCE OPERAND
5541
5542
5543 022534  012737  030000  177776            MOV     #30000,@#177776         ;SET PSW TO KERNEL MODE
5544 022542  012701  022704                      MOV     #TE116A,R1              ;SETUP POINTERS TO TABLES
5545 022546  012702  022710                      MOV     #TE116B,R2              ;
5546 022552  012703  022732                      MOV     #TE116D,R3              ;
5547 022556  010104                              MOV     R1,R4                   ;SAVE A COPY OF R1 INTO R4
5548 022560  004767  000060                      JSR     PC,TA116                ; TEST INSTRUCTION AND CHECK PSW
5549                                                                             ;AND SOURCE OPERAND
5550
5551 022564  005037  177776                      CLR     @#177776                ;SET PSW TO KERNEL MODE
5552 022570  106427  177412                      MTPS    #177412                 ;TEST INSTRUCTION
5553 022574  022737  000012  177776            CMP     #12,@#177776            ;IS PSW CORRECT
5554 022602  001401                              BEQ     100$                    ;YES GO ON
5555 022604  104001                              ERROR   +1                      ;CPU ERROR
5556                                                                             ;NO GO TO ERROR
5557 022606                            100$:
5559 022606  000167  000130                      JMP     FIN116
5559                                   ;
5560 022612  012105                    T116:     MOV     (R1)+,R5                ;MOVE TEST DATA TO R5
5561 022614  106405                              MTPS    R5                      ; TEST INSTRUCTION
5562 022616  023722  177776                      CMP     @#177776,(R2)+          ;IS PSW CORRECT
5563 022622  001401                              BEQ     1$                      ;YES GO ON
5564 022624  104001                              ERROR   +1                      ;CPU ERROR
5565                                                                             ;NO GO TO ERROR
5566 022626  022305                    1$:       CMP     (R3)+,R5                ;IS R5 CORRECT
5567 022630  001401                              BEQ     2$                      ;YES GO ON
5568 022632  104001                              ERROR   +1                      ;CPU ERROR
5569                                                                             ;NO GO TO ERROR
5570 022634  021227  177777            2$:       CMP     (R2),#177777            ;ARE WE DONE
```

```
********** DOUBLE OPERAND TESTS *****************

5571 022640  001364                        BNE     T116                    ;NO GO TO T116
5572 022642  000207                        RTS     PC                      ;RETURN
5573
5574 022644  106421                TA116:  MTPS    (R1)+                   ; TEST INSTRUCTION
5575 022646  023722  177776                CMP     @#177776,(R2)+          ;IS PSW CORRECT
5576 022652  001401                        BEQ     1$                      ;YES GO ON
5577 022654  104001                        ERROR   +1              ;CPU ERROR
                                                                   ;NO GO TO ERROR
5578
5579 022656  122423                1$:     CMPB    (R4)+,(R3)+             ;CHECK TEST LOCATION
5580 022660  001401                        BEQ     2$                      ;OK GO ON
5581 022662  104001                        ERROR   +1              ;CPU ERROR
                                                                   ;NO GO TO ERROR
5582
5583 022664  020104                2$:     CMP     R1,R4                   ;IS SOURCE OPERAND CORRECT
5584 022666  001401                        BEQ     3$                      ;YES GO ON
5585 022670  104001                        ERROR   +1              ;CPU ERROR
                                                                   ;NO GO TO ERROR
5586
5587 022672  005203                3$:     INC     R3                      ;POINT TO NEXT WORD
5588 022674  021227  177777                CMP     (R2),#177777            ;ARE WE DONE
5589 022700  001361                        BNE     TA116                   ;NO GO TO TA116
5590 022702  000207                        RTS     PC                      ;RETURN
5591
5592 022704     377                TE116A: .BYTE   377
5593 022705     000                        .BYTE   0
5594 022706     252                        .BYTE   252
5595 022707     125                        .BYTE   125
5596 022710  030357                TE116B: .WORD   30357
5597 022712  030000                        .WORD   30000
5598 022714  030252                        .WORD   30252
5599 022716  030105                        .WORD   30105
5600 022720  177777                        .WORD   177777
5601 022722  140017                TE116C: .WORD   140017
5602 022724  140000                        .WORD   140000
5603 022726  140012                        .WORD   140012
5604 022730  140005                        .WORD   140005
5605 022732  177777                TE116D: .WORD   177777
5606 022734  177400                        .WORD   177400
5607 022736  177652                        .WORD   177652
5608 022740  177525                        .WORD   177525
5609
5610 022742                        FIN116:
5611
5621 022742                        TE117:
5622                               ;       TEST MFPT (MOVE FROM PROCESSOR TYPE)
5623 022742  013746  000010                MOV     @#10,-(SP)              ;SAVE VECTOR
5624 022746  012737  023054  000010        MOV     #TE117A,@#10            ;SETUP VECTOR TO HANDLE POSSIBLE ILLEGAL INST TRAP
5625 022754  012700  177777                MOV     #177777,R0              ;INIT R0
5626 022760  012737  030000  177776        MOV     #30000,@#177776         ;SETUP PSW
5627 022766  000007                        .WORD   7                       ; TEST INSTRUCTION
5628 022770  022737  030000  177776        CMP     #30000,@#177776         ;IS PSW CORRECT
5629 023776  001401                        BEQ     1$                      ;YES GO ON
5630 023000  104001                        ERROR   +1              ;CPU ERROR
                                                                   ;NO GO TO ERROR
5631
5632 023002  020027  000005          1$:   CMP     R0,#5                   ;IS R0 CORRECT
5633 023006  001401                        BEQ     2$                      ;YES GO ON
5634 023010  104001                        ERROR   +1              ;CPU ERROR
                                                                   ;NO GO TO ERROR
5635
5636 023012  012700  177777          2$:   MOV     #177777,R0              ;INIT R0
```

C9


```
********** DOUBLE OPERAND TESTS *****************

5637 023016  000277                        SCC                             ;SET ALL CC BITS
5638 023020  000007                        .WORD   7                       ; TEST INSTRUCTION
5639 023022  022737  030017  177776        CMP     #30017,@#177776         ;IS PSW CORRECT
5640 023030  001401                        BEQ     3$                      ;YES GO ON
5641 023032  104001                        ERROR   .+1             ;CPU ERROR
                                           ;NO GO TO ERROR
5642
5643 023034  020027  000005          3$:   CMP     R0,#5                   ;IS R0 CORRECT
5644 023040  001401                        BEQ     4$                      ;YES GO ON
5645 023042  104001                        ERROR   .+1             ;CPU ERROR
                                           ;NO GO TO ERROR
5646
5647 023044  012637  000010          4$:   MOV     (SP)+,@#10              ;RESTORE VECTOR
5648
5649 023050  000167  000002                JMP     FIN117
5650
5651 023054  104001                TE117A: ERROR   .+1             ;CPU ERROR
                                           ;GO TO ERROR IF TRAP TAKES PLACE
5652
5653
5654 023056  000240                FIN117: NOP
5655
5667 023060                        TE120:
5668                               ;       TEST HALT (NOT KERNEL MODE)
5669 023060  005037  177766                CLR     @#177766                ;INIT CPU ERROR REG
5670 023064  005037  177776                CLR     @#177776                ;INIT PSW-SET KERNEL MODE
5671 023070  013746  000004                MOV     @#4,-(SP)               ;SAVE VECTOR
5672 023074  013746  000006                MOV     @#6,-(SP)               ;SAVE VECTOR
5673 023100  012737  023134  000004        MOV     #TE120A,@#4             ;SET UP VECTOR TO HANDLE ILLEGAL HALT
5674 023106  005037  000006                CLR     @#6                     ;SET UP VECTOR TO COME BACK IN KERNEL MODE
5675 023112  012767  140000  154656        MOV     #140000,PS              ;SET IN USER MODE
5676 023120  012706  000600                MOV     #600,R6                 ;INITIALIZE THE USER STACK POINTER
5677 023124  000000                        HALT                            ; TEST INSTRUCTION
5678 023126  104001                PROCNT: ERROR   .+1             ;CPU ERROR
                                           ;IF NOTHING HAPPENED GO TO ERROR
5679
5680 023130  000167  000044                JMP     FIN120                  ;
5681                               ;
5682                               ;
5683 023134  022737  030000  177776 TE120A: CMP     #30000,@#177776         ;IS PSW CORRECT/PREVIOUS MODE   USER?
5684 023142  001401                        BEQ     1$                      ;YES GO ON
5685 023144  104001                        ERROR   .+1             ;CPU ERROR
                                           ;NO GO TO ERROR
5686
5687 023146  022737  000200  177766  1$:   CMP     #200,@#177766           ; TEST CPU ERROR REGISTER
5688 023154  001401                        BEQ     2$                      ;YES GO ON
5689 023156  104001                        ERROR   .+1             ;CPU ERROR
                                           ;NO GO TO ERROR
5690
5691 023160  022627  023126          2$:   CMP     (SP)+,#PROCNT           ;DOES STACK CONTAIN CORRECT PC
5692 023164  001401                        BEQ     3$                      ;YES GO ON
5693 023166  104001                        ERROR   .+1             ;CPU ERROR
                                           ;NO GO TO ERROR
5694
5695 023170  022627  140000          3$:   CMP     (SP)+,#140000           ;DOES STACK CONTAIN CORRECT PSW
5696 023174  001401                        BEQ     FIN120                  ;YES GO ON
5697 023176  104001                        ERROR   .+1             ;CPU ERROR
                                           ;NO GO TO ERROR
5698
5699 023200  012637  000006        FIN120: MOV     (SP)+,@#6               ;RESTORE VECTOR
5700 023204  012637  000004                MOV     (SP)+,@#4               ;RESTORE VECTOR
5701
5702                               ;
5705 023210                        TE121:
5706                               ;       TEST RESET
```

********** DOUBLE OPERAND TESTS **********

```
5707                                    ;       CMPB    #APTENV,$ENV            ;ARE WE IN APT MODE?
5708                                    ;       BNE     1$                      ;IF NOT: DO THIS TEST
5709 023210  005767  155772                      TST     $PASS                   ;FIRST PASS??
5710 023214  001402                              BEQ     1$                      ;IF YES, DO IT
5711 023216  000167  000622                      JMP     FIN122                  ;ELSE SKIP THIS TEST BECAUSE RESETS
5712                                                                             ;SCREW UP THE APT MONITOR.
5713 023222  012737  030340  177776  1$:         MOV     #30340,@#177776         ;SETUP PSW TO KERNEL MODE
5714 023230  012737  160000  177572              MOV     #160000,@#177572        ;SETUP MMR0
5715 023236  012737  000077  172516              MOV     #77,@#172516            ;SETUP MMR3
5716 023244  005037  177772                      CLR     @#177772                ;CLEAR PIRQ
5717 023250  023727  177772  000000              CMP     @#177772,#0             ;IS PIRQ CORRECT
5718 023256  001401                              BEQ     C121A                   ;YES GO ON
5719 023260  104001                              ERROR   +1              ;CPU ERROR
5720                                                                     ;NO GO TO ERROR
5721 023262  012737  025000  177772  C121A:      MOV     #25000,@#177772         ;MOVE AN ALTERNATING PATTERN TO PIRQ
5722 023270  022737  025252  177772              CMP     #25252,@#177772         ;IS PIRQ C ?RECT
5723 023276  001401                              BEQ     C121B                   ;YES GO ON
5724 023300  104001                              ERROR   +1              ;CPU ERROR
5725                                                                     ;NO GO TO ERROR
5726 023302  012737  077000  177772  C121B:      MOV     #77000,@#177772         ;SETUP PIRQ
5727 023310  022737  077314  177772              CMP     #77314,@#177772         ;IS PIRQ CORRECT
5728 023316  001401                              BEQ     C121C                   ;YES GO ON
5729 023320  104001                              ERROR   +1              ;CPU ERROR
5730                                                                     ;NO GO TO ERROR
5731 023322  000277                      C121C:  SCC                             ;SET ALL CC BITS
5732 023324  000005                              RESET                           ; TEST INSTRUCTION
5733 023326  022737  030357  177776              CMP     #30357,@#177776         ;IS PSW CORRECT
5734 023334  001401                              BEQ     1$                      ;YES GO ON
5735 023336  104001                              ERROR   +1              ;CPU ERROR
5736                                                                     ;NO GO TO ERROR
5737 023340  013701  177572          1$:         MOV     @#SR0,R1                ;SAVE SR0 IN R1.
5738 023344  042701  000176                      BIC     #176,R1                 ;STRIP OFF UNDEFINED BITS 1-6 FROM MMR0
5739 023350  022701  000000                      CMP     #0,R1                   ;IS MMR0 CORRECT
5740 023354  001401                              BEQ     2$                      ;YES GO ON
5741 023356  104001                              ERROR   +1              ;CPU ERROR
5742                                                                     ;NO GO TO ERROR
5743 023360  022737  000000  172516  2$:         CMP     #0,@#172516             ;IS MMR3 CORRECT
5744 023366  001401                              BEQ     3$                      ;YES GO ON
5745 023370  104001                              ERROR   +1              ;CPU ERROR
5746                                                                     ;NO GO TO ERROR
5747 023372  022737  000000  177772  3$:         CMP     #0,@#177772             ;IS PIRQ CORRECT
5748 023400  001401                              BEQ     4$                      ;YES GO ON
5749 023402  104001                              ERROR   +1              ;CPU ERROR
5750                                                                     ;NO GO TO ERROR
5751
5752 023404  013702  000004          4$:         MOV     @#4,R2                  ;SAVE LOC 4 IN R2
5753 023410  013703  000006                      MOV     @#6,R3                  ;SAVE LOC 6 IN R3
5754 023414  012737  023552  000004              MOV     #8$,@#4                 ;SETUP VECTORS IN CASE AN ERROR CAUSES
5755                                                                             ;A HALT TO OCCUR IN USER MODE.
5756 023422  012737  000340  000006              MOV     #340,@#6                ;THIS SETS KERNEL MODE, AND PREVENTS PRIQ
5757                                                                             ;INTERRUPTS FROM TRASHING THE STACK IF A
5758                                                                             ;USER MODE HALT OCCURS.
5759 023430  012737  140340  177776              MOV     #140340,@#177776        ;SETUP PSW TO USER MODE
5760 023436  012737  160000  177572              MOV     #160000,@#177572        ;SETUP MMR0
5761 023444  012737  000077  172516              MOV     #77,@#172516            ;SETUP MMR3
5762 023452  012737  077000  177772              MOV     #77000,@#177772         ;SETUP PIRQ
5763 023460  000277                              SCC                             ;SET ALL CC BITS
```

```
********** DOUBLE OPERAND TESTS *****************

 5764 023462  000005                          RESET                           ; TEST INSTRUCTION
 5765 023464  022737  140357  177776          CMP     #140357,@#177776        ;IS PSW CORRECT
 5766 023472  001402                          BEQ     5$                      ;YES GO ON
 5767 023474  000000                          HALT                            ;USER MODE HALT; WILL TRAP TO LOC 4
 5768 023476  104001                          ERROR   .+1             ;CPU ERROR
                                                                      ;NO GO TO ERROR
 5769
 5770 023500  013701  177572          5$:     MOV     @#177572,R1             ;SAVE MMR0 IN R1
 5771 023504  042701  000176                  BIC     #176,R1                 ;CLEAR BITS 1 6 FROM MMR0
 5772 023510  022701  160000                  CMP     #160000,R1              ;IS MMR0 CORRECT
 5773 023514  001402                          BEQ     6$                      ;YES GO ON
 5774 023516  000000                          HALT                            ;USER MODE HALT; WILL TRAP TO LOC 4
 5775 023520  104001                          ERROR   .+1             ;CPU ERROR
                                                                      ;NO GO TO ERROR
 5776
 5777 023522  022737  000077  172516  6$:     CMP     #77,@#172516            ;IS MMR3 CORRECT
 5778 023530  001402                          BEQ     7$                      ;YES GO ON
 5779 023532  000000                          HALT                            ;USER MODE HALT; WILL TRAP TO LOC 4
 5780 023534  104001                          ERROR   .+1             ;CPU ERROR
                                                                      ;NO GO TO ERROR
 5781
 5782 023536  022737  077314  177772  7$:     CMP     #77314,@#177772         ;IS PIRQ CORRECT
 5783 023544  001403                          BEQ     9$                      ;YES GO ON
 5784 023546  000000                          HALT                            ;USER MODE HALT; WILL TRAP TO LOC 4
 5785 023550  104001                          ERROR   .+1             ;CPU ERROR
                                                                      ;NO GO TO ERROR
 5786
 5787 023552  000002                  8$:     RTI                             ;USER MODE HALT OCCURRED; GO TO ERROR.
 5788
 5789 023554  005037  177772          9$:     CLR     @#177772                ;CLEAR PIRQ
 5790 023560  005037  177776                  CLR     @#177776                ;CLEAR PSW
 5791 023564  010237  000004                  MOV     R2,@#4                  ;RESTORE VECTORS TO PREVIOUS STATE
 5792 023570  010337  000006                  MOV     R3,@#6                  ;
 5793 023574                          FIN121:
 5794                                 ;
 5797 023574                          TE122:
 5798                                 ;       TEST SPL (SET PRIORITY LEVEL)
 5799
 5800 023574  012705  000010                  MOV     #8.,R5                  ;INIT COUNTER
 5801 023600  012701  024024                  MOV     #T1228,R1               ;SETUP POINTER TO DATA
 5802 023604  012737  030000  177776  1$:     MOV     #30000,@#177776         ;INIT PSW
 5803 023612  004767  000054                  JSR     PC,T122A                ; TEST INSTRUCTION
 5804 023616  022137  177776                  CMP     (R1)+,@#177776          ;IS PSW CORRECT
 5805 023622  001401                          BEQ     2$                      ;YES GO ON
 5806 023624  104001                          ERROR   .+1             ;CPU ERROR
                                                                      ;NO GO TO ERROR
 5807
 5808 023626  077512                  2$:     SOB     R5,1$                   ;REPEAT UNTIL ALL CASES ARE TESTED
 5809
 5810
 5811 023630  012705  000010                  MOV     #8.,R5                  ;INIT COUNTER
 5812 023634  012737  140000  177776  3$:     MOV     #140000,@#177776        ;SETUP PSW TO USER MODE
 5813 023642  012706  000600                  MOV     #600,R6                 ;SETUP USER STACK
 5814 023646  004767  000020                  JSR     PC,T122A                ; TEST INSTRUCTION
 5815 023652  022737  140017  177776          CMP     #140017,@#177776        ;IS PSW CORRECT
 5816 023660  001401                          BEQ     4$                      ;YES GO ON
 5817 023662  104001                          ERROR   .+1             ;CPU ERROR
                                                                      ;NO GO TO ERROR
 5818
 5819 023664  077515                  4$:     SOB     R5,3$                   ;REPEAT UNTIL ALL CASES ARE TESTED
 5820
 5821
 5822 023666  000167  000152                  JMP     FIN122
```

F9

********** DOUBLE OPERAND TESTS **********

```
5823
5824 023672  020527  000010          ;T122A:  CMP     R5,#8.          ;FIND OUT WHAT COUNTER IS
5825 023676  001003                   BNE     1$              ;IF NOT PRIORITY 0 GO TO 1$
5826 023700  000277                   SCC                     ;SET ALL CC BITS
5827 023702  000230                   SPL     0               ;SET PRIORITY TO 0
5828 023704  000446                   BR      8$              ;RETURN
5829 023706  020527  000007          1$:      CMP     R5,#7           ;FIND OUT WHAT COUNTER IS
5830 023712  001003                   BNE     2$              ;IF NOT PRIORITY 1 GO TO 2$
5831 023714  000277                   SCC                     ;SET ALL CC BITS
5832 023716  000231                   SPL     1               ;SET PRIORITY TO 1
5833 023720  000440                   BR      8$              ;RETURN
5834 023722  020527  000006          2$:      CMP     R5,#6           ;FIND OUT WHAT COUNTER IS
5835 023726  001003                   BNE     3$              ;IF NOT PRIORITY 2 GO TO 3$
5836 023730  000277                   SCC                     ;SET ALL CC BITS
5837 023732  000232                   SPL     2               ;SET PRIORITY TO 2
5838 023734  000432                   BR      8$              ;RETURN
5839 023736  020527  000005          3$:      CMP     R5,#5           ;FIND OUT WHAT COUNTER IS
5840 023742  001003                   BNE     4$              ;IF NOT PRIORITY 3 GO  O 4$
5841 023744  000277                   SCC                     ;SET ALL CC BITS
5842 023746  000233                   SPL     3               ;SET PRIORITY TO 3
5843 023750  000424                   BR      8$              ;RETURN
5844 023752  020527  000004          4$:      CMP     R5,#4           ;FIND OUT WHAT COUNTER IS
5845 023756  001003                   BNE     5$              ;IF NOT PRIORITY 4 GO TO 5$
5846 023760  000277                   SCC                     ;SET ALL CC BITS
5847 023762  000234                   SPL     4               ;SET PRIORITY TO 4
5848 023764  000416                   BR      8$              ;RETURN
5849 023766  020527  000003          5$:      CMP     R5,#3           ;FIND OUT WHAT COUNTER IS
5850 023772  001003                   BNE     6$              ;IF NOT PRIORITY 5 GO TO 6$
5851 023774  000277                   SCC                     ;SET ALL CC BITS
5852 023776  000235                   SPL     5               ;SET PRIORITY TO 5
5853 024000  000410                   BR      8$              ;RETURN
5854 024002  020527  000002          6$:      CMP     R5,#2           ;FIND OUT WHAT COUNTER IS
5855 024006  001003                   BNE     7$              ;IF NOT PRIORITY 6 GO TO 7$
5856 024010  000277                   SCC                     ;SET ALL CC BITS
5857 024012  000236                   SPL     6               ;SET PRIORITY TO 6
5858 024014  000402                   BR      8$              ;RETURN
5859 024016  000277                  7$:      SCC                     ;SET ALL CC BITS
5860 024020  000237                   SPL     7               ;SET PRIORITY TO 7
5861 024022  000207                  8$:      RTS     PC              ;RETURN
5862
5863 024024  030017                  ;T122B:  .WORD   30017
5864 024026  030057                   .WORD   30057
5865 024030  030117                   .WORD   30117
5866 024032  030157                   .WORD   30157
5867 024034  030217                   .WORD   30217
5868 024036  030257                   .WORD   30257
5869 024040  030317                   .WORD   30317
5870 024042  030357                   .WORD   30357
5871 024044  000240                  FIN122:  NOP
5872
5875 024046                          ;TE123:
5876                                 ;       TEST TSTSET INSTRUCTION (MULTI PROCESSING INST)
5877 024046  005037  177776                  CLR     @#177776        ;INIT PSW
5878 024052  012703  000012                  MOV     #10.,R3         ;INIT COUNTER
5879 024056  012701  000400                  MOV     #400,R1         ;SETUP DESTINATION
5880 024062  012700  024230                  MOV     #T123A,R0       ;SETUP SOURCE
5881 024066  012021                  100$:    MOV     (R0)+,(R1)+     ;RELOCATE TABLES
```

********** DOUBLE OPERAND TESTS **********

```
5882 024070  077302                        SOB     R3,100$                 ;ARE WE DONE
5883 024072  013746  000010                MOV     @#10,-(SP)              ;SAVE VECTOR
5884 024076  012737  024254  000010        MOV     #T123D,@#10             ;SETUP NEW VECTOR
5885 024104  005000                        CLR     R0                      ;INIT R0
5886 024106  012701  000400                MOV     #400,R1                 ;SETUP POINTERS TO TABLES
5887 024112  012702  000410                MOV     #410,R2                 ;
5888 024116  012703  000416                MOV     #416,R3                 ;
5889 024122  010104                        MOV     R1,R4                   ;
5890 024124  012737  030000  177776  1$:   MOV     #30000,@#177776         ;SETUP PSW
5891 024132  000262                        SEV                             ;SET V BIT
5892 024134  007221                        .WORD   7221                    ; TEST INSTRUCTION
5893 024136  022237  177776                CMP     (R2)+,@#177776          ;IS PSW CORRECT
5894 024142  001401                        BEQ     2$                      ;YES GO ON
5895 024144  104001                        ERROR   +1                      ;CPU ERROR
5896                                                                        ;NO GO TO ERROR
5897 024146  020013                2$:     CMP     R0,(R3)                 ;IS R0 CORRECT
5898 024150  001401                        BEQ     3$                      ;YES GO ON
5899 024152  104001                        ERROR   +1                      ;CPU ERROR
5900                                                                        ;NO GO TO ERROR
5901 024154  005204                3$:     INC     R4                      ;SETUP EXPECTED DATA
5902 024156  005204                        INC     R4                      ;
5903 024160  020401                        CMP     R4,R1                   ;IS R1 CORRECT
5904 024162  001401                        BEQ     4$                      ;YES GO ON
5905 024164  104001                        ERROR   +1                      ;CPU ERROR
5906                                                                        ;NO GO TO ERROR
5907 024166  052713  000001        4$:     BIS     #1,(R3)                 ;SETUP EXPECTED DATA
5908 024172  022341                        CMP     (R3)+,-(R1)             ;IS TEST LOCATION CORRECT
5909 024174  001401                        BEQ     5$                      ;YES GO ON
5910 024176  104001                        ERROR   +1                      ;CPU ERROR
5911                                                                        ;NO GO TO ERROR
5912 024200  005201                5$:     INC     R1                      ;POINT TO NEXT TEST LOCATION
5913 024202  005201                        INC     R1                      ;
5914 024204  021127  177777                CMP     (R1),#177777            ;ARE WE DONE
5915 024210  001345                        BNE     1$                      ;NO GO TO 1$
5916 024212  012737  024256  000010        MOV     #T123E,@#10             ;SETUP NEW VECTOR
5917 024220  007201                        .WORD   7201                    ; TEST INSTRUCTION ILLEGAL MODE
5918 024222  104001                        ERROR   +1                      ;CPU ERROR
5919                                                                        ;GO TO ERROR IF DIDN'T TRAP
5920 024224  000167  000032                JMP     T123F
5921                                ;
5922                                ;
5923 024230  167604                T123A:  .WORD   167604
5924 024232  000000                        .WORD   0
5925 024234  000001                        .WORD   1
5926 024236  177777                        .WORD   177777
5927 024240  030010                T123B:  .WORD   30010
5928 024242  030004                        .WORD   30004
5929 024244  030001                        .WORD   30001
5930 024246  167604                T123C:  .WORD   167604
5931 024250  000000                        .WORD   0
5932 024252  000001                        .WORD   1
5933 024254  104001                T123D:  ERROR   1                       ;CPU ERROR
5934                                                                        ;GO TO ERROR IF TRAPPED
5935 024256  005726                T123E:  TST     (SP)+                   ;CLEAN UP STACK
5936 024260  005726                        TST     (SP)+                   ;
5937 024262  012637  000010        T123F:  MOV     (SP)+,@#10              ;RESTORE VECTOR
5938
```

********** DOUBLE OPERAND TESTS **********

```
5939                                            ;
5942 024266                                     ;T124:
5943                                            ;       TEST WRTLCK (WRITE LOCK MULTI PROCESSING INST)
5944 024266  005037  177776                      ;       CLR     @#177776                ;INIT PSW
5945 024272  012703  000012                      ;       MOV     #10.,R3                 ;INIT COUNTER
5946 024276  012701  000400                      ;       MOV     #400,R1                 ;SETUP DESTINATION
5947 024302  012700  024462                      ;       MOV     #T124A,R0               ;SETUP SOURCE
5948 024306  012021                              ;100$:  MOV     (R0)+,(R1)+             ;RELOCATE TABLES
5949 024310  077302                              ;       SOB     R3,100$                 ;ARE WE DONE
5950 024312  013746  000010                      ;       MOV     @#10,-(SP)              ;SAVE VECTOR
5951 024316  012737  024506  000010              ;       MOV     #T124D,@#10             ;SETUP NEW VECTOR
5952 024324  012701  000400                      ;       MOV     #400,R1                 ;SETUP POINTERS TO TABLES
5953 024330  012702  000410                      ;       MOV     #410,R2                 ;
5954 024334  012703  000416                      ;       MOV     #416,R3                 ;
5955 024340  010204                              ;       MOV     R2,R4                   ;
5956 024342  012737  030000  177776              ;1$     MOV     #30000,@#177776         ;SETUP PSW
5957 024350  011100                              ;       MOV     (R1),R0                 ;SETUP R0
5958 024352  020327  000416                      ;       CMP     R3,#416                 ;IS THIS THE FIRST TEST CASE
5959 024356  001401                              ;       BEQ     2$                      ;YES GO TO 2$
5960 024360  000402                              ;       BR      3$                      ;NO GO TO 3$
5961 024362  000261                              ;2$:    SEC                             ;SET C BIT
5962 024364  000401                              ;       BR      4$                      ;
5963 024366  000241                              ;3$:    CLC                             ;CLEAR C BIT
5964 024370  000262                              ;4$:    SEV                             ;SET V BIT
5965 024372  007322                              ;       .WORD   7322                    ; TEST INSTRUCTION
5966 024374  022337  177776                      ;       CMP     (R3)+,@#177776          ;IS PSW CORRECT
5967 024400  001401                              ;       BEQ     5$                      ;YES GO ON
5968 024402  104001                              ;       ERROR   +1              ;CPU ERROR
5969                                            ;                               ;NO GO TO ERROR
5970 024404  021100                              ;5$:    CMP     (R1),R0                 ;IS R0 CORRECT
5971 024406  001401                              ;       BEQ     6$                      ;YES GO ON
5972 024410  104001                              ;       ERROR   +1              ;CPU ERROR
5973                                            ;                               ;NO GO TO ERROR
5974 024412  005204                              ;6$:    INC     R4                      ;SETUP EXPECTED DATA
5975 024414  005204                              ;       INC     R4                      ;
5976 024416  020204                              ;       CMP     R2,R4                   ;IS R2 CORRECT
5977 024420  001401                              ;       BEQ     7$                      ;YES GO ON
5978 024422  104001                              ;       ERROR   +1              ;CPU ERROR
5979                                            ;                               ;NO GO TO ERROR
5980 024424  022142                              ;7$:    CMP     (R1)+,-(R2)             ;IS TEST LOCATION CORRECT
5981 024426  001401                              ;       BEQ     8$                      ;YES GO ON
5982 024430  104001                              ;       ERROR   +1              ;CPU ERROR
5983                                            ;                               ;NO GO TO ERROR
5984 024432  005202                              ;8$:    INC     R2                      ;POINT TO NEXT TEST LOCATION
5985 024434  005202                              ;       INC     R2                      ;
5986 024436  021127  177777                      ;       CMP     (R1),#177777            ;ARE WE DONE
5987 024442  001337                              ;       BNE     1$                      ;NO GO TO 1$
5988 024444  012737  024510  000010              ;       MOV     #T124E,@#10             ;SETUP NEW VECTOR
5989 024452  007302                              ;       .WORD   7302                    ; TEST INSTRUCTION ILLEGAL MODE
5990 024454  104001                              ;       ERROR   +1              ;CPU ERROR
5991                                            ;                               ;GO TO ERROR IF DIDN'T TRAP
5992 024456  000167  000032                      ;       JMP     T124F
5993                                            ;
5994                                            ;
5995 024462  167604                              ;T124A: .WORD   167604
5996 024464  000000                              ;       .WORD   0
5997 024466  000001                              ;       .WORD   1
```

```
********** DOUBLE OPERAND TESTS *****************

 5998 024470  177777                    .WORD   177777
 5999 024472  177777            T124B:  .WORD   177777
 6000 024474  177777                    .WORD   177777
 6001 024476  177777                    .WORD   177777
 6002 024500  030011            T124C:  .WORD   30011
 6003 024502  030004                    .WORD   30004
 6004 024504  030000                    .WORD   30000
 6005 024506  104001            T124D:  ERROR   +1              ;CPU ERROR
 6006                                                           ;GO TO ERROR IF TRAPPED
 6007 024510     726            T124E:  TST     (SP)+                   ;CLEAN UP STACK
 6008 024512  00  26                    TST     (SP)+                   ;
 6009 024514     ,37  000010    T124F:  MOV     (SP)+,@#10              ;RESTORE VECTOR
 6010
 6011                           ;
 6014 024520                    TE125:
 6015                           ;       TEST MUL (MULTIPLY INST)
 6016 024520  005037  177776            CLR     @#177776                ;INIT PS
 6017 024524  012701  024760            MOV     #TE125A,R1              ;SETUP POINTERS TO TABLES
 6018
 6019 024530  010137  114146    1$:     MOV     R1,@#EXPDAT             ;
 6020 024534  062737  000002  114146    ADD     #2,@#EXPDAT             ;POINT TO SOURCE
 6021 024542  012703  122222            MOV     #122222,R3              ;INIT R3 TO A KNOWN STATE
 6022 024546  011102                    MOV     (R1),R2         ;INIT DESTINATION REG
 6023 024550  000277                    SCC                             ;SET ALL CC BITS
 6024 024552  070261  000002            MUL     2(R1),R2                ; TEST INSTRUCTION
 6025 024556  026137  000004  177776    CMP     4(R1),@#177776          ;IS PS CORRECT
 6026 024564  001401                    BEQ     2$                      ;YES GO ON
 6027 024566  104001                    ERROR   +1              ;CPU ERROR
 6028                                                           ;NO GO TO ERROR
 6029 024570  026103  000006    2$:     CMP     6(R1),R3                ;IS R3 CORRECT
 6030 024574  001401                    BEQ     3$                      ;YES GO ON
 6031 024576  104001                    ERROR   +1              ;CPU ERROR
 6032                                                           ;NO GO TO ERROR
 6033 024600  026102  000010    3$:     CMP     10(R1),R2               ;IS R2 CORRECT
 6034 024604  001401                    BEQ     4$                      ;YES GO ON
 6035 024606  104001                    ERROR   +1              ;CPU ERROR
 6036                                                           ;NO GO TO ERROR
 6037 024610  026177  000002  067330    4$:     CMP     2(R1),@EXPDAT           ;IS SOURCE LOCATION OK
 6038 024616  001401                    BEQ     5$                      ;YES GO ON
 6039 024620  104001                    ERROR   +1              ;CPU ERROR
 6040                                                           ;NO GO TO ERROR
 6041 024622  062701  000012    5$:     ADD     #12,R1                  ;GO TO NEXT TEST
 6042 024626  020127  025206            CMP     R1,#FIN125              ;ARE WE FINISHED
 6043 024632  001336                    BNE     1$                      ;NO GO TO 1$
 6044
 6045                           ;
 6046                           ;SECOND PART
 6047                           ;USING ODD REGISTER
 6048                           ;
 6049 024634  012701  024760    6$:     MOV     #TE125A,R1              ;SETUP POINTERS TO TABLES
 6050 024640                    7$:
 6051 024640  010102                    MOV     R1,R2                   ;
 6052 024642  012706  001000            MOV     #STBOT,R6               ;INIT R6 TO A KNOWN STATE
 6053 024646  012704  000004            MOV     #4,R4                   ;SETUP R4 VALUE
 6054 024652  011105                    MOV     (R1),R5         ;INIT DESTINATION REG
 6055 024654  000277                    SCC                             ;SET ALL CC BITS
 6056 024656  070561  000002            MUL     2(R1),R5                ; TEST INSTRUCTION
```

Q 0113

```
********** DOUBLE OPERAND TESTS *****************

6057 024662  026137  000004  177776          CMP     4(R1),@#177776          ;IS PS CORRECT
6058 024670  001401                          BEQ     8$                      ;YES GO ON
6059 024672  104001                          ERROR   .1                  ;CPU ERROR
6060                                                                     ;NO GO TO ERROR
6061 024674  026105  000006          6$:     CMP     6(R1),R5                ;IS R5 CORRECT
6062 024700  001401                          BEQ     9$                      ;YES GO ON
6063 024702  104001                          ERROR   .1                  ;CPU ERROR
6064                                                                     ;NO GO TO ERROR
6065 024704  020627  001000          9$:     CMP     R6,#STBOT               ;IS R6 CORRECT
6066 024710  001403                          BEQ     10$                     ;YES GO ON
6067 024712  012706  001000                  MOV     #STBOT,R6               ;RESTORE SP
6068 024716  104001                          ERROR   .1                  ;CPU ERROR
6069                                                                     ;NO GO TO ERROR
6070 024720  005722          10$:    TST     (R2)+                   ;POINT TO SOURCE OPERAND
6071 024722  021261  000002                  CMP     (R2),2(R1)              ;IS SOURCE LOCATION OK
6072 024726  001401                          BEQ     11$                     ;YES GO ON
6073 024730  104001                          ERROR   .1                  ;CPU ERROR
6074                                                                     ;NO GO TO ERROR
6075 024732  020427  000004          11$:    CMP     R4,#4                   ;IS R4 CORRECT
6076 024736  001401                          BEQ     12$                     ;YES GO ON
6077 024740  104001                          ERROR   .1                  ;CPU ERROR
6078                                                                     ;NO GO TO ERROR
6079 024742  062701  000012          12$:    ADD     #12,R1                  ;
6080 024746  020127  025206                  CMP     R1,#FIN125              ;ARE WE FINISHED
6081 024752  001332                          BNE     7$                      ;NO GO TO 7$
6082
6083
6084 024754  000167  000226                  JMP     FIN125
6085                                 ;
6086                                 ;
6087 024760  177777          TE125A: .WORD   177777                  ;MULTIPLICAND
6088 024762  177777                  .WORD   177777                  ;MULTIPLIER
6089 024764  000000                  .WORD   0
6090 024766  000001                  .WORD   1
6091 024770  000000                  .WORD   0
6092
6093 024772  006772                  .WORD   6772                    ;MULTIPLICAND
6094 024774  100000                  .WORD   100000                  ;MULTIPLIER
6095 024776  000011                  .WORD   11
6096 025000  000000                  .WORD   0
6097 025002  174403                   WORD   174403
6098
6099 025004  177777                  .WORD   177777                  ;MULTIPLICAND
6100 025006  077777                  .WORD   77777                   ;MULTIPLIER
6101 025010  000010                  .WORD   10
6102 025012  100001                  .WORD   100001
6103 025014  177777                  .WORD   177777
6104
6105 025016  077777                  .WORD   77777                   ;MULTIPLICAND
6106 025020  000456                  .WORD   456                     ;MULTIPLIER
6107 025022  000001                  .WORD   1
6108 025024  177322                  .WORD   177322
6109 025026  000226                  .WORD   226
6110
6111 025030  173210                  .WORD   173210                  ;MULTIPLICAND
6112 025032  000000                  .WORD   0                       ;MULTIPLIER
6113 025034  000004                  .WORD   4
```

********** DOUBLE OPERAND TESTS **********

```
 6114 025036  000000                    .WORD   0
 6115 025040  000000                    .WORD   0
 6116
 6117 025042  000000                    .WORD   0               ;MULTIPLICAND
 6118 025044  003251                    .WORD   3251            ;MULTIPLIER
 6119 025046  000004                    .WORD   4
 6120 025050  000000                    .WORD   0
 6121 025052  000000                    .WORD   0
 6122
 6123 025054  000000                    .WORD   0               ;MULTIPLICAND
 6124 025056  000000                    .WORD   0               ;MULTIPLIER
 6125 025060  000004                    .WORD   4
 6126 025062  000000                    .WORD   0
 6127 025064  000000                    .WORD   0
 6128
 6129 025066  100000                    .WORD   100000          ;MULTIPLICAND
 6130 025070  000001                    .WORD   1               ;MULTIPLIER
 6131 025072  000010                    .WORD   10
 6132 025074  100000                    .WORD   100000
 6133 025076  177777                    .WORD   177777
 6134
 6135 025100  077777                    .WORD   77777           ;MULTIPLICAND
 6136 025102  000001                    .WORD   1               ;MULTIPLIER
 6137 025104  000000                    .WORD   0
 6138 025106  077777                    .WORD   77777
 6139 025110  000000                    .WORD   0
 6140
 6141 025112  000010                    .WORD   10              ;MULTIPLICAND
 6142 025114  010000                    .WORD   10000           ;MULTIPLIER
 6143 025116  000001                    .WORD   1
 6144 025120  100000                    .WORD   100000
 6145 025122  000000                    .WORD   0
 6146
 6147 025124  001452                    .WORD   1452            ;MULTIPLICAND
 6148 025126  034527                    .WORD   34527           ;MULTIPLIER
 6149 025130  000001                    .WORD   1
 6150 025132  066506                    .WORD   66506
 6151 025134  000265                    .WORD   265
 6152
 6153 025136  000007                    .WORD   7               ;MULTIPLICAND
 6154 025140  000400                    .WORD   400             ;MULTIPLIER
 6155 025142  000000                    .WORD   0
 6156 025144  003400                    .WORD   3400
 6157 025146  000000                    .WORD   0
 6158
 6159 025150  000002                    .WORD   2               ;MULTIPLICAND
 6160 025152  100000                    .WORD   100000          ;MULTIPLIER
 6161 025154  000011                    .WORD   11
 6162 025156  000000                    .WORD   0
 6163 025160  177777                    .WORD   177777
 6164
 6165 025162  100000                    .WORD   100000          ;MULTIPLICAND
 6166 025164  077777                    .WORD   77777           ;MULTIPLIER
 6167 025166  000011                    .WORD   11
 6168 025170  100000                    .WORD   100000
 6169 025172  140000                    .WORD   140000
 6170
```

```
********** DOUBLE OPERAND TESTS *****************

6171 025174  000001                        .WORD   1                       ;MULTIPLICAND
6172 025176  177777                        .WORD   177777                  ;MULTIPLIER
6173 025200  000010                        .WORD   10
6174 025202  177777                        .WORD   177777
6175 025204  177777                        .WORD   177777
6176 025206                        FIN125:
6177                               ;
6180 025206                        TE126:
6181                               ;       TEST DIV (DIVIDE INST)
6182 025206  005037  177776                CLR     @#177776                ;INIT PSW
6183 025212  005006                        CLR     R6                      ;INIT SP
6184 025214  013705  000000                MOV     @#0,R5                  ;SAVE VECTORS
6185 025220  013701  000002                MOV     @#2,R1                  ;
6186 025224  012737  000137  000000        MOV     #137,@#0                ;SETUP NEW VECTORS
6187 025232  012737  025254  000002        MOV     #TE126A,@#2             ;
6188 025240  000277                        SCC                             ;SET ALL CC BITS
6189 025242  071627  000002                DIV     #2,R6                   ; TEST INSTRUCTION
6190 025246  012706  001000        A126:   MOV     #STBOT,R6               ;RESTORE SP BEFORE GOING TO ERROR
6191 025252  104001                        ERROR   +1                      ;CPU ERROR
6192                                                                       ;IF R7 ISN'T CORRECT GO TO ERPOR
6193 025254  022737  000000  177776  TE126A: CMP     #0,@#177776             ;IS PS CORRECT
6194 025262  001403                        BEQ     1$                      ;YES GO ON
6195 025264  012706  001000                MOV     #STBOT,R6               ;RESTORE SP BEFORE GOING TO ERROR
6196 025270  104001                        ERROR   +1                      ;CPU ERROR
6197                                                                       ;NO GO TO ERROR
6198 025272  012704  025246        1$:     MOV     #A126,R4                ;SETUP EXPECTED DATA
6199 025276  006204                        ASR     R4                      ;
6200 025300  020406                        CMP     R4,R6                   ;IS R6 CORRECT
6201 025302  001403                        BEQ     2$                      ;YES GO ON
6202 025304  012706  001000                MOV     #STBOT,R6               ;RESTORE SP BEFORE GOING TO ERROR
6203 025310  104001                        ERROR   +1                      ;CPU ERROR
6204                                                                       ;NO GO TO ERROR
6205 025312  010537  000000        2$:     MOV     R5,@#0                  ;RESTORE VECTORS
6206 025316  010137  000002                MOV     R1,@#2                  ;
6207 025322  012706  001000                MOV     #STBOT,R6               ;INIT SP
6208 025326  012702  000006                MOV     #6,R2                   ;INIT GPR 2
6209 025332  012703  000047                MOV     #47,R3                  ;INIT GPR 3
6210 025336  000277                        SCC                             ;SET ALL CC BITS
6211 025340  071302                        DIV     R2,R3                   ; TEST INSTRUCTION
6212 025342  022737  000002  177776        CMP     #2,@#177776             ;IS PS CORRECT
6213 025350  001401                        BEQ     3$                      ;YES GO ON
6214 025352  104001                        ERROR   +1                      ;CPU ERROR
6215                                                                       ;NO GO TO ERROR
6216 025354  022702  000006        3$:     CMP     #6,R2                   ;IS R2 CORRECT
6217 025360  001401                        BEQ     4$                      ;YES GO ON
6218 025362  104001                        ERROR   +1                      ;CPU ERROR
6219                                                                       ;NO GO TO ERROR
6220 025364  022703  000047        4$:     CMP     #47,R3                  ;IS R3 CORRECT
6221 025370  001401                        BEQ     5$                      ;YES GO ON
6222 025372  104001                        ERROR   +1                      ;CPU ERROR
6223                                                                       ;NO GO TO ERROR
6224 025374  005004                5$:     CLR     R4                      ;INIT R4
6225 025376  012705  000004                MOV     #4,R5                   ;INIT R5
6226 025402  000277                        SCC                             ;SET ALL CC BITS
6227 025404  071427  000000                DIV     #0,R4                   ; TEST INSTRUCTION
6228 025410  022737  000007  177776        CMP     #7,@#177776             ;IS PS CORRECT
6229 025416  001401                        BEQ     6$                      ;YES GO ON
```

M9

.Q 0116

********** DOUBLE OPERAND TESTS *****************

```
6230 025420  104001                          ERROR   +1              ;CPU ERROR
6231                                                                 ;NO GO TO ERROR
6232 025422  022704  000000          6$:     CMP     #0,R4                   ;IS R4 CORRECT
6233 025426  001401                          BEQ     7$                      ;YES GO ON
6234 025430  104001                          ERROR   +1              ;CPU ERROR
6235                                                                 ;NO GO TO ERROR
6236 025432  022705  000004          7$:     CMP     #4,R5                   ;IS R5 CORRECT
6237 025436  001401                          BEQ     8$                      ;YES GO ON
6238 025440  104001                          ERROR   +1              ;CPU ERROR
6239                                                                 ;NO GO TO ERROR
6240 025442  012700  000004          8$:     MOV     #4,R0                   ;INIT R0
6241 025446  012705  000010                  MOV     #10,R5                  ;INIT R5
6242 025452  005004                          CLR     R4                      ;INIT R4
6243 025454  000277                          SCC                             ;SET ALL CC BITS
6244 025456  071400                          DIV     R0,R4                   ; TEST INSTRUCTION
6245 025460  022737  000000  177776          CMP     #0,@#177776             ;IS PS CORRECT
6246 025466  001405                          BEQ     9$                      ;YES GO ON
6247 025470  010067  152704                  MOV     R0,400                  ;SAVE R0
6248 025474  104001                          ERROR   +1              ;CPU ERROR
6249                                                                 ;NO GO TO ERROR
6250 025476  016700  152676                  MOV     400,R0                  ;RESTORE R0
6251 025502  022700  000004          9$:     CMP     #4,R0                   ;IS R0 CORRECT
6252 025506  001401                          BEQ     10$                     ;YES GO ON
6253 025510  104001                          ERROR   +1              ;CPU ERROR
6254                                                                 ;NO GO TO ERROR
6255 025512  022704  000002          10$:    CMP     #2,R4                   ;IS R4 CORRECT
6256 025516  001401                          BEQ     11$                     ;YES GO ON
6257 025520  104001                          ERROR   +1              ;CPU ERROR
6258                                                                 ;NO GO TO ERROR
6259 025522  022705  000000          11$:    CMP     #0,R5                   ;IS R5 CORRECT
6260 025526  001401                          BEQ     12$                     ;YES GO ON
6261 025530  104001                          ERROR   +1              ;CPU ERROR
6262                                                                 ;NO GO TO ERROR
6263 025532  012705  000010          12$:    MOV     #10,R5                  ;INIT R5
6264 025536  005004                          CLR     R4                      ;INIT R4
6265 025540  000277                          SCC                             ;SET ALL CC BITS
6266 025542  071427  000003                  DIV     #3,R4                   ; TEST INSTRUCTION
6267 025546  022737  000000  177776          CMP     #0,@#177776             ;IS PS CORRECT
6268 025554  001401                          BEQ     13$                     ;YES GO ON
6269 025556  104001                          ERROR   +1              ;CPU ERROR
6270                                                                 ;NO GO TO ERROR
6271 025560  022704  000002          13$:    CMP     #2,R4                   ;IS R4 CORRECT
6272 025564  001401                          BEQ     14$                     ;YES GO ON
6273 025566  104001                          ERROR   +1              ;CPU ERROR
6274                                                                 ;NO GO TO ERROR
6275 025570  022705  000002          14$:    CMP     #2,R5                   ;IS R5 CORRECT
6276 025574  001401                          BEQ     15$                     ;YES GO ON
6277 025576  104001                          ERROR   +1              ;CPU ERROR
6278                                                                 ;NO GO TO ERROR
6279                                 ;
6280                                 ;
6281                                 ;
6282 025600  012701  025730          15$:    MOV     #TE126B,R1              ;SETUP POINTERS TO TABLES
6283
6284 025604  010137  114146          16$:    MOV     R1,@#EXPDAT             ;SAVE A COPY OF R1
6285 025610  011104                          MOV     (R1),R4         ;INIT R4
6286 025612  016103  000004                  MOV     4(R1),R3                ;SAVE SOURCE
```

```
********** DOUBLE OPERAND TESTS *****************

6287                                    ;
6288 025616  016105  000002            MOV     2(R1),R5                ;INIT R5
6289 025622  000277                    SCC                             ;SET ALL CC BITS
6290 025624  071461  000004            DIV     4(R1),R4                ; TEST INSTRUCTION
6291 025630  026137  000006  177776    CMP     6(R1),@#177776          ;IS PS CORRECT
6292 025636  001401                    BEQ     17$                     ;YES GO ON
6293 025640  104001                    ERROR   +1              ;CPU ERROR
6294                                                            ;NO GO TO ERROR
6295 025642  026105  000010    17$:    CMP     10(R1),R5               ;IS R5 CORRECT
6296 025646  001401                    BEQ     18$                     ;YES GO ON
6297 025650  104001                    ERROR   +1              ;CPU ERROR
6298                                                            ;NO GO TO ERROR
6299 025652  026104  000012    18$:    CMP     12(R1),R4               ;IS R4 CORRECT
6300 025656  001401                    BEQ     19$                     ;YES GO ON
6301 025660  104001                    ERROR   +1              ;CPU ERROR
6302                                                            ;NO GO TO ERROR
6303 025662  023701  114146    19$:    CMP     @#EXPDAT,R1             ;IS R1 CORRECT
6304 025666  001403                    BEQ     20$                     ;YES GO ON
6305 025670  104001                    ERROR   +1              ;CPU ERROR
6306                                                            ;NO GO TO ERROR
6307 025672  013701  114146            MOV     @#EXPDAT,R1             ;RESTORE CORRECT VALUE
6308 025676  026103  000004    20$:    CMP     4(R1),R3                ;IS SOURCE CORRECT
6309 025702  001403                    BEQ     21$                     ;YES GO ON
6310 025704  104001                    ERROR   +1              ;CPU ERROR
6311                                                            ;NO GO TO ERROR
6312 025706  010361  000004            MOV     R3,4(R1)                        ;TRY TO RESTORE CODE
6313 025712  062701  000014    21$:    ADD     #14,R1                  ;POINT TO NEXT LOCATION
6314 025716  021127  000333            CMP     (R1),#333               ;ARE WE DONE
6315 025722  001330                    BNE     16$                     ;NO GO TO 16$
6316
6317
6318 025724  000167  000316            JMP     FIN126
6319                                    ;
6320                                    ;
6321 025730  177777            TE126B: .WORD   177777  ;DIVIDEND
6322 025732  177777                    .WORD   177777  ;INIT R5
6323 025734  177777                    .WORD   177777  ;DIVISOR
6324 025736  000000                    .WORD   0       ;PSW
6325 025740  000000                    .WORD   0       ;R5 RESULT
6326 025742  000001                    .WORD   1       ;R4 RESULT
6327
6328 025744  000000                    .WORD   0       ;DIVIDEND
6329 025746  177777                    .WORD   177777  ;INIT R5
6330 025750  177777                    .WORD   177777  ;DIVISOR
6331 025752  000012                    .WORD   12      ;PSW
6332 025754  177777                    .WORD   177777  ;R5 RESULT
6333 025756  000000                    .WORD   0       ;R4 RESULT
6334
6335 025760  177777                    .WORD   177777  ;DIVIDEND
6336 025762  000000                    .WORD   0       ;INIT R5
6337 025764  177777                    .WORD   177777  ;DIVISOR
6338 025766  000002                    .WORD   2       ;PSW
6339 025770  000000                    .WORD   0       ;R5 RESULT
6340 025772  177777                    .WORD   177777  ;R4 RESULT
6341
6342 025774  000000                    .WORD   0       ;DIVIDEND
6343 025776  007642                    .WORD   7642    ;INIT R5
```

********** DOUBLE OPERAND TESTS **********

```
6344 026000  007643                    .WORD   7643    ;DIVISOR
6345 026002  000004                    .WORD   4       ;PSW
6346 026004  007642                    .WORD   7642    ;R5 RESULT
6347 026006  000000                    .WORD   0       ;R4 RESULT
6348
6349 026010  000000                    .WORD   0       ;DIVIDEND
6350 026012  000137                    .WORD   137     ;INIT R5
6351 026014  177543                    .WORD   177543  ;DIVISOR
6352 026016  000004                    .WORD   4       ;PSW
6353 026020  000137                    .WORD   137     ;R5 RESULT
6354 026022  000000                    .WORD   0       ;R4 RESULT
6355
6356 026024  000000                    .WORD   0       ;DIVIDEND
6357 026026  007643                    .WORD   7643    ;INIT R5
6358 026030  007643                    .WORD   7643    ;DIVISOR
6359 026032  000000                    .WORD   0       ;PSW
6360 026034  000000                    .WORD   0       ;R5 RESULT
6361 026036  000001                    .WORD   1       ;R4 RESULT
6362
6363 026040  100000                    .WORD   100000  ;DIVIDEND
6364 026042  004376                    .WORD   4376    ;INIT R5
6365 026044  010021                    .WORD   10021   ;DIVISOR
6366 026046  000012                    .WORD   12      ;PSW
6367 026050  004376                    .WORD   4376    ;R5 RESULT
6368 026052  100000                    .WORD   100000  ;R4 RESULT
6369
6370 026054  177700                    .WORD   177700  ;DIVIDEND
6371 026056  170033                    .WORD   170033  ;INIT R5
6372 026060  010021                    .WORD   10021   ;DIVISOR
6373 026062  000010                    .WORD   10      ;PSW
6374 026064  171307                    .WORD   171307  ;R5 RESULT
6375 026066  176024                    .WORD   176024  ;R4 RESULT
6376
6377 026070  177700                    .WORD   177700  ;DIVIDEND
6378 026072  170033                    .WORD   170033  ;INIT R5
6379 026074  167757                    .WORD   167757  ;DIVISOR
6380 026076  000000                    .WORD   0       ;PSW
6381 026100  171307                    .WORD   171307  ;R5 RESULT
6382 026102  001754                    .WORD   1754    ;R4 RESULT
6383
6384 026104  000000                    .WORD   0       ;DIVIDEND
6385 026106  177777                    .WORD   177777  ;INIT R5
6386 026110  000001                    .WORD   1       ;DIVISOR
6387 026112  000002                    .WORD   2       ;PSW
6388 026114  177777                    .WORD   177777  ;R5 RESULT
6389 026116  000000                    .WORD   0       ;R4 RESULT
6390
6391 026120  177777                    .WORD   177777  ;DIVIDEND
6392 026122  045716                    .WORD   45716   ;INIT R5
6393 026124  000001                    .WORD   1       ;DIVISOR
6394 026126  000012                    .WORD   12      ;PSW
6395 026130  045716                    .WORD   45716   ;R5 RESULT
6396 026132  177777                    .WORD   177777  ;R4 RESULT
6397
6398 026134  000000                    .WORD   0       ;DIVIDEND
6399 026136  000002                    .WORD   2       ;INIT R5
6400 026140  177770                    .WORD   177770  ;DIVISOR
```

C10

********** DOUBLE OPERAND TESTS **********

```
6401 026142  000004                        .WORD   4         ;PSW
6402 026144  000002                         WORD   2         ;R5 RESULT
6403 026146  000000                        .WORD   0         ;R4 RESULT
6404
6405 026150  177777                        .WORD   177777    ;DIVIDEND
6406 026152  177776                        .WORD   177776    ;INIT R5
6407 026154  000010                        .WORD   10        ;DIVISOR
6408 026156  000004                        .WORD   4         ;PSW
6409 026160  177776                        .WORD   177776    ;R5 RESULT
6410 026162  000000                        .WORD   0         ;R4 RESULT
6411
6412 026164  000001                        .WORD   1         ;D.VIDEND
6413 026166  177777                        .WORD   177777    ;INIT R5
6414 026170  000001                        .WORD   1         ;DIVISOR
6415 026172  000002                        .WORD   2         ;PSW
6416 026174  177777                        .WORD   177777    ;R5 RESULT
6417 026176  000001                        .WORD   1         ;R4 RESULT
6418
6419 026200  000001                        .WORD   1         ;DIVIDEND
6420 026202  000000                        .WORD   0         ;INIT R5
6421 026204  000002                        .WORD   2         ;DIVISOR
6422 026206  000002                        .WORD   2         ;PSW
6423 026210  000000                        .WORD   0         ;R5 RESULT
6424 026212  000001                        .WORD   1         ;R4 RESULT
6425
6426 026214  000001                        .WORD   1         ;DIVIDEND
6427 026216  000000                        .WORD   0         ;INIT R5
6428 026220  000003                        .WORD   3         ;DIVISOR
6429 026222  000000                        .WORD   0         ;PSW
6430 026224  000001                        .WORD   1         ;R5 RESULT
6431 026226  052525                        .WORD   52525     ;R4 RESULT
6432
6433 026230  000023                        .WORD   23        ;DIVIDEND
6434 026232  016054                        .WORD   16054     ;INIT R5
6435 026234  016537                        .WORD   16537     ;DIVISOR
6436 026236  000000                        .WORD   0         ;PSW
6437 026240  010222                        .WORD   10222     ;R5 RESULT
6438 026242  000246                        .WORD   246       ;R4 RESULT
6439
6440 026244  000333                        .WORD   333
6441 026246                         FIN126:
6442                                ;
6445 026246                         TE127:
6446                                ;       TEST ASH (ARITHMETIC SHIFT)
6447 026246  005037  177776                CLR     @#177776                ;INIT PSW
6448 026252  012702  000001                MOV     #1,R2                   ;SETUP OPERAND
6449 026256  000277                        SCC                             ;SET ALL CC BITS
6450 026260  072202                        ASH     R2,R2                   ; TEST INSTRUCTION
6451 026262  022737  000000  177776        CMP     #0,@#177776             ;IS PS CORRECT
6452 026270  001401                        BEQ     1$                      ;YES GO ON
6453 026272  104001                        ERROR   +1              ;CPU ERROR
                                                                   ;NO GO TO ERROR
6454
6455 026274  020227  000002          1$:   CMP     R2,#2                   ;IS R2 CORRECT
6456 026300  001401                        BEQ     2$                      ;YES GO ON
6457 026302  104001                        ERROR   +1              ;CPU ERROR
                                                                   ;NO GO TO ERROR
6458
6459 026304  012702  100000          2$:   MOV     #100000,R2              ;SETUP R2
```

```
********** DOUBLE OPERAND TESTS *****************

6460 026310  012703  000001          MOV     #1,R3                   ;SETUP R3
6461 026314  000257                  CCC                             ;CLEAR ALL CC BITS
6462 026316  072203                  ASH     R3,R2                   ; TEST INSTRUCTION
6463 026320  022737  000007  177776  CMP     #7,@#177776             ;IS PS CORRECT
6464 026326  001401                  BEQ     3$                      ;YES GO ON
6465 026330  104001                  ERROR   ;1                      ;CPU ERROR
6466                                                                 ;NO GO TO ERROR
6467 026332  020327  000001    3$:   CMP     R3,#1                   ;IS R3 CORRECT
6468 026336  001401                  BEQ     4$                      ;YES GO ON
6469 026340  104001                  ERROR   ;1                      ;CPU ERROR
6470                                                                 ;NO GO TO ERROR
6471 026342  020227  000000    4$:   CMP     R2,#0                   ;IS R2 CORRECT
6472 026346  001401                  BEQ     5$                      ;YES GO ON
6473 026350  104001                  ERROR   ;1                      ;CPU ERROR
6474                                                                 ;NO GO TO ERROR
6475 026352  012701  026446    5$:   MOV     #TE127A,R1              ;SETUP POINTERS TO TABLES
6476
6477 026356  010103    6$:   MOV     R1,R3                   ;
6478 026360  016102  000002          MOV     2(R1),R2                ;SETUP R2
6479 026364  000277                  SCC                             ;SET ALL CC BITS
6480 026366  072211                  ASH     (R1),R2                 ; TEST INSTRUCTION
6481 026370  026137  000004  177776  CMP     4(R1),@#177776          ;IS PS CORRECT
6482 026376  001401                  BEQ     7$                      ;YES GO ON
6483 026400  104001                  ERROR   ;1                      ;CPU ERROR
6484                                                                 ;NO GO TO ERROR
6485 026402  026102  000006    7$:   CMP     6(R1),R2                ;IS R2 CORRECT
6486 026406  001401                  BEQ     8$                      ;YES GO ON
6487 026410  104001                  ERROR   ;1                      ;CPU ERROR
6488                                                                 ;NO GO TO ERROR
6489 026412  020301    8$:   CMP     R3,R1                   ;IS R1 CORRECT
6490 026414  001402                  BEQ     9$                      ;YES GO ON
6491 026416  104001                  ERROR   ;1                      ;CPU ERROR
6492                                                                 ;NO GO TO ERROR
6493 026420  010301                  MOV     R3,R1                   ;RESTORE R1
6494 026422  021311    9$:   CMP     (R3),(R1)               ;IS SOURCE CORRECT
6495 026424  001401                  BEQ     10$                     ;YES GO ON
6496 026426  104001                  ERROR   ;1                      ;CPU ERROR
6497                                                                 ;NO GO TO ERROR
6498                                                                 ;SOURCE LOOKS INCORRECT
6499 026430  062701  000010  10$:   ADD     #10,R1                  ;INCREMENT POINTER
6500 026434  020127  026706          CMP     R1,#FIN127              ;ARE WE DONE
6501 026440  001346                  BNE     6$                      ;NO GO TO 6$
6502
6503
6504 026442  000167  000240          JMP     FIN127
6505                                 ;
6506                                 ;
6507 026446  177761    TE127A: .WORD   177761          ;SOURCE
6508 026450  077777            .WORD   77777           ;DEST
6509 026452  000005            .WORD   5
6510 026454  000000            .WORD   0
6511
6512 026456  177700            .WORD   177700          ;SOURCE
6513 026460  017777            .WORD   17777           ;DEST
6514 026462  000000            .WORD   0
6515 026464  017777            .WORD   17777
6516
```

```
********** DOUBLE OPERAND TESTS *****************

6517 026466  177700                    .WORD   177700          ;SOURCE
6518 026470  100000                    .WORD   100000          ;DEST
6519 026472  000010                    .WORD   10
6520 026474  100000                    .WORD   100000
6521
6522 026476  177777                    .WORD   177777          ;SOURCE
6523 026500  100000                    .WORD   100000          ;DEST
6524 026502  000010                    .WORD   10
6525 026504  140000                    .WORD   140000
6526
6527                            ;*****************************************************************jpb
6528                            ;The following parameter source was modified to test both the
6529                            ;DCJ11-AA and DCJ11-AB processor chips.
6530
6531 026506  177761                    .WORD   177761          ;SOURCE
6532 026510  177777                    .WORD   177777          ;DEST
6533 026512  000011                    .WORD   11
6534 026514  177777                    .WORD   177777
6535                            ;*****************************************************************
6536
6537 026516  177706                    .WORD   177706          ;SOURCE
6538 026520  102000                    .WORD   102000          ;DEST
6539 026522  000007                    .WORD   7
6540 026524  000000                    .WORD   0
6541
6542 026526  177710                    .WORD   177710          ;SOURCE
6543 026530  017777                    .WORD   17777           ;DEST
6544 026532  000013                    .WORD   13
6545 026534  177400                    .WORD   177400
6546
6547 026536  177713                    .WORD   177713          ;SOURCE
6548 026540  000012                    .WORD   12              ;DEST
6549 026542  000000                    .WORD   0
6550 026544  050000                    .WORD   50000
6551
6552 026546  177707                    .WORD   177707          ;SOURCE
6553 026550  170001                    .WORD   170001          ;DEST
6554 026552  000002                    .WORD   2
6555 026554  000200                    .WORD   200
6556
6557 026556  177717                    .WORD   177717          ;SOURCE
6558 026560  000001                    .WORD   1               ;DEST
6559 026562  000012                    .WORD   12
6560 026564  100000                    .WORD   100000
6561
6562 026566  177740                    .WORD   177740          ;SOURCE
6563 026570  017777                    .WORD   17777           ;DEST
6564 026572  000004                    .WORD   4
6565 026574  000000                    .WORD   0
6566
6567 026576  177771                    .WORD   177771          ;SOURCE
6568 026600  150000                    .WORD   150000          ;DEST
6569 026602  000010                    .WORD   10
6570 026604  177640                    .WORD   177640
6571
6572 026606  177742                    .WORD   177742          ;SOURCE
6573 026610  100000                    .WORD   100000          ;DEST
```

F10


```
********** DOUBLE OPERAND TESTS *****************

6574 026612  000011                        .WORD   11
6575 026614  177777                        .WORD   177777
6576
6577 026616  177764                        .WORD   177764          ;SOURCE
6578 026620  100000                        .WORD   100000          ;DEST
6579 026622  000010                        .WORD   10
6580 026624  177770                        .WORD   177770
6581
6582 026626  177750                        .WORD   177750          ;SOURCE
6583 026630  052525                        .WORD   52525           ;DEST
6584 026632  000004                        .WORD   4
6585 026634  000000                        .WORD   0
6586
6587  026636  177760                       .WORD   177760          ;SOURCE
6588 026640  100000                        .WORD   100000          ;DEST
6589 026642  000011                        .WORD   11
6590 026644  177777                        .WORD   177777
6591
6592 026646  177770                        .WORD   177770          ;SOURCE
6593 026650  100000                        .WORD   100000          ;DEST
6594 026652  000010                        .WORD   10
6595 026654  177600                        .WORD   177600
6596
6597 026656  177712                        .WORD   177712          ;SOURCE
6598 026660  004367                        .WORD   4367            ;DEST
6599 026662  000013                        .WORD   13
6600 026664  156000                        .WORD   156000
6601
6602 026666  177764                        .WORD   177764          ;SOURCE
6603 026670  017777                        .WORD   17777           ;DEST
6604 026672  000001                        .WORD   1
6605 026674  000001                        .WORD   1
6606
6607 026676  177701                        .WORD   177701          ;SOURCE
6608 026700  110000                        .WORD   110000          ;DEST
6609 026702  000003                        .WORD   3
6610 026704  020000                        .WORD   20000
6611
6612 026706  000240            FIN127: NOP
6613                           ;
6616 026710                    TE130:
6617                           ;       TEST ASHC (ARITHMETIC SHIFT COMBINED)
6618 026710  005037  177776              CLR     @#177776                ;INIT PSW
6619 026714  012701  000023              MOV     #23,R1                  ;SETUP R1
6620 026720  012705  052525              MOV     #52525,R5               ;SETUP R5
6621 026724  005004                      CLR     R4                      ;SETUP R4
6622 026726  000277                      SCC                             ;SET ALL CC BITS
6623 026730  073401                      ASHC    R1,R4                   ; TEST INSTRUCTION
6624 026732  023727  177776  000012      CMP     @#177776,#12            ;IS PS CORRECT
6625 026740  001401                      BEQ     1$                      ;YES GO ON
6626 026742  104001                      ERROR   ;1              ;CPU ERROR
                                                                 ;NO GO TO ERROR
6627
6628 026744  020127  000023      1$:     CMP     R1,#23                  ;IS R1 CORRECT
6629 026750  001401                      BEQ     2$                      ;YES GO ON
6630 026752  104001                      ERROR   ;1              ;CPU ERROR
                                                                 ;NO GO TO ERROR
6631
6632 026754  020427  125250      2$:     CMP     R4,#125250              ;IS R4 CORRECT
```

G10


```
********** DOUBLE OPERAND TESTS *****************

6633 026760  001401                        BEQ     3$                      ;YES GO ON
6634 026762  104001                        ERROR   +1              ;CPU ERROR
6635                                                               ;NO GO TO ERROR
6636 026764  020527  000000          3$:   CMP     R5,#0                   ;IS R5 CORRECT
6637 026770  001401                        BEQ     4$                      ;YES GO ON
6638 026772  104001                        ERROR   +1              ;CPU ERROR
6639                                                               ;NO GO TO ERROR
6640 026774  012703  052525          4$:   MOV     #52525,R3               ;SETUP R3
6641 027000  005002                        CLR     R2                      ;SETUP R2
6642 027002  012704  164731                MOV     #164731,R4              ;SETUP R4
6643 027006  000277                        SCC                             ;SET ALL CC BITS
6644 027010  073327  000023                ASHC    #23,R3                  ; TEST INSTRUCTION
6645 027014  023727  177776  000012        CMP     @#177776,#12            ;IS PS CORRECT
6646 027022  001401                        BEQ     5$                      ;YES GO ON
6647 027024  104001                        ERROR   +1              ;CPU ERROR
6648                                                               ;NO GO TO ERROR
6649 027026  020227  000000          5$:   CMP     R2,#0                   ;IS R2 CORRECT
6650 027032  00140?                        BEQ     6$                      ;YES GO ON
6651 027034  104001                        ERROR   +1              ;CPU ERROR
6652                                                               ;NO GO TO ERROR
6653 027036  020327  000000          6$:   CMP     R3,#0                   ;IS R3 CORRECT
6654 027042  001401                        BEQ     7$                      ;YES GO ON
6655 027044  104001                        ERROR   +1              ;CPU ERROR
6656                                                               ;NO GO TO ERROR
6657 027046  020427  164731          7$:   CMP     R4,#164731              ;IS R4 CORRECT
6658 027052  001401                        BEQ     8$                      ;YES GO ON
6659 027054  104001                        ERROR   +1              ;CPU ERROR
6660                                                               ;NO GO TO ERROR
6661                                                               ;
6662                                                               ;
6663 027056  012701  027166          8$:   MOV     #TE130A,R1              ;SETUP POINTERS TO TABLES
6664
6665 027062  010104                  9$:   MOV     R1,R4                   ;SAVE A COPY OF R1
6666 027064  016102  000002                MOV     2(R1),R2                ;SETUP R2
6667 027070  016103  000004                MOV     4(R1),R3                ;SETUP R3
6668 027074  000277                        SCC                             ;SET ALL CC BITS
6669 027076  073211                        ASHC    (R1),R2         ; TEST INSTRUCTION
6670 027100  023761  177776  000006        CMP     @#177776,6(R1)          ;IS PS CORRECT
6671 027106  001401                        BEQ     10$                     ;YES GO ON
6672 027110  104001                        ERROR   +1              ;CPU ERROR
6673                                                               ;NO GO TO ERROR
6674 027112  026102  000010         10$:   CMP     10(R1),R2               ;IS R2 CORRECT
6675 027116  001401                        BEQ     11$                     ;YES GO ON
6676 027120  104001                        ERROR   +1              ;CPU ERROR
6677                                                               ;NO GO TO ERROR
6678 027122  026103  000012         11$:   CMP     12(R1),R3               ;IS R3 CORRECT
6679 027126  001401                        BEQ     12$                     ;YES GO ON
6680 027130  104001                        ERROR   +1              ;CPU ERROR
6681                                                               ;NO GO TO ERROR
6682 027132  020401                  12$:  CMP     R4,R1                   ;IS R1 CORRECT
6683 027134  001402                        BEQ     13$                     ;YES GO ON
6684 027136  104001                        ERROR   +1              ;CPU ERROR
6685                                                               ;NO GO TO ERROR
6686 027140  010401                        MOV     R4,R1
6687 027142  021114                  13$:  CMP     (R1),(R4)               ;IS SOURCE CORRECT
6688 027144  001401                        BEQ     14$                     ;YES GO ON
6689 027146  104001                        ERROR   +1              ;CPU ERROR
```

H10


```
********** DOUBLE OPERAND TESTS *****************

                                                                ;NO GO TO ERROR
                                                                ;POSSIBLE SOURCE CODE CORRUPTION
                                                                ;GO TO NEXT TEST
027150  062701  000014          14$:    ADD     #14,R1          ;ARE WE DONE
027154  020127  027576                  CMP     R1,#FIN130      ;NO GO TO 9$
027160  001340                          BNE     9$

027162  000167  000410                  JMP     FIN130
                                ;
                                ;
027166  177700                  TE130A: .WORD   177700          ;SOURCE
027170  100125                          .WORD   100125          ;DESTINATION WORD 1
027172  177777                          .WORD   177777          ;DESTINATION WORD 2
027174  000010                          .WORD   10              ;TEST PSW
027176  100125                          .WORD   100125          ;RESULT WORD 1
027200  177777                          .WORD   177777          ;RESULT WORD 2

027202  177777                          .WORD   177777          ;SOURCE
027204  000001                          .WORD   1               ;DESTINATION WORD 1
027206  000000                          .WORD   0               ;DESTINATION WORD 2
027210  000000                          .WORD   0               ;TEST PSW
027212  000000                          .WORD   0               ;RESULT WORD 1
027214  100000                          .WORD   100000          ;RESULT WORD 2

027216  177701                          .WORD   177701          ;SOURCE
027220  047777                          .WORD   47777           ;DESTINATION WORD 1
027222  100000                          .WORD   100000          ;DESTINATION WORD 2
027224  000012                          .WORD   12              ;TEST PSW
027226  117777                          .WORD   117777          ;RESULT WORD 1
027230  000000                          .WORD   0               ;RESULT WORD 2

027232  177706                          .WORD   177706          ;SOURCE
027234  004256                          .WORD   4256            ;DESTINATION WORD 1
027236  177700                          .WORD   177700          ;DESTINATION WORD 2
027240  000002                          .WORD   2               ;TEST PSW
027242  025677                          .WORD   25677           ;RESULT WORD 1
027244  170000                          .WORD   170000          ;RESULT WORD 2

027246  177711                          .WORD   177711          ;SOURCE
027250  065700                          .WORD   65700           ;DESTINATION WORD 1
027252  000012                          .WORD   12              ;DESTINATION WORD 2
027254  000013                          .WORD   13              ;TEST PSW
027256  100000                          .WORD   100000          ;RESULT WORD 1
027260  012000                          .WORD   12000           ;RESULT WORD 2

                        ;*****************************************************************jpb
                        ;The following parameter source was modified to test both the
                        ;DCJ11-AA and DCJ11-AB processor chips.

027262  177761                          .WORD   177761          ;SOURCE
027264  000000                          .WORD   0               ;DESTINATION WORD 1
027266  000001                          .WORD   1               ;DESTINATION WORD 2
027270  000004                          .WORD   4               ;TEST PSW
027272  000000                          .WORD   0               ;RESULT WORD 1
027274  000000                          .WORD   0               ;RESULT WORD 2
                        ;*************************************************************
```

********** DOUBLE OPERAND TESTS **********

```
6747 027276  177736                .WORD   177736          ;SOURCE
6748 027300  000000                .WORD   0               ;DESTINATION WORD 1
6749 027302  000001                .WORD   1               ;DESTINATION WORD 2
6750 027304  000000                .WORD   0               ;TEST PSW
6751 027306  040000                .WORD   40000           ;RESULT WORD 1
6752 027310  000000                .WORD   0               ;RESULT WORD 2
6753
6754 027312  177740                .WORD   177740          ;SOURCE
6755 027314  100000                .WORD   100000          ;DESTINATION WORD 1
6756 027316  000000                .WORD   0               ;DESTINATION WORD 2
6757 027320  000011                .WORD   11              ;TEST PSW
6758 027322  177777                .WORD   177777          ;RESULT WORD 1
6759 027324  177777                .WORD   177777          ;RESULT WORD 2
6760
6761 027326  177725                .WORD   177725          ;SOURCE
6762 027330  177777                .WORD   177777          ;DESTINATION WORD 1
6763 027332  174000                .WORD   174000          ;DESTINATION WORD 2
6764 027334  000007                .WORD   7               ;TEST PSW
6765 027336  000000                .WORD   0               ;RESULT WORD 1
6766 027340  000000                .WORD   0               ;RESULT WORD 2
6767
6768 027342  177724                .WORD   177724          ;SOURCE
6769 027344  177777                .WORD   177777          ;DESTINATION WORD 1
6770 027346  174000                .WORD   174000          ;DESTINATION WORD 2
6771 027350  000011                .WORD   11              ;TEST PSW
6772 027352  100000                .WORD   100000          ;RESULT WORD 1
6773 027354  000000                .WORD   0               ;RESULT WORD 2
6774
6775 027356  177733                .WORD   177733          ;SOURCE
6776 027360  177777                .WORD   177777          ;DESTINATION WORD 1
6777 027362  157023                .WORD   157023          ;DESTINATION WORD 2
6778 027364  000012                .WORD   12              ;TEST PSW
6779 027366  114000                .WORD   114000          ;RESULT WORD 1
6780 027370  000000                .WORD   0               ;RESULT WORD 2
6781
6782 027372  177727                .WORD   177727          ;SOURCE
6783 027374  000000                .WORD   0               ;DESTINATION WORD 1
6784 027376  177777                .WORD   177777          ;DESTINATION WORD 2
6785 027400  000013                .WORD   13              ;TEST PSW
6786 027402  177600                .WORD   177600          ;RESULT WORD 1
6787 027404  000000                .WORD   0               ;RESULT WORD 2
6788
6789 027406  177717                .WORD   177717          ;SOURCE
6790 027410  177777                .WORD   177777          ;DESTINATION WORD 1
6791 027412  000001                .WORD   1               ;DESTINATION WORD 2
6792 027414  000011                .WORD   11              ;TEST PSW
6793 027416  100000                .WORD   100000          ;RESULT WORD 1
6794 027420  100000                .WORD   100000          ;RESULT WORD 2
6795
6796 027422  177741                .WORD   177741          ;SOURCE
6797 027424  100000                .WORD   100000          ;DESTINATION WORD 1
6798 027426  000000                .WORD   0               ;DESTINATION WORD 2
6799 027430  000010                .WORD   10              ;TEST PSW
6800 027432  177777                .WORD   177777          ;RESULT WORD 1
6801 027434  177777                .WORD   177777          ;RESULT WORD 2
6802
6803 027436  177742                .WORD   177742          ;SOURCE
```

```
********** DOUBLE OPERAND TESTS *****************

6804 027440  037777                    .WORD   37777           ;DESTINATION WORD 1
6805 027442  177777                    .WORD   177777          ;DESTINATION WORD 2
6806 027444  000005                    .WORD   5               ;TEST PSW
6807 027446  000000                    .WORD   0               ;RESULT WORD 1
6808 027450  00[illegible]             .WORD   0               ;RESULT WORD 2
6809
6810 027452  [illegible]               .WORD   177742          ;SOURCE
6811 027454  07[illegible]             .WORD   77777           ;DESTINATION WORD 1
6812 027456  177[illegible]            .WORD   177777          ;DESTINATION WORD 2
6813 027460  000001                    .WORD   1               ;TEST PSW
6814 027462  000000                    .WORD   0               ;RESULT WORD 1
6815 027464  000001                    .WORD   1               ;RESULT WORD 2
6816
6817 027466  177711                    .WORD   177711          ;SOURCE
6818 027470  065600                    .WORD   65600           ;DESTINATION WORD 1
6819 027472  000012                    .WORD   12              ;DESTINATION WORD 2
6820 027474  000003                    .WORD   3               ;TEST PSW
6821 027476  000000                    .WORD   0               ;RESULT WORD 1
6822 027500  012000                    .WORD   12000           ;RESULT WORD 2
6823
6824 027502  177740                    .WORD   177740          ;SOURCE
6825 027504  077777                    .WORD   77777           ;DESTINATION WORD 1
6826 027506  177777                    .WORD   177777          ;DESTINATION WORD 2
6827 027510  000004                    .WORD   4               ;TEST PSW
6828 027512  000000                    .WORD   0               ;RESULT WORD 1
6829 027514  000000                    .WORD   0               ;RESULT WORD 2
6830
6831                            ;**************************************************************jpb
6832                            ;The following parameter source was modified to test both the
6833                            ;DCJ11 AA and DCJ11-AB processor chips.
6834
6835 027516  177761                    .WORD   177761          ;SOURCE
6836 027520  177777                    .WORD   177777          ;DESTINATION WORD 1
6837 027522  177774                    .WORD   177774          ;DESTINATION WORD 2
6838 027524  000011                    .WORD   11              ;TEST PSW
6839 027526  177777                    .WORD   177777          ;RESULT WORD 1
6840 027530  177777                    .WORD   177777          ;RESULT WORD 2
6841                            ;**************************************************************
6842
6843 027532  177747                    .WORD   177747          ;SOURCE
6844 027534  100000                    .WORD   100000          ;DESTINATION WORD 1
6845 027536  174000                    .WORD   174000          ;DESTINATION WORD 2
6846 027540  000010                    .WORD   10              ;TEST PSW
6847 027542  17777[illegible]          .WORD   177777          ;RESULT WORD 1
6848 027544  177700                    .WORD   177700          ;RESULT WORD 2
6849
6850 027546  177753                    .WORD   177753          ;SOURCE
6851 027550  006324                    .WORD   6324            ;DESTINATION WORD 1
6852 027552  071002                    .WORD   71002           ;DESTINATION WORD 2
6853 027554  000001                    .WORD   1               ;TEST PSW
6854 027556  000000                    .WORD   0               ;RESULT WORD 1
6855 027560  000146                    .WORD   146             ;RESULT WORD 2
6856
6857 027562  177765                    .WORD   177765          ;SOURCE
6858 027564  102351                    .WORD   102351          ;DESTINATION WORD 1
6859 027566  177231                    .WORD   177231          ;DESTINATION WORD 2
6860 027570  000011                    .WORD   11              ;TEST PSW
```

```
********** DOUBLE OPERAND TESTS *****************

 6861 027572  177760                        .WORD   177760          ;RESULT WORD 1
 6862 027574  116477                        .WORD   .16477          ;RESULT WORD 2
 6863 027576                        FTN130:
 6866 027576                        MSPAU:
 6867
 6868                               ;       TEST THAT AUTO DEC/INC OPERATIONS USING SP ARE ON WORD BOUNDRYS
 6869 027576  005006                        CLR     R6                      ;CLEAR SP
 6870 027600  112667  153314                MOVB    (R6)+,COUNT             ;TRY AUTOINC ON R6
 6871 027604  022706  000002                CMP     #2,R6                   ;VERIFY AUTO INC BY 2
 6872 027610  001401                        BEQ     SPAU1                   ;BRANCH IF GOOD
 6873                                                                       ;BAD AUTO-INC
 6874 027612  104001                        ERROR   +1              ;CPU ERROR
 6875 027614  005006                SPAU1:  CLR     R6                      ;CLEAR R6
 6876 027616  112667  153276                MOVB    (R6)+,COUNT
 6877 027622  112667  153272                MOVB    (R6)+,COUNT             ;DOUBLE BYTE AUTO INC
 6878 027626  022706  000004                CMP     #4,R6           ;VERIFY RESULT
 6879 027632  001401                        BEQ     SPAU2           ;BRANCH IF GOOD
 6880 027634  104001                        ERROR   +1              ;CPU ERROR
 6881                                                       ;BAD DOUBLE AUTO-INC
 6882 027636  012706  001000        SPAU2:  MOV     #STBOT,R6               ;LOAD R6
 6883 027642  114667  153252                MOVB    -(R6),COUNT             ; TEST AUTO-DEC
 6884 027646  022706  000776                CMP     #776,R6         ;VERIFY RESULT
 6885 027652  001401                        BEQ     SPAU3           ;BRANCH IF GOOD
 6886 027654  104001                        ERROR   +1              ;CPU ERROR
 6887
 6888 027656  012706  001000        SPAU3:  MOV     #STBOT,R6               ;LOAD R6
 6889 027662  114667  153232                MOVB    -(R6),COUNT             ; TEST AUTO-DEC
 6890 027666  114667  153226                MOVB    -(R6),COUNT             ; TEST AUTO-DEC
 6891 027672  022706  000774                CMP     #774,R6         ;VERIFY RESULT
 6892 027676  001401                        BEQ     SPAU4           ;BRANCH IF GOOD
 6893 027700  104001                        ERROR   +1              ;CPU ERROR
 6894
 6895 027702  005006                SPAU4:  CLR     R6              ; TEST AUTO-INC ON SOP
 6896 027704  105726                        TSTB    (R6)+           ; TEST AUTO-INC
 6897 027706  020627  000002                CMP     R6,#2
 6898 027712  001401                        BEQ     SPAU5           ;BRANCH IF GOOD
 6899 027714  104001                        ERROR   +1              ;CPU ERROR
 6900
 6901 027716  012706  001000        SPAU5:  MOV     #STBOT,R6       ;LOAD R6
 6902 027722  105746                        TSTB    -(R6)           ; TEST AUTO-DEC
 6903 027724  022706  000776                CMP     #776,R6         ;VERIFY RESULT
 6904 027730  001401                        BEQ     SPAU6           ;BRANCH IF GOOD
 6905 027732  104001                        ERROR   +1              ;CPU ERROR
 6906
 6907 027734  012706  001000        SPAU6:  MOV     #STBOT,R6
 6908
 6910                               ;
 6912 027740                        MTRY:
 6913
 6914                               ;       VERIFY YELLOW ZONE TRAP ON AUTO DEC OF R6
 6915 027740  005067  150022                CLR     CPEREG          ;INIT CPU ERROR REGISTER
 6916 027744  012706  000150                MOV     #150,R6         ;LOAD R6 WITH A VALUE THAT WILL
 6917                                                               ;CAUSE A YELLOW STACK TRAP(IE. <400)
 6918 027750  016767  150030  153034        MOV     4,SLOC00        ;SAVE VECTOR
 6919 027756  012767  030014  150020        MOV     #MTRYA,4        ;SETUP THE STACK OVERFLOW TRAP POINTER
 6920 027764  016701  150156                MOV     146,R1          ;SAVE VECTOR
 6921 027770  016702  150150                MOV     144,R2          ;SAVE VECTOR
```

```
********** DOUBLE OPERAND TESTS *****************

 6922 027774  016703  150142                MOV     142,R3          ;SAVE VECTOR
 6923 030000  005067  150142                CLR     146             ;JUST AS A PRECAUTION
 6924 030004  005046                        CLR     (R6)            ;CAUSE A STACK OVERFLOW TRAP
 6925 030006  012706  001000                MOV     #STBOT,R6       ;RESTORE R6 FOR ERROR CALL
 6926 030012  104001                        ERROR   +1              ;CPU ERROR
 6927                                                       ;OVERFLOW TRAP FAILED
 6928 030014                        MTRYA:
 6929 030014  022767  000010  147744        CMP     #BIT03,CPEREG   ;WAS CPU ERROR REG SET PROPERLY?
 6930 030022  001003                        BNE     1$              ;GO TO ERROR IF NOT
 6931 030024  020627  000142                CMP     R6,#142         ;VERIFY CORRECT DECRIMENT OF R6
 6932 030030  001401                        BEQ     MTRYB           ;BRANCH IF GOOD
 6933 030032  104001                1$:     ERROR   +1              ;CPU ERROR
 6934                                                       ;R6 IMPROPERLY DECRIMENTED
 6935                                                       ;OR CPU ERROR REGISTER NOT CORRECT
 6936 030034                        MTRYB:
 6937 030034  005067  147726                CLR     CPEREG          ;CLEAR THE CPU ERROR REGISTER
 6938 030040  016767  152746  147736        MOV     SLOC00,4        ;RESTORE VECTOR
 6939 030046  010167  150074                MOV     R1,146          ;RESTORE VECTORS
 6940 030052  010267  150066                MOV     R2,144          ;       "
 6941 030056  010367  150060                MOV     R3,142          ;       "
 6942 030062  012706  001000                MOV     #STBOT,R6
 6943
 6944
 6946                                       ;
 6948 030066                        MTRYM:
 6949
 6950                                       ;       TEST STACK OVERFLOW TRAPS IN VARIOUS MODES
 6951 030066  005067  147674                CLR     CPEREG                  ;CLEAR CPU ERROR REGISTER
 6952 030072  012706  000400                MOV     #400,R6                 ;SETUP OVERFLOW R6 DATA
 6953 030076  016767  147702  152706        MOV     4,SLOC00                ;SAVE VECTOR
 6954 030104  012767  030126  147672        MOV     #TRYMA,4
 6955 030112  005067  150260                CLR     376                     ;JUST AS A PRECAUTION
 6956 030116  005046                        CLR     -(R6)                   ;CAUSE OVERFLOW TRAP
 6957 030120  012706  001000                MOV     #STBOT,R6               ;RESTORE R6 FOR ERROR CALL
 6958 030124  104001                        ERROR   +1              ;CPU ERROR
 6959                                                               ;NO OVERFLOW TRAP
 6960 030126                        TRYMA:
 6961 030126  005067  147634                CLR     CPEREG                  ;CLEAR CPU ERROR REGISTER
 6962 030132  012705  001000                MOV     #1000,R5                ;SETUP R5 DATA
 6963 030136  012706  000400                MOV     #400,R6                 ;SETUP OVERFLOW R6 DATA
 6964 030142  012767  030160  147634        MOV     #TRYMB,4
 6965 030150  064645                        ADD     -(R6),-(R5)             ;CAUSE OVERFLOW TRAP
 6966 030152  012706  001000                MOV     #STBOT,R6               ;RESTORE R6 FOR ERROR CALL
 6967 030156  104001                        ERROR   +1              ;CPU ERROR
 6968                                                               ;NO OVERFLOW TRAP
 6969 030160                        TRYMB:
 6970 030160  005067  147602                CLR     CPEREG                  ;CLEAR CPU ERROR REGISTER
 6971 030164  012706  000150                MOV     #150,R6                 ;SETUP OVERFLOW R6 DATA
 6972 030170  012767  030206  147606        MOV     #TRYMC,4
 6973 030176  044546                        BIC     -(R5),-(R6)             ;CAUSE OVERFLOW TRAP
 6974 030200  012706  001000                MOV     #STBOT,R6               ;RESTORE R6 FOR ERROR CALL
 6975 030204  104001                        ERROR   +1              ;CPU ERROR
 6976                                                               ;NO OVERFLOW TRAP
 6977 030206  005067  147554        TRYMC:  CLR     CPEREG                  ;CLEAR CPU ERROR REGISTER
 6978 030212  016767  152574  147564        MOV     SLOC00,4                ;RESTORE VECTOR
 6979 030220  012706  001000                MOV     #STBOT,R6
 6980
```

```
********** DOUBLE OPERAND TESTS *****************

                                        ;
030224                                  MILL0:
                                        ;       TEST STACK OVERFLOW ON ILLEGAL INST TRAP
030224  005067  147536                          CLR     CPEREG                  ;CLEAR CPU ERROR REGISTER
030230  012706  000400                          MOV     #400,R6                 ;SETUP FOR OVERFLOW TRAP
030234  016767  147550  152550                MOV     10,SLOC00               ;SAVE VECTOR
030242  012767  030270  147540                MOV     #MILLOA,10              ;SETUP ILLEGAL TRAP VECTOR
030250  016767  147530  152536                MOV     4,SLOC01                ;SAVE VECTOR
030256  012767  030276  147520                MOV     #MILL0B,4               ;SETUP OVERFLOW TRAP VECTOR
030264  000077                                77                              ;UNUSED INSTRUCTION TRAP
030266  000240                                NOP
030270  012706  001000          MILLOA:       MOV     #STBOT,R6               ;RESTORE R6 FOR ERROR CALL
030274  104001                                ERROR   +1              ;CPU ERROR
                                                                        ;UNUSED INSTRUCTION TRAP
030276                                  MILL0B:
030276  016767  152512  147500                MOV     SLOC01,4                ;RESTORE VECTOR
030304  016767  152502  147476                MOV     SLOC00,10               ;RESTORE VECTOR
030312  005067  147450                        CLR     CPEREG                  ;CLEAR CPU ERROR REGISTER
030316  012706  001000                        MOV     #STBOT,R6               ;RESTORE R6



                                        ;
030322                                  MIOTO:
                                        ;       TEST STACK OVERFLOW ON IOT TRAP
030322  005067  147440                          CLR     CPEREG                  ;CLEAR CPU ERROR REGISTER
030326  012706  000400                          MOV     #400,R6                 ;SETUP STACK FOR OVERFLOW
030332  016767  147462  152452                MOV     20,SLOC00               ;SAVE OLD IOT VECTOR
030340  012767  030366  147452                MOV     #IOTOA,20               ;SETUP ERROR ACTION ON IOT
030346  016767  147432  152440                MOV     4,SLOC01                ;SAVE VECTOR
030354  012767  030374  147422                MOV     #IOTOB,4                ;SETUP CORRECT TRAP VECTOR FOR
                                                                                ;OVERFLOW
030362  000004                                IOT                             ; TEST INSTRUCTION
030364  000240                                NOP
030366  012706  001000          IOTOA:        MOV     #STBOT,R6               ;RESTORE R6 FOR ERROR CALL
030372  104001                                ERROR   +1              ;CPU ERROR
                                                                        ;FAILURE OF STACK OVERFLOW
030374                                  IOTOB:
030374  005067  147366                        CLR     CPEREG                  ;CLEAR CPU ERROR REGISTER
030400  012706  001000                        MOV     #STBOT,R6
030404  016767  152404  147372                MOV     SLOC01,4                ;RESTORE VECTOR
030412  016767  152374  147400                MOV     SLOC00,20               ;RESTORE TRAP VECTOR


                                        ;
                                        ;
030420                                  MEMTO:
                                        ;       TEST STACK OVERFLOW ON EMT TRAP
030420  005067  147342                          CLR     CPEREG                  ;CLEAR CPU ERROR REGISTER
030424  012706  000400                          MOV     #400,R6                 ;SETUP STACK FOR OVERFLOW
030430  016767  147374  152354                MOV     30,SLOC00               ;SAVE OLD EMT VECTOR
030436  012767  030464  147364                MOV     #EMTOA,30               ;SETUP ERROR ACTION ON EMT
030444  016767  147334  152342                MOV     4,SLOC01                ;SAVE VECTOR
030452  012767  030472  147324                MOV     #EMTOB,4                ;SETUP CORRECT TRAP VECTOR FOR
```

N10

********** DOUBLE OPERAND TESTS **********

```
7044                                                                    ;OVERFLOW
7045 030460  104000                             EMT                     ; TEST INSTRUCTION
7046 030462  000240                             NOP
7047 030464  012706  001000          EMTOA:     MOV     #STBOT,R6       ;RESTORE R6 FOR ERROR CALL
7048 030470  104001                             ERROR   +1          ;CPU ERROR
7049                                                                    ;FAILURE OF STACK OVERFLOW
7050 030472                          EMTOB:
7051 030472  016767  152314  147330             MOV     SLOC00,30       ;RESTORE TRAP VECTOR
7052 030500  016767  152310  147276             MOV     SLOC01,4        ;RESTORE VECTOR
7053 030506  005067  147254                     CLR     CPEREG      ;CLEAR CPU ERROR REGISTER
7054 030512  012706  001000                     MOV     #STBOT,R6
7055
7058 030516                          MTRPO:
7059
7060                                 ;          TEST STACK OVERFLOW ON TRAP
7061 030516  005067  147244                     CLR     CPEREG      ;CLEAR CPU ERROR REGISTER
7062 030522  012706  000400                     MOV     #400,R6             ;SETUP STACK FOR OVERFLOW
7063 030526  016767  147302  152256             MOV     34,SLOC00           ;SAVE OLD TRP VECTOR
7064 030534  012767  030562  147272             MOV     #TRPOA,34           ;SETUP ERROR ACTION ON TRP
7065 030542  016767  147236  152244             MOV     4,SLOC01            ;SAVE VECTOR
7066 030550  012767  030570  147226             MOV     #TRPOB,4            ;SETUP CORRECT TRAP VECTOR FOR
7067                                                                    ;OVERFLOW
7068 030556  104400                             TRAP                    ; TEST INSTRUCTION
7069 030560  000240                             NOP
7070 030562  012706  001000          TRPOA:     MOV     #STBOT,R6       ;RESTORE R6 FOR ERROR CALL
7071 030566  104001                             ERROR   +1          ;CPU ERROR
7072                                                                    ;FAILURE OF STACK OVERFLOW
7073 030570                          TRPOB:
7074 030570  016767  152216  147236             MOV     SLOC00,34       ;RESTORE TRAP VECTOR
7075 030576  016767  152212  147200             MOV     SLOC01,4        ;RESTORE VECTOR
7076 030604  005067  147156                     CLR     CPEREG      ;CLEAR CPU ERROR REGISTER
7077 030610  012706  001000                     MOV     #STBOT,R6
7078
7080                                 ;
7081                                 ;
7083 030614                          MBPTO:
7084
7085                                 ;          TEST STACK OVERFLOW ON BPT
7086 030614  005067  147146                     CLR     CPEREG      ;CLEAR CPU ERROR REGISTER
7087 030620  012706  000400                     MOV     #400,R6             ;SETUP STACK FOR OVERFLOW
7088 030624  016767  147164  152160             MOV     14,SLOC00           ;SAVE OLD BPT VECTOR
7089 030632  012767  030660  147154             MOV     #BPTOA,14           ;SETUP ERROR ACTION ON BPT
7090 030640  016767  147140  152146             MOV     4,SLOC01            ;SAVE VECTOR
7091 030646  012767  030666  147130             MOV     #BPTOB,4            ;SETUP CORRECT TRAP VECTOR FOR
7092                                                                    ;OVERFLOW
7093 030654  000003                             BPT                     ; TEST INSTRUCTION
7094 030656  000240                             NOP
7095 030660  012706  001000          BPTOA:     MOV     #STBOT,R6       ;RESTORE R6 FOR ERROR CALL
7096 030664  104001                             ERROR   +1          ;CPU ERROR
7097                                                                    ;FAILURE OF STACK OVERFLOW
7098 030666                          BPTOB:
7099 030666  005067  147074                     CLR     CPEREG      ;CLEAR CPU ERROR REGISTER
7100 030672  016767  152114  147114             MOV     SLOC00,14           ;RESTORE TRAP VECTOR
7101 030700  016767  152110  147076             MOV     SLOC01,4            ;RESTORE VECTOR
7102 030706  012706  001000                     MOV     #STBOT,R6
7103
7105                                 ;
```

```
********** DOUBLE OPERAND TESTS **********

                                        ;
                                MILAO:
                                        ;       TEST STACK OVERFLOW AND ILLEGAL JMP INSTRUCTION
                                                CLR     CPEREG                  ;CLEAR CPU ERROR REGISTER
                                                MOV     #400,R6                         ;SETUP STACK FOR OVERFLOW
                                                MOV     10,SLOC00               ;SAVE OLD ILLEGAL INST. VECTOR
                                                MOV     #ILAOA,10               ;SETUP ERROR ACTION ILLEGAL OPCODE
                                                MOV     4,SLOC01                ;SAVE VECTOR
                                                MOV     #ILBOB,4                ;SETUP CORRECT TRAP VECTOR FOR
                                                                                ;OVERFLOW
                                                CLR     R1
                                                JMP     R1                      ; TEST INSTRUCTION
                                                NOP
                                ILAOA:          MOV     #STBOT,R6               ;RESTORE R6 FOR ERROR CALL
                                                ERROR   +1              ;CPU ERROR
                                                                        ;FAILURE OF STACK OVERFLOW
                                ILBOB:
                                                MOV     SLOC01,4                ;RESTORE VECTOR
                                                MOV     SLOC00,10               ;RESTORE TRAP VECTOR
                                                CLR     CPEREG                  ;CLEAR CPU ERROR REGISTER
                                                MOV     #STBOT,R6

                                        ;
                                        ;
                                MILLBO:
                                        ;       TEST STACK OVERFLOW ON ILLEGAL JSR INST
                                                MOV     #400,R6                 ;SETUP STACK FOR OVERFLOW
                                                MOV     10,SLOC00               ;SAVE OLD VECTOR
                                                MOV     #ILLBOA,10              ;SETUP ERROR ACTION ON ILL. OPCODE
                                                MOV     4,SLOC01                ;SAVE VECTOR
                                                MOV     #ILLBOB,4               ;SETUP CORRECT TRAP VECTOR FOR
                                                                                ;OVERFLOW
                                                CLR     R1
                                                JSR     R5,R1                   ; TEST INSTRUCTION
                                                NOP
                                ILLBOA:         MOV     #STBOT,R6               ;RESTORE R6 FOR ERROR CALL
                                                ERROR   +1              ;CPU ERROR
                                                                        ;FAILURE OF STACK OVERFLOW
                                ILLBOB:
                                                MOV     SLOC01,4                ;RESTORE VECTOR
                                                MOV     SLOC00,10               ;RESTORE TRAP VECTOR
                                                MOV     #STBOT,R6

                                        ;
                                        ;
                                MSTO:
                                        ;       TEST FOR FALSE STACK OVERFLOW
                                                MOV     4,SLOC00                ;SAVE VECTOR
                                                MOV     #MSTOE,4                ;ANTICIPATE OVERFLOW ERROR
                                                MOV     #1002,R6                ;SETUP LEGAL R6
                                                TST     -(R6)                   ;TRY TO CAUSE STACK OVERFLOW
                                                MOV     #2002,R6                ;SETUP LEGAL R6
                                                TST     -(R6)                   ;TRY TO CAUSE STACK OVERFLOW
                                                MOV     #4002,R6                ;SETUP LEGAL R6
```

```
********** DOUBLE OPERAND TESTS *****************

7168 031136  005746                          TST     -(R6)                   ;TRY TO CAUSE STACK OVERFLOW
7169 031140  012706  010002                  MOV     #10002,R6               ;SETUP LEGAL R6
7170 031144  005746                          TST     -(R6)                   ;TRY TO CAUSE STACK OVERFLOW
7171 031146  012706  100402                  MOV     #100402,R6              ;SETUP  LEGAL R6
7172 031152  005746                          TST      (R6)                   ;TRY TO CAUSE STACK OVERFLOW
7173 031154  000403                          BR      MSTOEE                  ;EXIT MODULE
7174 031156  012706  001000          MSTOE:  MOV     #STBOT,R6               ;RESTORE R6 FOR ERROR CALL
7175 031162  104001                          ERROR   .+1             ;CPU ERROR
7176                                                                 ;STACK OVERFLOW ERROR
7177 031164  016767  151622  146612  MSTOEE: MOV     SLOC00,4                ;RESTORE VECTOR
7178 031172  012706  001000                  MOV     #STBOT,R6
7179
7181                                 ;
7182                                 ;
7184 031176                          MTT:
7185
7186                                 ;       TEST T-BIT TRAPS
7187 031176  012706  001000                  MOV     #STBOT,R6               ;SETUP STACK
7188 031202  016767  146606  151602          MOV     14,SLOC00               ;SAVE OLD T-BIT VECTOR
7189 031210  012746  000020                  MOV     #20,-(R6)               ;PUSH T-BIT
7190 031214  012746  031232                  MOV     #MTTA,-(R6)             ;SETUP ERROR TRAP VECTOR
7191 031220  012767  031234  146566          MOV     #MTTB,14                ;SETUP NEW T-BIT VECTOR
7192 031226  000002                          RTI                             ;CAUSE A T BIT SET IN PSW
7193 031230  104001                          ERROR   .+1             ;CPU ERROR
7194                                                                 ;SHOULD NEVER BE EXECUTED
7195 031232  104001                  MTTA:   ERROR   .+1             ;CPU ERROR
7196                                                                 ;DIDNT TAKE CORRECT TRAP
7197 031234  022706  000774          MTTB:   CMP     #STBOT-4,R6             ;VERIFY SP DECIRMENT
7198 031240  001401                          BEQ     MTTD                    ;BRANCH IF GOOD
7199 031242  104001                          ERROR   .+1             ;CPU ERROR
7200                                                                 ;BAD SP
7201 031244  021627  031232          MTTD:   CMP     (R6),#MTTA              ;VERIFY PC SAVED ON STACK
7202 031250  001401                          BEQ     MTTE                    ;BRANCH IF GOOD
7203 031252  104001                          ERROR   .+1             ;CPU ERROR
7204                                                                 ;INCORRECT PC ON STACK
7205 031254                          MTTE:
7206 031254  016767  151532  146532          MOV     SLOC00,14               ;RESTORE VECTOR 14
7207
7208 031262  012706  001000                  MOV     #STBOT,R6
7209
7211                                 ;
7213 031266                          MTTS:
7214
7215                                 ;       TEST T-BIT TRAPS WITH RTT
7216 031266  012706  001000                  MOV     #STBOT,R6               ;SETUP STACK
7217 031272  016767  146516  151512          MOV     14,SLOC00               ;SAVE OLD T-BIT VECTOR
7218 031300  012746  000020                  MOV     #20,-(R6)               ;PUSH T-BIT
7219 031304  012746  031322                  MOV     #MTTSA,-(R6)            ;SETUP ERROR TRAP VECTOR
7220 031310  012767  031326  146476          MOV     #MTTSB,14               ;SETUP NEW T-BIT VECTOR
7221 031316  000006                          RTT                             ;CAUSE A T BIT SET IN PSW
7222 031320  104001                          ERROR   .+1             ;CPU ERROR
7223                                                                 ;SHOULD NEVER BE EXECUTED
7224 031322  000240                  MTTSA:  NOP                             ;RTT WILL EXECUTE THIS INSTRUCTION
7225                                                                 ;WITH A T-BIT TRAP
7226 031324  104001                  MTTSQ:  ERROR   .+1             ;CPU ERROR
7227                                                                 ;DIDNT TAKE CORRECT TRAP
7228 031326  022706  000774          MTTSB:  CMP     #STBOT-4,R6             ;VERIFY SP DECIRMENT
```

```
********** DOUBLE OPERAND TESTS *****************

7229 031332  001401                          BEQ     MTTSD                   ;BRANCH IF GOOD
7230 031334  104001                          ERROR   +1                  ;CPU ERROR
7231                                                                     ;BAD SP
7232 031336  021627  031324        MTTSD:    CMP     (R6),#MTTSQ             ;VERIFY PC SAVED ON STACK
7233 031342  001401                          BEQ     MTTSE                   ;BRANCH IF GOOD
7234 031344  104001                          ERROR   +1                  ;CPU ERROR
7235                                                                     ;INCORRECT PC ON STACK
7236 031346                        MTTSE:
7237 031346  016767  151440  146440          MOV     SLOC00,14               ;RESTORE VECTOR 14
7238 031354  012706  001000                  MOV     #STBOT,R6
7239
7242 031360                        MTTR:
7243
7244                               ;         TEST OLD STATUS ON T-BIT TRAP
7245 031360  012706  001000                  MOV     #STBOT,R6               ;SETUP STACK
7246 031364  016767  146424  151420          MOV     14,SLOC00               ;SAVE OLD  VECTOR
7247 031372  012746  000020                  MOV     #20, (R6)               ;PUSH T-BIT
7248 031376  012746  031424                  MOV     #MTTRA,-(R6)            ;SETUP ERROR TRAP VECTOR
7249 031402  012767  031426  146404          MOV     #MTTRB,14               ;SETUP NEW T-BIT VECTOR
7250 031410  012767  000357  146360          MOV     #357,PS                 ;SET PRIORITY AND COND C
7251 031416  000277                          SCC
7252 031420  000002                          RTI
7253 031422  104001                          ERROR   +1                  ;CPU ERROR
7254                                                                     ;SHOULD NEVER EXECUTE
7255 031424  104001                MTTRA:    ERROR   +1                  ;CPU ERROR
7256                                                                     ;DIDNT TAKE CORRECT TRAP
7257 031426  026727  147344  000020  MTTRB:    CMP     STBOT-2,#20             ;VERIFY PSW ON STACK
7258 031434  001401                          BEQ     MTTRC                   ;BRANCH IF CORRECT STATUS
7259 031436  104001                          ERROR   +1                  ;CPU ERROR
7260                                                                     ;BAD STATUS ON STACK
7261 031440  012706  001000        MTTRC:    MOV     #STBOT,R6               ;SETUP STACK
7262 031444  012746  000377                  MOV     #377,-(R6)              ;PUSH T-BIT
7263 031450  012746  031476                  MOV     #MTTRD, (R6)            ;SETUP ERROR TRAP VECTOR
7264 031454  012767  031500  146332          MOV     #MTTRE,14               ;SETUP NEW T-BIT VECTOR
7265 031462  012767  000000  146306          MOV     #0,PS                   ;CLEAR PRIORITY
7266 031470  000257                          CCC                             ;CLEAR CONDITION CODES
7267 031472  000002                          RTI
7268 031474  104001                          ERROR   +1                  ;CPU ERROR
7269                                                                     ;SHOULD NEVER EXECUTE
7270 031476  104001                MTTRD:    ERROR   +1                  ;CPU ERROR
7271                                                                     ;DIDNT TAKE CORRECT TRAP
7272 031500  026727  147272  000377  MTTRE:    CMP     STBOT-2,#377            ;VERIFY OLD PSW ON STACK
7273 031506  001401                          BEQ     MTTRF                   ;BRANCH IF GOOD
7274 031510  104001                          ERROR   +1                  ;CPU ERROR
7275                                                                     ;OLD PSW INCORRECT
7276 031512                        MTTRF:
7277 031512  016767  151274  146274          MOV     SLOC00,14               ;RESTORE VECTOR
7278 031520  012706  001000                  MOV     #STBOT,R6
7279
7281                               ;
7283 031524                        MRT:
7284
7285                               ;         TEST RESERVED INST TRAP
7286 031524  012706  001000                  MOV     #STBOT,R6               ;SETUP STACK
7287 031530  016767  146254  151254          MOV     10,SLOC00               ;SAVE OLD  VECTOR
7288 031536  012767  031550  146244          MOV     #MRTB,10                ;SETUP NEW RESERVED VECTOR
7289 031544  000077                          77
```

```
********** DOUBLE OPERAND TESTS *****************

 7290 031546  104001                  MRTA:   ERROR   ·1              ;CPU ERROR
                                                                       ;DIDNT TAKE CORRECT TRAP
 7291
 7292 031550  022706  000774           MRTB:   CMP     #STBOT 4,R6             ;VERIFY SP DECRIMENT
 7293 031554  001401                          BEQ     MRTE                    ;BRANCH IF GOOD
 7294 031556  104001                          ERROR   ·1              ;CPU ERROR
                                                                       ;BAD PC ON STAC
 7295
 7296 031560  021627  031546           MRTE:   CMP     (R6),#MRTA              ;VERFY PROPER PC ON STACK
 7297 031564  001401                          BEQ     MRTF                    ;BRANCH IF GOOD
 7298 031566  104001                          ERROR   ·1              ;CPU ERROR
                                                                       ;INCORRECT PC ON STACK
 7299
 7300 031570                          MRTF:
 7301 031570  016767  151216  146212         MOV     SLOC00,10               ;RESTORE TRAP VECTOR
 7302 031576  012706  001000                  MOV     #STBOT,R6
 7303
 7305                                 ;
 7306                                 ;
 7308 031602                          MRTO:
 7309
 7310                                 ;       TEST OLD STATUS ON RESERVED INST TRAP
 7311 031602  012706  001000                  MOV     #STBOT,R6               ;SETUP STACK
 7312 031606  016767  146176  151176         MOV     10,SLOC00               ;SAVE OLD  VECTOR
 7313 031614  012767  031634  146166         MOV     #MRTOB,10               ;SETUP NEW VECTOR
 7314 031622  005067  146150                  CLR     PS                      ;CLEAR PRIORITY AND COND C
 7315 031626  000257                          CCC
 7316 031630  000077                          77
 7317 031632  104001                  MRTOA:  ERROR   ·1              ;CPU ERROR
                                                                       ;DIDNT TAKE CORRECT TRAP
 7318
 7319 031634  026727  147136  000000 MRTOB:  CMP     STBOT-2,#0              ;VERIFY PSW ON STACK
 7320 031642  001401                          BEQ     MRTOC                   ;BRANCH IF CORRECT STATUS
 7321 031644  104001                          ERROR   ·1              ;CPU ERROR
                                                                       ;BAD STATUS ON STACK
 7322
 7323 031646  012706  001000           MRTOC:  MOV     #STBOT,R6               ;SETUP STACK
 7324 031652  012767  031674  146130         MOV     #MRTOE,10               ;SET UP TRAP VECTOR
 7325 031660  012767  000357  146110         MOV     #357,PS                 ;SET PRIORITY
 7326 031666  000277                          SCC                             ;SET CONDITION CODES
 7327 031670  000077                          77                              ;RESERVED INSTRUCTION
 7328
 7329 031672  104001                  MRTOD:  ERROR   ·1              ;CPU ERROR
 7330                                                                   ;DIDNT TAKE CORRECT TRAP
 7331 031674  026727  147076  000357 MRTOE:  CMP     STBOT-2,#357            ;VERIFY OLD PSW ON STACK
 7332 031702  001401                          BEQ     MRTOF                   ;BRANCH IF GOOD
 7333 031704  104001                          ERROR   ·1              ;CPU ERROR
                                                                       ;OLD PSW INCORRECT
 7334
 7335 031706                          MRTOF:
 7336 031706  016767  151100  146074         MOV     SLOC00,10               ;RESOTRE TRAP VECTOR
 7337 031714  012706  001000                  MOV     #STBOT,R6
 7338
 7341 031720                          MTP:
 7342
 7343                                 ;       TEST TRAP INST
 7344 031720  012706  001000                  MOV     #STBOT,R6               ;SETUP STACK
 7345 031724  016767  146104  151060         MOV     34,SLOC00               ;SAVE OLD  VECTOR
 7346 031732  012767  031752  146074         MOV     #MTPB,34                ;SETUP NEW TRAP VECTOR
 7347 031740  005067  146032                  CLR     PS                      ;CLEAR PRIORITY ABND COND C
 7348 031744  000257                          CCC
 7349 031746  104400                          TRAP
 7350 031750  104001                  MTPR:   ERROR   ·1              ;CPU ERROR
```

```
********** DOUBLE OPERAND TESTS *****************

                                                                    ,DIDNT TAKE CORRECT TRAP
7351
7352 031752  022706  000774          MTPB:   CMP     #STBOT 4,R6            ;VERIFY SP DECRIMENT
7353 031756  001401                          BEQ     MTPQ                   ;BRANCH IF GOOD
7354 031760  104001                          ERROR   .1                  ;CPU ERROR
7355                                                                    ;BAD PC ON STACK
7356 031762  021627  031750          MTPQ:   CMP     (R6),#MTPR             ;VERFY PROPER PC ON STACK
7357 031766  001401                          BEQ     MTPF                   ;BRANCH IF GOOD
7358 031770  104001                          ERROR   .1                  ;CPU ERROR
7359                                                                    ;INCORRECT PC ON STACK
7360 031772                          MTPF:
7361 031772  016767  151014  146034          MOV     SLOC00,34              ;RESTORE VECTOR
7362 032000  012706  001000                  MOV     #STBOT,R6
7363
7365                                 ;
7366                                 ;
7368 032004                          MTPO:
7369
7370                                 ;       TEST OLD STATUS SAVED ON TRAP
7371 032004  012706  001000                  MOV     #STBOT,R6              ;SETUP STACK
7372 032010  016767  146020  150774          MOV     34,SLOC00              ;SAVE OLD  VECTOR
7373 032016  012767  032036  146010          MOV     #MTPOB,34              ;SETUP NEW TRAP VECTOR
7374 032024  005067  145746                  CLR     PS                     ;CLEAR PRIORITY AND COND C
7375 032030  000257                          CCC
7376 032032  104400                          TRAP
7377 032034  104001                  MTPOA:  ERROR   .1                  ;CPU ERROR
7378                                                                    ;DIDNT TAKE CORRECT TRAP
7379 032036  026727  146734  000000  MTPOB:  CMP     STBOT-2,#0             ;VERIFY PSW ON STACK
7380 032044  001401                          BEQ     MTPOC                  ;BRANCH IF CORRECT STATUS
7381 032046  104001                          ERROR   .1                  ;CPU ERROR
7382                                                                    ;BAD STATUS ON STACK
7383 032050  012706  001000          MTPOC:  MOV     #STBOT,R6              ;SETUP STACK
7384 032054  012767  032076  145752          MOV     #MTPOE,34              ;SET UP TRAP VECTOR
7385 032062  012767  000357  145706          MOV      357,PS                ;SET PRIORITY
7386 032070  000277                          SCC                            ;SET CONDITION CODES
7387 032072  104400                          TRAP                           ;ISSUE TRAP
7388 032074  104001                  MTPOD:  ERROR   .1                  ;CPU ERROR
7389                                                                    ;DIDNT TAKE CORRECT TRAP
7390 032076  026727  146674  000357  MTPOE:  CMP     STBOT-2,#357           ;VERIFY OLD PSW ON STACK
7391 032104  001401                          BEQ     MTPOF                  ;BRANCH IF GOOD
7392 032106  104001                          ERROR   .1                  ;CPU ERROR
7393                                                                    ;OLD PSW INCORRECT
7394 032110                          MTPOF:
7395 032110  016767  150676  145716          MOV     SLOC00,34              ;RESTORE TRAP VECTOR
7396 032116  012706  001000                  MOV     #STBOT,R6
7397
7399                                 ;
7400                                 ;
7402 032122                          MTPA:
7403
7404                                 ;       TEST ALL TRAP OPCODES - SELF MODIFYING
7405 032122  005003                          CLR     R3                     ;SETUP REGISTER TO INDICATE OPCODE
7406 032124  012706  001000                  MOV     #STBOT,R6              ;SETUP STACK
7407 032130  016767  145700  150654          MOV     34,SLOC00              ;SAVE OLD  VECTOR
7408 032136  016767  145642  150650          MOV     4,SLOC01               ;SAVE IN CASE OF HALT
7409 032144  012767  032174  145632          MOV     #MTPAH,4               ;SETUP HALT TRAP
7410 032152  012767  032176  145654          MOV     #MTPAA,34              ;SETUP NEW TRAP VECTOR
7411 032160  000167  000012                  JMP     MTPAA                  ;GO INTO LOOPING CODE
```

```
********** DOUBLE OPERAND TESTS *****************

7412
7413 032164  000000                    ;MTPAL:  HALT                    ;SET TO A ZERO
7414 032166  104001                             ERROR   +1              ;CPU ERROR
7415                                                                    ;TRAP INSTRUCTION FAILED TO TRAP
7416                                                                            ;EXAMINE ·CCODE AT LOCATION MTPAL:
7417 032170  000167  000002                      JMP     MTPAA                  ;ATTEMPT TO GO ON
7418
7419 032174  104001                    ;MTPAH:  ERROR   +1              ;CPU ERROR
7420                                                                    ;ERROR, EITHER CANT MODIFY LOCATION MTPAL
7421                                                                            ;OR TRAP INSTRUCTION FAILED
7422 032176                            MTPAA:
7423 032176  005203                             INC     R3                     ;GET NEXT OPCODE
7424
7425 032200  012706  001000                      MOV     #STBOT,R6              ;RESTORE STACK
7426 032204  020327  000400                      CMP     R3,#400                ;SEE IF LAST OPCODE
7427 032210  001406                              BEQ     MTPAE                  ;BRANCH IF DONE
7428 032212  012767  104400  177744              MOV     #104400,MTPAL          ;TRAP OPCODE INTO LOCATION
7429 032220  060367  177740                      ADD     R3,MTPAL               ;FORM TEST OPCODE
7430 032224  000757                              BR      MTPAL                  ;EXECUTE TEST
7431 032226                            MTPAE:
7432
7433 032226  016767  150560  145600              MOV     SLOC00,34
7434 032234  016767  150554  145542              MOV     SLOC01,4               ;RESTORE VECTORS
7435
7436 032242  012706  001000                      MOV     #STBOT,R6
7437
7439                                   ;
7440                                   ;
7442 032246                            MIOT:
7443
7444                                   ;        TEST IOT TRAP
7445 032246  012706  001000                      MOV     #STBOT,R6              ;SETUP STACK
7446 032252  016767  145542  150532              MOV     20,SLOC00              ;SAVE OLD  VECTOR
7447 032260  012767  032272  145532              MOV     #MIOTB,20              ;SETUP NEW IOT VECTOR
7448 032266  000004                              IOT
7449 032270  104001                    MIOTA:   ERROR   +1              ;CPU ERROR
7450                                                                    ;DIDNT TAKE CORRECT TRAP
7451 032272  022706  000774            MIOTB:   CMP     #STBOT-4,R6            ;VERIFY SP DECRIMENT
7452 032276  001401                              BEQ     MIOTD                  ;BRANCH IF GOOD
7453 032300  104001                              ERROR   +1              ;CPU ERROR
7454                                                                    ;BAD PC ON STACK
7455 032302  021627  032270            MIOTD:   CMP     (R6),#MIOTA            ;VERFY PROPER PC ON STACK
7456 032306  001401                              BEQ     MIOTF                  ;BRANCH IF GOOD
7457 032310  104001                              ERROR   +1              ;CPU ERROR
7458                                                                    ;INCORRECT PC ON STACK
7459 032312  016767  150474  145500  MIOTF:     MOV     SLOC00,20              ;RESTORE VECTOR
7460 032320  012706  001000                      MOV     #STBOT,R6
7461
7464 032324                            MITO:
7465
7466                                   ;        TEST OLD STATUS ON IOT TRAP
7467 032324  012706  001000                      MOV     #STBOT,R6              ;SETUP STACK
7468 032330  016767  145464  ·2454              MOV     20,SLOC00              ;SAVE OLD  VECTOR
7469 032336  012767  032356  ·45454             MOV     #MITOB,20              ;SETUP NEW IOT VECTOR
7470 032344  005067  145426                      CLR     PS                     ;CLEAR PRIORITY AND COND C
7471 032350  000257                              CCC
7472 032352  000004                              IOT
```

```
********** DOUBLE OPERAND TESTS *****************

7473 032354  104001                    MITOA:  ERROR   ·1              ;CPU ERROR
7474                                                                   ;DIDNT TAKE CORRECT TRAP
7475 032356  026727  146414  000000     MITOB:  CMP     STBOT 2,#0              ;VERIFY PSW ON STACK
7476 032364  001401                            BEQ     MITOC                   ;BRANCH IF CORRECT STATUS
7477 032366  104001                            ERROR   ·1              ;CPU ERROR
7478                                                                   ;BAD STATUS ON STACK
7479 032370  012706  001000             MITOC:  MOV     #STBOT,R6               ;SETUP STACK
7480 032374  012767  032416  145416             MOV     #MITOE,20               ;SET UP TRAP VECTOR
7481 032402  012767  000357  145366             MOV     #357,PS                 ;SET PRIORITY
7482 032410  000277                            SCC                             ;SET CONDITION CODES
7483 032412  000004                            IOT
7484 032414  104001                    MITOD:  ERROR   ·1              ;CPU ERROR
7485                                                                   ;DIDNT TAKE CORRECT TRAP
7486 032416  026727  146354  000357     MITOE:  CMP     STBOT 2,#357            ;VERIFY OLD PSW ON STACK
7487 032424  001401                            BEQ     MITOF                   ;BRANCH IF GOOD
7488 032426  104001                            ERROR   ·1              ;CPU ERROR
7489                                                                   ;OLD PSW INCORRECT
7490 032430                            MITOF:
7491 032430  016767  150356  145362             MOV     SLOC00,20               ;RESTORE VECTOR
7492 032436  012706  001000                    MOV     #STBOT,R6
7493
7495                                   ;
7497 032442                            METT:
7498
7499                                   ;       TEST EMULATOR TRAP INSTRUCTION  (EMT)
7500 032442  012706  001000                    MOV     #STBOT,R6               ;SETUP STACK
7501 032446  016767  145356  150336             MOV     30,SLOC00               ;SAVE OLD  VECTOR
7502 032454  012767  032510  145346             MOV     #METB,30                ;SETUP NEW EMT VECTOR
7503 032462  016767  145346  150324             MOV     34,SLOC01               ;SAVE TRAP VECTOR
7504 032470  012767  140776  145336             MOV     #$ERROR,34              ;SET UP TO HANDLE EMT ERROR
7505 032476  104000                            EMT
7506 032500  104400                    META:   TRAP                            ;TRAP ON ERROR
7507 032502  001057                            .WORD   559.
7508 032504  000001                            .WORD   1                       ;CPUERR
7509 032506  000001                            .WORD   1                       ;ERRTN
7510                                                                   ;DIDNT TAKE CORRECT TRAP
7511 032510  022706  000774             METB:   CMP     #STBOT-4,R6             ;VERIFY SP DECRIMENT
7512 032514  001401                            BEQ     METD                    ;BRANCH IF GOOD
7513 032516  104001                            ERROR   ·1              ;CPU ERROR
7514                                                                   ;BAD PC ON STACK
7515 032520  021627  032500             METD:   CMP     (R6),#META              ;VERFY PROPER PC ON STACK
7516 032524  001401                            BEQ     METF                    ;BRANCH IF GOOD
7517 032526  104001                            ERROR   ·1              ;CPU ERROR
7518                                                                   ;INCORRECT PC ON STACK
7519 032530  016767  150260  145276     METF:   MOV     SLOC01,34               ;RESTORE VECTOR
7520 032536  016767  150250  145264             MOV     SLOC00,30               ;RESTORE VECTOR
7521 032544  012706  001000                    MOV     #STBOT,R6
7522
7524                                   ;
7526 032550                            METO:
7527
7528                                   ;       TEST OLD STATUS ON EMT TRAP
7529 032550  012706  001000                    MOV     #STBOT,R6               ;SETUP STACK
7530 032554  016767  145250  150230             MOV     30,SLOC00               ;SAVE OLD  VECTOR
7531 032562  012767  032624  145240             MOV     #METOB,30               ;SETUP NEW EMT VECTOR
7532 032570  016767  145240  150216             MOV     34,SLOC01               ;SAVE TRAP VECTOR
7533 032576  012767  140776  145230             MOV     #$ERROR,34              ;SET UP TRAP VECTOR
```

```
********** DOUBLE OPERAND TESTS *****************

7534 032604  005067  145166                    CLR     PS                      ;CLEAR PRIORITY AND COND C
7535 032610  000257                            CCC
7536 032612  104000                            EMT
7537 032614  104400                  METOA:    TRAP
7538 032616  001062                            .WORD   562.
7539 032620  000001                            .WORD   1                       ;CPUERR
7540 032622  000001                            .WORD   1                       ;ERRTN
7541                                                                    ;DIDNT TAKE CORRECT TRAP
7542 032624  026727  146146  000000  METOB:    CMP     STBOT-2,#0              ;VERIFY PSW ON STACK
7543 032632  001401                            BEQ     METOC                   ;BRANCH IF CORRECT STATUS
7544 032634  104001                            ERROR   +1                      ;CPU ERROR
7545                                                                    ;BAD STATUS ON STACK
7546 032636  012706  001000            METOC:   MOV     #STBOT,R6               ;SETUP STACK
7547 032642  012767  032672  145160            MOV     #METOE,30               ;SET UP TRAP VECTOR
7548 032650  012767  000357  145120            MOV     #357,PS                 ;SET PRIORITY
7549 032656  000277                            SCC                             ;SET CONDITION CODES
7550 032660  104000                            EMT
7551 032662  104400                  METOD:    TRAP
7552 032664  001064                            .WORD   564.
7553 032666  000001                            .WORD   1                       ;CPUERR
7554 032670  000001                            .WORD   1                       ;ERRTN
7555                                                                    ;DIDNT TAKE CORRECT TRAP
7556 032672  026727  146100  000357  METOE:    CMP     STBOT-2,#357            ;VERIFY OLD PSW ON STACK
7557 032700  001401                            BEQ     METOF                   ;BRANCH IF GOOD
7558 032702  104001                            ERROR   +1                      ;CPU ERROR
7559                                                                    ;OLD PSW INCORRECT
7560 032704                            METOF:
7561 032704  016767  150104  145122            MOV     SLOC01,34               ;RESTORE VECTOR
7562 032712  016767  150074  145110            MOV     SLOC00,30               ;RESTORE VECTOR
7563 032720  012706  001000                    MOV     #STBOT,R6
7564
7566                                   ;
7567                                   ;
7569 032724                            MBT:
7570
7571                                   ;        TEST BPT TRAP
7572 032724  012706  001000                    MOV     #STBOT,R6               ;SETUP STACK
7573 032730  016767  145060  150054            MOV     14,SLOC00               ;SAVE OLD  VECTOR
7574 032736  012767  032750  145050            MOV     #MBTB,14                ;SETUP NEW BPT VECTOR
7575 032744  000003                            BPT
7576 032746  104001                  MBTA:     ERROR   +1                      ;CPU ERROR
7577                                                                    ;DIDNT TAKE CORRECT TRAP
7578 032750  022706  000774            MBTB:    CMP     #STBOT-4,R6             ;VERIFY SP DECRIMENT
7579 032754  001401                            BEQ     MBTD                    ;BRANCH IF GOOD
7580 032756  104001                            ERROR   +1                      ;CPU ERROR
7581                                                                    ;BAD PC ON STACK
7582 032760  021627  032746            MBTD:    CMP     (R6),#MBTA              ;VERFY PROPER PC ON STACK
7583 032764  001401                            BEQ     MBTF                    ;BRANCH IF GOOD
7584 032766  104001                            ERROR   +1                      ;CPU ERROR
7585                                                                    ;INCORRECT PC ON STACK
7586 032770  016767  150016  145016  MBTF:     MOV     SLOC00,14               ;RESTORE VECTOR
7587 032776  012706  001000                    MOV     #STBOT,R6
7588
7590                                   ;
7591                                   ;
7593 033002                            MBTO:
7594
```

```
********** DOUBLE OPERAND TESTS *****************

 7595                                           ;       TEST OLD STATUS ON BPT TRAP
 7596 033002  012706  001000                            MOV     #STBOT,R6               ;SETUP STACK
 7597 033006  016767  145002  '47776                    MOV     14,SLOC00               ;SAVE OLD  VECTOR
 7598 033014  012767  033034  144772                    MOV     #MBT0B,14               ;SETUP NEW BPT VECTOR
 7599 033022  005067  144750                            CLR     PS                      ;CLEAR PRIORITY AND COND C
 7600 033026  000257                                    CCC
 7601 033030  000003                                    BPT
 7602 033032  104001                            MBT0A:  ERROR   +1              ;CPU ERROR
                                                        ;DIDNT TAKE CORRECT TRAP
 7603
 7604 033034  026727  145736  000000          MBT0B:  CMP     STBOT-2,#0              ;VERIFY PSW ON STACK
 7605 033042  001401                                    BEQ     MBT0C           ;BRANCH IF CORRECT STATUS
 7606 033044  104001                                    ERROR   +1              ;CPU ERROR
 7607                                                   ;BAD STATUS ON STACK
 7608 033046  012706  001000                    MBT0C:  MOV     #STBOT,R6               ;SETUP STACK
 7609 033052  012767  033074  144734                    MOV     #MBT0E,14               ;SET UP TRAP VECTOR
 7610 033060  012767  000357  144710                    MOV     #357,PS                 ;SET PRIORITY
 7611 033066  000277                                    SCC                             ;SET CONDITION CODES
 7612 033070  000003                                    BPT
 7613 033072  104001                            MBT0D:  ERROR   +1              ;CPU ERROR
                                                        ;DIDNT TAKE CORRECT TRAP
 7614
 7615 033074  026727  145676  000357          MBT0E:  CMP     STBOT-2,#357            ;VERIFY OLD PSW ON STACK
 7616 033102  001401                                    BEQ     MBT0F                   ;BRANCH IF GOOD
 7617 033104  104001                                    ERROR   +1              ;CPU ERROR
 7618                                                   ;OLD PSW INCORRECT
 7619 033106                                    MBT0F:
 7620 033106  016767  147700  144700                    MOV     SLOC00,14               ;RESTORE VECTOR
 7621 033114  012706  001000                            MOV     #STBOT,R6
 7622
 7624                                           ;
 7625                                           ;
 7627 033120                                    MIL:
 7628
 7629                                           ;       TEST ILLEGAL JUMP INSTRUCTION TRAP
 7630 033120  012706  001000                            MOV     #STBOT,R6               ;SETUP STACK
 7631 033124  016767  144660  147660                    MOV     10,SLOC00               ;SAVE OLD  VECTOR
 7632 033132  012767  033146  144650                    MOV     #MILB,10                ;SETUP NEW ILLEGAL VECTOR
 7633 033140  005001                                    CLR     R1
 7634 033142  000101                                    JMP     R1                      ;**TEST INSTRUCTIO
 7635 033144  104001                            MILA:   ERROR   +1              ;CPU ERROR
                                                        ;DIDNT TAKE CORRECT TRAP
 7636
 7637 033146  022706  000774                    MILB:   CMP     #STBOT 4,R6             ;VERIFY SP DECRIMENT
 7638 033152  001401                                    BEQ     MILD                    ;BRANCH IF GOOD
 7639 033154  104001                                    ERROR   +1              ;CPU ERROR
 7640                                                   ;BAD PC ON STACK
 7641 033156  021627  033144                    MILD:   CMP     (R6),#MILA              ;VERFY PROPER PC ON STACK
 7642 033162  001401                                    BEQ     MILF                    ;BRANCH IF GOOD
 7643 033164  104001                                    ERROR   +1              ;CPU ERROR
 7644                                                   ;INCORRECT PC ON STACK
 7645 033166  016767  147620  144614          MILF:   MOV     SLOC00,10               ;RESTORE VECTOR
 7646 033174  012706  001000                            MOV     #STBOT,R6
 7647
 7650 033200                                    MIL0:
 7651
 7652                                           ;       TEST OLD STATUS ON ILLEGAL JUMP TRAP
 7653 033200  012706  001000                            MOV     #STBOT,R6               ;SETUP STACK
 7654 033204  016767  144600  147600                    MOV     10,SLOC00               ;SAVE OLD  VECTOR
 7655 033212  012767  033234  144570                    MOV     #MIL0B,10               ;SETUP NEW ILLEGAL VECTOR
```

```
********** DOUBLE OPERAND TESTS *****************

 7656 033220  005067  144552                CLR     PS                      ;CLEAR PRIORITY AND COND C
 7657 033224  000257                        CCC
 7658 033226  005001                        CLR     R1
 7659 033230  000101                        JMP     R1
 7660 033232  104001            MILOA:      ERROR   ;1              ;CPU ERROR
 7661                                                               ;DIDNT TAKE CORRECT TRAP
 7662 033234  026727  145536  000004  MILOB: CMP     STBOT 2,#4              ;VERIFY PSW ON STACK
 7663 033242  001401                        BEQ     MILOC                   ;BRANCH IF CORRECT STATUS
 7664 033244  104001                        ERROR   ;1              ;CPU ERROR
 7665                                                               ;BAD STATUS ON STACK
 7666 033246  012706  001000        MILOC:  MOV     #STBOT,R6               ;SETUP STACK
 7667 033252  012767  033274  144530        MOV     #MILOE,10               ;SET UP TRAP VECTOR
 7668 033260  012767  000357  144510        MOV     #357,PS                 ;SET PRIORITY
 7669 033266  000277                        SCC                             ;SET CONDITION CODES
 7670 033270  000101                        JMP     R1
 7671 033272  104001            MILOD:      ERROR   ;1              ;CPU ERROR
 7672                                                               ;DIDNT TAKE CORRECT TRAP
 7673 033274  026727  145476  000357  MILOE: CMP     STBOT-2,#357            ;VERIFY OLD PSW ON STACK
 7674 033302  001401                        BEQ     MILOF                   ;BRANCH IF GOOD
 7675 033304  104001                        ERROR   ;1              ;CPU ERROR
 7676                                                               ;OLD PSW INCORRECT
 7677 033306                    MILOF:
 7678 033306  016767  147500  144474        MOV     SLOC00,10               ;RESTORE VECTOR
 7679 033314  012706  001000                MOV     #STBOT,R6
 7680
 7682                           ;
 7683                           ;
 7685 033320                    MIALL:
 7686
 7687                           ;       TEST ILLEGAL JSR INSTRUCTION TRAP
 7688 033320  012706  001000                MOV     #STBOT,R6               ;SETUP STACK
 7689 033324  016767  144460  147460        MOV     10,SLOC00               ;SAVE OLD  VECTOR
 7690 033332  012767  033346  144450        MOV     #MIALLB,10              ;SETUP NEW ILLEGAL VECTOR
 7691 033340  005003                        CLR     R3
 7692 033342  004303                        JSR     R3,R3
 7693 033344  104001            MIALLA:     ERROR   ;1              ;CPU ERROR
 7694                                                               ;DIDNT TAKE CORRECT TRAP
 7695 033346  022706  000774        MIALLB: CMP     #STBOT-4,R6             ;VERIFY SP DECRIMENT
 7696 033352  001401                        BEQ     MIALLD                  ;BRANCH IF GOOD
 7697 033354  104001                        ERROR   ;1              ;CPU ERROR
 7698                                                               ;BAD PC ON STACK
 7699 033356  021627  033344        MIALLD: CMP     (R6),#MIALLA            ;VERFY PROPER PC ON STACK
 7700 033362  001401                        BEQ     MIALLF                  ;BRANCH IF GOOD
 7701 033364  104001                        ERROR   ;1              ;CPU ERROR
 7702                                                               ;INCORRECT PC ON STACK
 7703 033366  016767  147420  144414  MIALLF: MOV     SLOC00,10               ;RESTORE VECTOR
 7704 033374  012706  001000                MOV     #STBOT,R6
 7705
 7707                           ;
 7708                           ;
 7710 033400                    MJSI:
 7711
 7712                           ;       TEST OLD STATUS ON ILLEGAL JSR TRAP
 7713 033400  012706  001000                MOV     #STBOT,R6               ;SETUP STACK
 7714 033404  016767  144400  147400        MOV     10,SLOC00               ;SAVE OLD  VECTOR
 7715 033412  012767  033434  144370        MOV     #MJSIB,10               ;SETUP NEW VECTOR
 7716 033420  005067  144352                CLR     PS                      ;CLEAR PRIORITY AND COND C
```

```
********** DOUBLE OPERAND TESTS ******************

 7717 033424  000257                          CCC
 7718 033426  005003                          CLR     R3
 7719 033430  004303                          JSR     R3,R3
 7720 033432  104001                MJSIA:    ERROR   +1              ;CPU ERROR
 7721                                                                 ;DIDNT TAKE CORRECT TRAP
 7722 033434  026727  145336  000004 MJSIB:    CMP     STBOT 2,#4              ;VERIFY PSW ON STACK
 7723 033442  001401                          BEQ     MJSIC                   ;BRANCH IF CORRECT STATUS
 7724 033444  104001                          ERROR   +1              ;CPU ERROR
 7725                                                                 ;BAD STATUS ON STACK
 7726 033446  012706  001000        MJSIC:    MOV     #STBOT,R6               ;SETUP STACK
 7727 033452  012767  033474  144330         MOV     #MJSIE,10               ;SET UP TRAP VECTOR
 7728 033460  012767  000357  144310         MOV     #357,PS                 ;SET PRIORITY
 7729 033466  000277                          SCC                             ;SET CONDITION CODES
 7730 033470  004303                          JSR     R3,R3
 7731 033472  104001                MJSID:    ERROR   +1              ;CPU ERROR
 7732                                                                 ;DIDNT TAKE CORRECT TRAP
 7733 033474  026727  145276  000357 MJSIE:    CMP     STBOT 2,#357            ;VERIFY OLD PSW ON STACK
 7734 033502  001401                          BEQ     MJSIF                   ;BRANCH IF GOOD
 7735 033504  104001                          ERROR   +1              ;CPU ERROR
 7736                                                                 ;OLD PSW INCORRECT
 7737 033506                        MJSIF:
 7738 033506  016767  147300  144274         MOV     SLOC00,10               ;RESTORE VECTOR
 7739 033514  012706  001000                  MOV     #STBOT,R6
 7740
 7742                               ;
 7743                               ;
 7745 033520                        IOXXX:
 7746                               ;       I/O TIME OUT TEST
 7747 033520  005067  144242                  CLR     CPEREG                  ;CLEAR CPU ERROR REGISTER
 7748 033524  016767  144254  147260         MOV     4,SLOC00                ;SAVE VECTOR
 7749 033532  012767  033554  144244         MOV     #2$,4                   ;SET UP VECTOR TO HANDLE NXM
 7750 033540  012767  030000  144230         MOV     #30000,PS               ;INIT THE PSW TO A KNOWN STATE
 7751 033546  005737  177700                  TST     @#177700                ;TRY TO ACCESS HARDWARE ADDRESS
 7752                                                                         ;FOR GENERAL PURPOSE REG 0. THIS
 7753                                                                         ;IS NOT IMPLEMENTED ON KDJ11
 7754                                                                         ;SHOULD CAUSE TIME OUT.
 7755 033552  104001                1$:       ERROR   +1              ;CPU ERROR
 7756 033554  022767  000020  144204 2$:       CMP     #BIT04,CPEREG           ;IS CPU ERROR REGISTER CORRECT?
 7757 033562  001401                          BEQ     3$
 7758 033564  104001                          ERROR   +1              ;CPU ERROR
 7759 033566  022627  033552        3$:       CMP     (SP)+,#1$               ;CHECK THAT STACK CONTAINS CORRECT ADDR.
 7760 033572  001401                          BEQ     4$                      ;
 7761 033574  104001                          ERROR   +1              ;CPU ERROR
 7762 033576  022627  030000        4$:       CMP     (SP)+,#30000            ;IS THE PSW OK?
 7763 033602  001401                          BEQ     5$                      ;
 7764 033604  104001                          ERROR   +1              ;CPU ERROR
 7765 033606  005067  144154        5$:       CLR     CPEREG                  ;CLEAR THE CPU ERROR REGISTER
 7766 033612  016767  147174  144164         MOV     SLOC00,4                ;RESTORE VECTOR
 7767
 7770 033620                        ODDXX:
 7779
 7780                               ;       ODD ADDRESS/ILLEGAL INST FETCH TRAP TEST
 7781                               ;;**************************************************************
                                    ;THIS PROGRAM GENERATES AN ODD ADDRESS IN THE PC. THE KDJ11 SHOULD
                                    ;TRAP THROUGH ADDR 4
 7782 033620  005067  144142                  CLR     CPEREG                  ;INIT THE CPU ERROR REG
 7783 033624  016767  144154  147160         MOV     4,SLOC00                ;SAVE VECTOR
```

```
********** DOUBLE OPERAND TESTS *****************

7784 033632  012767  033666  144144          MOV     #2$,4                   ;SET UP VECTOR TO HANDLE ODD ADDR TRAP
7785 033640  016767  144142  147146          MOV     6,SLOC01                ;SAVE VECTOR
7786 033646  005067  144134                  CLR     6                       ;INIT VECTOR
7787 033652  012746  030000                  MOV     #30000,-(SP)            ;PUSH A KNOWN PSW ON THE STACK
7788 033656  012746  033665                  MOV     #1$+1,-(SP)             ;PUSH AN ODD NUMBER ON THE STACK
7789 033662  000002                          RTI                             ;POP ODD ADDRESS OFF STACK INTO PC
7790                                                                         ;SHOULD TRAP HERE
7791 033664  104001                  1$:     ERROR   +1              ;CPU ERROR
7792 033666  022767  000100  144072  2$:     CMP     #BIT06,CPEREG           ;IS CPU ERROR REGISTER CORRECT?
7793 033674  001401                          BEQ     3$
7794 033676  104001                          ERROR   +1              ;CPU ERROR
7795 033700  022726  033665          3$:     CMP     #1$+1,(SP)+             ;IS STACK CONTENTS CORRECT?
7796 033704  001401                          BEQ     4$                      ;
7797 033706  104001                          ERROR   +1              ;CPU ERROR
7798 033710  022726  030000          4$:     CMP     #30000,(SP)+            ;IS PSW CORRECT ON STACK?
7799 033714  001401                          BEQ     5$                      ;
7800 033716  104001                          ERROR   +1              ;CPU ERROR
7801 033720  005067  144042          5$:     CLR     CPEREG                  ;CLEAR CPU ERROR REG
7802                                         ;
7803                                         ;NOW WE'LL TRY TO FETCH AN INSTRUCTION FROM AN INTERNAL REGISTER.
7804                                         ;THIS SHOULD CAUSE A TRAP TO ADDR 4 AND SET BIT 6 IN THE CPU
7805                                         ;ERROR REGISTER.
7806                                         ;
7807 033724  012767  033752  144052          MOV     #7$,4                   ;LOAD VECTOR WITH TRAP HANDLER ADDR
7808 033732  012767  030340  144046          MOV     #30340,6                ;LOAD VEC WITH PSW VALUE ON TRAP
7809 033740  005067  144032                  CLR     PS                      ;CLEAR THE PSW
7810 033744  000167  144026                  JMP     PS                      ;********TEST INSTRUCTION********
7811                                                                         ;TRY INSTRUCTION FETCH FROM INTERNAL
7812                                                                         ;REGISTER-- SHOULD TRAP VIA ADDR 4
7813 033750  104001                  6$:     ERROR   +1              ;CPU ERROR
7814 033752  016701  144020          7$:     MOV     PS,R1                   ;SAVE CONTENTS OF PSW IN R1
7815 033756  022767  000100  144002          CMP     #BIT06,CPEREG           ;IS CPU ERROR REGISTER CORRECT?
7816 033764  001401                          BEQ     8$                      ;
7817 033766  104001                          ERROR   +1              ;CPU ERROR
7818 033770  022726  177776          8$:     CMP     #PS,(SP)+               ;IS STACK CONTENTS CORRECT?
7819 033774  001401                          BEQ     9$                      ;
7820 033776  104001                          ERROR   +1              ;CPU ERROR
7821 034000  022726  000000          9$:     CMP     #0,(SP)+                ;IS STACK CONTENTS CORRECT?
7822 034004  001401                          BEQ     10$                     ;
7823 034006  104001                          ERROR   +1              ;CPU ERROR
7824 034010  022701  000340          10$:    CMP     #340,R1                 ;WAS PSW LOADED PROPERLY ON TRAP?
7825 034014  001401                          BEQ     11$                     ;
7826 034016  104001                          ERROR   +1              ;CPU ERROR
7827 034020  005067  143742          11$:    CLR     CPEREG                  ;CLEAR CPU ERROR REGISTER
7828 034024  016767  146762  143752          MOV     SLOC00,4                ;RESTORE VECTOR
7829 034032  016767  146756  143746          MOV     SLOC01,6                ;       "
7830
7833 034040                          RXXX:
7834
7835                                 ;       RED ZONE TRAP TEST
7836 034040  013767  000004  146744          MOV     @#4,SLOC00              ;SAVE VECTOR
7837 034046  012737  034076  000004          MOV     #2$,@#4                 ;SET UP VECTOR
7838 034054  012706  000777                  MOV     #777,R6                 ;SET UP THE STACK WITH ODD ADDRESS
7839 034060  005067  143702                  CLR     CPEREG                  ;CLEAR THE CPU ERROR REGISTER
7840 034064  005067  143706                  CLR     PSW                     ;CLEAR THE PSW
7841 034070  005737  177700                  TST     @#177700                ;ACCESS NON-EXISTANT I/O ADDRESS
7842 034074  104001                  1$:     ERROR   +1              ;CPU ERROR
```

```
********** DOUBLE OPERAND TESTS *****************

 7843 034076  022706  000000          2$:     CMP     #0,SP                   ;IS R6 CORRECT?
 7844 034102  001401                          BEQ     3$                      ;BRANCH IF YES
 7845 034104  104001                          ERROR   +1              ;CPU ERROR
 7846 034106  022726  034074          3$:     CMP     #1$,(SP)+               ;IS DATA AT ADDR 0 CORRECT?
 7847 034112  001401                          BEQ     4$                      ;BRANCH IF YES
 7848 034114  104001                          ERROR   +1              ;CPU ERROR
 7849 034116  022716  000000          4$:     CMP     #0,(SP)                 ;IS PSW DATA IN ADDR 2 CORRECT?
 7850 034122  001401                          BEQ     5$                      ;BRANCH IF YES
 7851 034124  104001                          ERROR   +1              ;CPU ERROR
 7852 034126  022767  000124  143632  5$:     CMP     #124,CPEREG             ;IS CPU ERROR REGISTER CORRECT?
 7853 034134  001401                          BEQ     6$                      ;BRANCH IF YES
 7854 034136  104001                          ERROR   +1              ;CPU ERROR
 7855 034140  005067  143622          6$:     CLR     CPEREG                  ;CLEAR CPU ERROR REGISTER
 7856 034144  012706  001000                  MOV     #STBOT,SP               ;RESTORE STACK
 7857 034150  016737  146636  000004          MOV     SLOC00,@#4              ;RESTORE VECTOR
 7858 034156  005037  000000                  CLR     @#0                     ;RESTORE ADDR 0
 7859 034162  005037  000002                  CLR     @#2                     ;RESTORE ADDR 2
 7860
 7862
 7864
 7865 034166                          PIRXXX:
 7866                                 ;       TEST PIRQ REGISTER RESPONSE
 7867 034166  016767  143612  146616          MOV     4,SLOC00                ;SAVE CONTENTS OF VECTORS
 7868 034174  012767  034222  143602          MOV     #PIRQNX,4               ;SETUP INTERRUPT VECTOR
 7869 034202  005067  143560                  CLR     CPEREG                  ;CLEAR CPU ERROR REGISTER
 7870 034206  005737  177772                  TST     @#PIRQ                  ;LOOK FOR REPLY FROM PIRQ
 7871 034212  016767  146574  143564          MOV     SLOC00,4                ;RESTORE VECTORS
 7872 034220  000401                          BR      PIRTEX                  ;IF IT RESPONDS, CONTINUE TESTING.
 7873                                                                         ;ERROR! NO RESPONSE FROM PIRQ REG
 7874 034222  104001                  PIRQNX: ERROR   +1              ;CPU ERROR
 7875 034224                          PIRTEX:
 7876
 7877 034224                          PIR1:
 7878                                 ;       TEST PIRQ REGISTER DATA
 7879 034224  013767  000240  146560          MOV     @#PIRQVEC,SLOC00        ;SAVE PIRQ VECTORS
 7880 034232  013767  000242  146554          MOV     @#PIRQVEC+2,SLOC01      ;       "
 7881 034240  012737  034450  000240          MOV     #UNXPIR,@#PIRQVEC       ;SET UP PIRQ VECTOR FOR UNEXPECTED INTERRUPT
 7882 034246  012737  000340  000242          MOV     #PR7,@#PIRQVEC+2        ;
 7883 034254  012703  001000                  MOV     #1000,R3                ;PUT 1000 IN R3: START TESTING
 7884                                                                         ;BITS IN PIRQ REG BY FLOATING
 7885                                                                         ;A BIT THROUGH BITS 9-15.
 7886 034260  012704  034324                  MOV     #PIRTBL,R4              ;SET UP R4 AS A POINTER TO EXPECTED
 7887                                                                         ;ENCODED PRIORITY LEVELS IN PIRTBL
 7888 034264  000237                  1$:     SPL     7                       ;DON'T ALLOW INTERRUPTS.
 7889 034266  005037  177772                  CLR     @#PIRQ                  ;CLEAR OUT THE PIRQ
 7890 034272  023724  177772                  CMP     @#PIRQ,(R4)+            ;IS PIRQ OK??
 7891 034276  001401                          BEQ     2$                      ;BRANCH IF OK
 7892                                                                         ;ERROR; PIRQ REG WAS NOT CLEAR
 7893 034300  104001                          ERROR   +1              ;CPU ERROR
 7894 034302  010337  177772          2$:     MOV     R3,@#PIRQ               ;SET A BIT IN PIRQ REGISTER
 7895 034306  023724  177772                  CMP     @#PIRQ,(R4)+            ;COMPARE THE ENCODED PRIORITY BITS
 7896                                                                         ;WITH DATA IN THE TABLE (PIRTBL)
 7897 034312  001401                          BEQ     3$                      ;BRANCH IF ITS OK
 7898                                                                         ;ERROR; ENCODED PRIORITY LEVELS ARE
 7899                                                                         ;NOT CORRECT
 7900 034314  104001                          ERROR   +1              ;CPU ERROR
 7901 034316  006303                  3$:     ASL     R3                      ;FLOAT A "1" THRU BITS 9-15
```

```
********** DOUBLE OPERAND TESTS **********

7902 034320  103370                    BCC     2$                      ;IF CARRY BIT ISN'T SET, DO AGAIN.
7903 034322  000410                    BR      EXPIR1                  ;GO TO EXIT TEST.
7904
7905 034324  000000            PIRTBL. .WORD   0
7906 034326  001042                    .WORD   1042
7907 034330  002104                    .WORD   2104
7908 034332  004146                    .WORD   4146
7909 034334  010210                    .WORD   10210
7910 034336  020252                    .WORD   20252
7911 034340  040314                    .WORD   40314
7912 034342  100356                    .WORD   100356
7913
7914 034344                     EXPIR1:
7915
7925
7926 034344                     PIR2:
7927                            ;       TEST PIRQ REGISTER LEVEL ENCODING
7928                            ;;*****************************************************************
                                ;THIS TEST IS TO CHECK THAT THE HIGHEST PRIORITY LEVEL SET IN THE
                                ;PROGRAM INTERRUPT REQUEST BITS IS REFLECTED IN THE ENCODED PROGRAM
                                ;INTERRUPT ACTIVE BITS.
7929 034344  000237                    SPL     7                       ;SHUT OFF INTERRUPTS
7930 034346  005037  177772            CLR     @#PIRQ                  ;CLEAR PIRQ REGISTER
7931 034352  012767  000001  146450    MOV     #1,FLAG                 ;SETUP END OF LOOP SIGNAL
7932 034360  012703  177000            MOV     #177000,R3              ;SET UP DESIRED PATTERN IN R3
7933 034364  012704  034430            MOV     #PITBL1,R4              ;SETUP R4 AS A TABLE POINTER
7934 034370  010337  177772    1$:     MOV     R3,@#PIRQ               ;SET PATERN IN PIRQ REGISTER
7935
7936 034374  023724  177772            CMP     @#PIRQ,(R4)+            ;COMPARE PATTERN IN PIRQ WITH PATTERN IN
7937                                                                    ;EXPECTED PATTERN TABLE.
7938 034400  001012                    BNE     2$                      ;GO TO ERROR IF NOT THE SAME
7939 034402  005737  003030            TST     @#FLAG                  ;IS THIS THE LAST TIME THROUGH??
7940 034406  001421                    BEQ     PIR2EX                  ;YES, IF FLAG IS ZERO
7941 034410  006003                    ROR     R3                      ;SHIFT PATTERN TO RIGHT FOR NEXT TEST
7942 034412  042703  000777            BIC     #777,R3                 ;STRIP OFF BITS  8-0
7943 034416  001364                    BNE     1$                      ;IF R3 IS NOT = 0 GO DO THE NEXT PATTERN
7944 034420  005037  003030            CLR     @#FLAG                  ;CLR THE FLAG TO INDICATE LAST
7945                                                                    ;TIME THROUGH THE LOOP
7946 034424  000761                    BR      1$                      ;GO THROUGH LOOP ONCE MORE
7947                                                                    ;ERROR; PATTERNS WERE NOT THE SAME
7948 034426  104001            2$:     ERROR   +1              ;CPU ERROR
7949
7950 034430  177356  077314  037252  PITBL1: .WORD   177356,77314,37252,17210,7146,3104,1042,0
     034436  017210  007146  003104
     034444  001042  000000
7951
7952 034450                     UNXPIR:                                 ;UNEXPECTED PIRQ INTERRUPT
7953 034450  104001                    ERROR   +1              ;CPU ERROR
7954
7955 034452                     PIR2EX:
7963
7964 034452                     PIR3:
7965                            ;       TEST PIRQ INTERRUPTS
7966                            ;;*****************************************************************
                                ;THIS TEST CHECKS THAT EACH PROGRAM INTERRUPT OCCURS PROPERLY
7967 034452  012703  001000            MOV     #1000,R3                ;SETUP R3 AS A WORKING REGISTER
7968 034456  012737  034522  000240    MOV     #PIRRTN,@#PIRQVEC       ;SET UP INTERRUPT ROUTINE AT PIR VECTOR
```

```
********** DOUBLE OPERAND TESTS *****************

7969 034464  012737  000340  000242          MOV     #340,@#PIRQVEC+2        ;
7970 034472  005004                          CLR     R4                      ;INITIALIZE R4 AS EXPECTED DATA HOLDER.
7971 034474  005724                  PI1:    TST     (R4)+                   ;INCREMENT R4 BY 2
7972 034476  050337  177772                  BIS     R3,@#PIRQ               ;SET PIRQ TO INTERRUPT
7973 034502  000230                          SPL     0                       ;ENABLE INTERRUPT TO OCCUR
7974                                                                         ;ERROR! INTERRUPT DID NOT OCCUR
7975 034504  104001                          ERROR   +1              ;CPU ERROR
7976                                                                         ;ERROR! INTERRUPT OCCURRED IN WRONG SEQUENCE
7977 034506  104001                  PI2:    ERROR   +1              ;CPU ERROR
7978 034510  062706  000004          PI3:    ADD     #4,SP                   ;CLEAN UP THE STACK
7979 034514  006303                          ASL     R3                      ;SHIFT R3 TO LEFT FOR NEXT INTERRUPT
7980 034516  103366                          BCC     PI1                     ;END IF CARRY BIT IS SET
7981 034520  000412                          BR      PIR3EX                  ;ON TO THE NEXT TEST
7982
7983 034522  013705  177772          PIRRTN: MOV     @#PIRQ,R5               ;MOVE THE CONTENTS OF PIRQ REG TO R5
7984 034526  005037  177772                  CLR     @#PIRQ                  ;KILL PIRQ INTERRUPT.
7985 034532  042705  177761                  BIC     #177761,R5              ;MASK OFF ALL BITS EXCEPT 1-4.
7986 034536  020405                          CMP     R4,R5                   ;DID THE CORRECT LEVEL INTERRUPT OCCUR?
7987 034540  001362                          BNE     PI2                     ;IF NOT; GO TO ERROR.
7988 034542  000167  177742                  JMP     PI3                     ;RETURN FROM SUCCESSFUL INTERRUPT,GET
7989                                                                         ;READY FOR THE NEXT ONE.
7990
7991 034546                          PIR3EX:
7992
8001
8002 034546                          PIR4:
8003                                 ;       TEST PIRQ VS PSW INTERRUPT LEVEL
8004                                 ;;*****************************************************************
                                     ;THIS TEST IS TO ENSURE THAT PIRQ CANNOT INTERRUPT WHEN PSW INTERRUPT
                                     ;LEVEL IS SET TO BLOCK THEM OUT.
8005 034546  005037  177772                  CLR     @#PIRQ                  ;CLEAR THE PIRQ
8006 034552  005037  177776                  CLR     @#PSW                   ;CLEAR THE PSW.
8007 034556  000237                          SPL     7                       ;BLOCK INTERRUPTS FROM BEING SERVICED.
8008 034560  012703  001000                  MOV     #1000,R3                ;USE R3 AS A WORKING REGISTER.
8009 034564  012737  034616  000240          MOV     #2$,@#PIRQVEC           ;SETUP INTERRUPT VECTORS
8010 034572  012737  000340  000242          MOV     #340,@#PIRQVEC+2        ;
8011 034600  010337  177772          1$:     MOV     R3,@#PIRQ               ;SET INTERRUPT LEVEL IN PIRQ.
8012 034604  006303                          ASL     R3                      ;SHIFT A "1" LEFT TO INCREASE INTERRUPT
8013                                                                         ;PRIORITY LEVEL.
8014 034606  103374                          BCC     1$                      ;IF C BIT NOT SET THEN DO IT AGAIN.
8015 034610  005037  177772                  CLR     @#PIRQ                  ;ELSE CLEAR THE PIRQ
8016 034614  000403                          BR      3$                      ;EXIT TEST.
8017
8018 034616  005037  177772          2$:     CLR     @#PIRQ                  ;STOP PIRQ FROM INTERRUPTING.
8019                                                                         ;ERROR! NO INTERRUPTS SHOULD OCCUR.
8020 034622  104001                          ERROR   +1              ;CPU ERROR
8021 034624                          3$:
8022
8036
8037 034624                          PIR5:
8038                                 ;       TEST PIRQ INTERRUPTS OCCUR AT PROPER LEVEL
8039                                 ;;*****************************************************************
                                     ;THIS TEST ENSURES THAT INTERRUPTS OCCUR AT THE PROPER LEVEL
                                     ;THE PRIORITY LEVEL IS INITIALLY SET TO PRI6. THE PIRQ THEN STARTS INTERRUPTING
                                     ;AT PRI1. NO INTERRUPTS SHOULD OCCUR UNTIL THE INTERRUPTING LEVEL EXCEEDS THE
                                     ;PRIORITY LEVEL IN THE PSW. AFTER EACH LEVEL OF INTERRUPT HAS BEEN TRIED; THE
                                     ;PSW PRIORITY LEVEL IS LOWERED ONE NOTCH, AND THE CYCLE REPEATS UNTIL PSW
```

D.2


```
********** DOUBLE OPERAND TESTS *****************

                                        ;PRIORITY LEVEL 0 IS REACHED. INTERRUPTS AT PSW PRIORITY LEVEL 0 HAVE BEEN DONE
                                        ;PREVIOUSLY.
8040 034624  005037  177772                     CLR     @#PIRQ                  ;CLR THE PIRQ
8041 034630  012737  034742  000240             MOV     #10$,@#PIRQVEC          ;SET UP INTERRUPT VECTORS
8042 034636  012737  000340  000242             MOV     #PR7,@#PIRQVEC+2        ;
8043 034644  012705  035020                     MOV     #PITBL2,R5              ;SETUP R5 AS A TABLE POINTER.
8044 034650  012737  000006  114144             MOV     #6,@#DCOUNT             ;INIT LOCATION DCOUNT:
8045 034656  005004                     1$:     CLR     R4                      ;INITIALIZE R4 AS A COUNTER.
8046 034660  012703  001000                     MOV     #1000,R3                ;USE R3 AS A WORKING REGISTER.
8047 034664  012567  143106                     MOV     (R5)+,PS                ;MOVE PRIORITY LEVEL TO PSW
8048 034670  005724                     3$:     TST     (R4)+                   ;INCREMENT R4 BY 2
8049 034672  010337  177772                     MOV     R3,@#PIRQ               ;SET INTERRUPT LEVEL IN PIRQ.
8050                                    ;*******INTERRUPTS WILL HAPPEN HERE*********
8051 034676  013701  177776                     MOV     @#PS,R1                 ;SAVE PS IN R1          !ONLY EXECUTED
8052 034702  013702  177772                     MOV     @#PIRQ,R2               ;SAVE PIRQ IN R2        !   IF NO
8053 034706  042701  177437                     BIC     #177437,R1              ;CLEAR EXTRANEOUS BITS  ! INTERRUPT
8054 034712  042702  177437                     BIC     #177437,R2              ;                       !    HAS
8055 034716  020102                             CMP     R1,R2                   ;R1 SHOULD BE >= R2     !  OCCURRED
8056 034720  002001                             BGE     4$                      ;IF IT IS GO ON         !------
8057                                                                            ;ERROR! SHOULD HAVE INTERRUPTED.
8058 034722  104001                             ERROR   +1                      ;CPU ERROR
8059 034724  006303                     4$:     ASL     R3                      ;SHIFT A "1" LEFT UNTIL INTERRUPT LEVEL
8060                                                                            ;IS REACHED.
8061 034726  103360                             BCC     3$                      ;BRANCH BACK IF CARRY IS NOT SET.
8062 034730  005337  114144                     DEC     @#DCOUNT                ;LOWER INTERRUPT PRIORITY LEVEL
8063 034734  001350                             BNE     1$                      ;IF DCOUNT= 0 , WE'RE DONE.
8064 034736  000167  000072                     JMP     PIR5EX                  ;JUMP TO END OF THIS TEST.
8065
8066                                    ;PIRQ INTERRUPT SERVICE ROUTINE
8067                                    ;
8068
8069 034742  013746  177772             10$:    MOV     @#PIRQ,-(SP)            ;SAVE PIRQ DATA ON STACK
8070 034746  005037  177772                     CLR     @#PIRQ                  ;SHUT OFF PIRQ INTERRUPTS
8071 034752  016601  000004                     MOV     4(SP),R1                ;GET OLD PSW. SAVE IN R1.
8072 034756  011602                             MOV     (SP),R2                 ;GET PIRQ FROM STACK.
8073 034760  042702  177437                     BIC     #177437,R2              ;CLEAR UNWANTED BITS FROM R2
8074 034764  042701  177437                     BIC     #177437,R1              ;CLEAR UNWANTED BITS FROM R1
8075 034770  020102                             CMP     R1,R2                   ;R2 SHOULD BE > R1.
8076 034772  100401                             BMI     20$                     ;GO CHECK SEQUENCE OF INTERRUPT
8077                                                                            ;ERROR! PRIORITY OF INTERRUPT WAS NOT
8078                                                                            ;HIGH ENOUGH. SHOULDN'T HAVE OCCURRED.
8079 034774  104001                             ERROR   +1                      ;CPU ERROR
8080 034776  012602                     20$:    MOV     (SP)+,R2                ;POP OLD PIRQ OFF THE STACK.
8081 035000  042702  177761                     BIC     #177761,R2              ;CLEAR OFF EXTRANEOUS BITS.
8082 035004  020402                             CMP     R4,R2                   ;SHOULD BE EQUAL.
8083 035006  001401                             BEQ     21$                     ;IF THEY ARE EQUAL, CLEAN UP STACK AND
8084                                                                            ;GET READY FOR THE NEXT INTERRUPT
8085                                                                            ;ELSE
8086                                                                            ;ERROR! INTERRUPT OCCURRED OUT OF SEQUENCE.
8087 035010  104001                             ERROR   +1                      ;CPU ERROR
8088 035012  012716  034724             21$:    MOV     #4$,(SP)                ;PUT RETURN ADDRESS ON THE STACK.
8089 035016  000002                             RTI                             ;RETURN FROM INTERRUPT. RESTORE PSW.
8090
8091 035020  000300             PITBL2: .WORD   300                     ;PRIORITY LEVEL 6
8092 035022  000240                     .WORD   240                     ;PRIORITY LEVEL 5
8093 035024  000200                     .WORD   200                     ;PRIORITY LEVEL 4
8094 035026  000140                     .WORD   140                     ;PRIORITY LEVEL 3
```

E12


```
********** DOUBLE OPERAND TESTS *****************

8095 035030  000100                        .WORD   100             ;PRIORITY LEVEL 2
8096 035032  000040                        .WORD   40              ;PRIORITY LEVEL 1
8097
8098 035034                          PIR5EX:
8099
8114
8115 035034                          PIR6:
8116                                 ;       TEST THAT PIRQS ARE SERVICED IN CORRECT ORDER
8117                                 ;;*****************************************************************
                                     ;THIS TEST CHECKS THAT ALL PIRQ INTERRUPTS ARE SERVICED
                                     ;IN THE CORRECT DECENDING ORDER. I.E. IRQ6 IS NOT SERVICED
                                     ;BEFORE IRQ7 ETC. THE PIRQ IS LOADED WITH ALL INTERRUPTS SIMULTANEOUSLY
                                     ;TURNED ON.  THE PSW PRIORITY LEVEL IS THEN LOWERED TO 0. EACH INTERRUPT
                                     ;SHOULD BE SERVICED IN DECENDING ORDER. EACH TIME AN INTERRUPT OCCURRS, THE PSW
                                     ;IS LOADED WITH PRIORITY LEVEL 7 WHICH STOPS THE PIRQ FROM FURTHER INTERRUPTS.
                                     ;AFTER A CORRECT INTERRUPT, AN RTI IS EXECUTED, AND THE PSW PRIORITY LEVEL
                                     ;RETRUNS TO ZERO ALLOWING THE NEXT LOWER INTERRUPT TO OCCUR.
8118 035034  005037  177772                CLR     @#PIRQ                  ;CLEAR OUT THE PIRQ
8119 035040  000237                        SPL     7                       ;NO INTERRUPTS WILL BE SERVICED.
8120 035042  012737  035102  000240        MOV     #10$,@#PIRQVEC          ;SET UP VECTORS
8121 035050  012737  000340  000242        MOV     #PR7,@#PIRQVEC+2        ;
8122 035056  012703  177000                MOV     #177000,R3              ;SETUP DESIRED PATTERN IN R3
8123 035062  010337  177772                MOV     R3,@#PIRQ               ;MOVE PATTERN IN R3 TO PIRQ.
8124 035066  012704  000016                MOV     #16,R4                  ;PRELOAD R4 AS A DECREMENTING COUNTER.
8125 035072  000230                        SPL     0                       ;LET THE PIRQ INTERRUPT
8126                                 ;*********INTERRUPTS OCCUR HERE!!!*********************
8127 035074  005704                        TST     R4                      ;IS R4 ZERO? IT SHOULD BE. THIS
8128                                                                        ;INSTRUCTION SHOULDN'T BE EXECUTED
8129                                                                        ;UNTIL ALL INTERRUPTS HAVE OCCURRED.
8130 035076  001417                        BEQ     PIR6EX                  ;GO TO NEXT TEST.
8131                                                                        ;ERROR! ALL INTERRUPTS DID NOT OCCUR.
8132 035100  104001                        ERROR   +1                      ;CPU ERROR
8133 035102  013702  177772          10$:  MOV     @#PIRQ,R2               ;SAVE THE PIRQ IN R2
8134 035106  042702  177761                BIC     #177761,R2              ;STRIP OFF EXTRANEOUS BITS.
8135 035112  020402                        CMP     R4,R2                   ;SHOULD BE EQUAL.
8136 035114  001401                        BEQ     15$                     ;SETUP FOR NEXT INTERRUPT.
8137                                                                        ;ERROR! INTERRUPT WAS SERVICED OUT OF ORDER
8138 035116  104001                        ERROR   +1                      ;CPU ERROR
8139 035120  005744                15$:  TST     -(R4)                   ;DECREMENT R4 BY 2
8140 035122  006003                        ROR     R3                      ;CHANGE PATTERN IN R3; LOWER INTERRUPT PRIORITY
8141 035124  042703  000777                BIC     #777,R3                 ;STRIP OFF UNNEEDED BITS
8142 035130  010337  177772                MOV     R3,@#PIRQ               ;LOWER INTERRUPT PRIORITY LEVEL.
8143 035134  000002                        RTI                             ;
8144
8145 035136  016737  145650  000240  PIR6EX: MOV   SLOC00,@#PIRQVEC       ;RESTORE VECTORS
8146 035144  016737  145644  000242        MOV     SLOC01,@#PIRQVEC+2              ;       "
8147                                                                        ;
8149
8152                                 .SBTTL  MEMORY MANAGEMENT TESTS
8153                                 ;*****************************************************************
8154                                 ;*****************************************************************
8155                                 ;                       BEGIN MMU TESTING
8156                                 ;*****************************************************************
8157                                 ;*****************************************************************
8159 035152                          TSMMU1:
8160                                 ;       STATUS REGISTER TEST
8161 035152  005067  142610                CLR     CPEREG                  ;CLEAR CPU ERROR REGISTER
```

MEMORY MANAGEMENT TESTS

```
8162 035156  005037  177572                    CLR     @#177572                ;TURN MMU OFF
8163 035162  005037  003030                    CLR     @#FLAG                  ;CLEAR MMU TRAP FLAG
8164 035166  013746  000004                    MOV     @#4,-(SP)               ;SAVE OLD VECTOR
8165 035172  012737  137720  000004           MOV     #ADDTRP,@#4             ;SETUP NEW VECTOR
8166 035200  005005                            CLR     R5                      ;CLEAR FLAG
8167 035202  013701  177572                    MOV     @#177572,R1             ; TEST MMR0
8168 035206  013701  177574                    MOV     @#177574,R1             ; TEST MMR1
8169 035212  013701  177576                    MOV     @#177576,R1             ; TEST MMR2
8170 035216  013701  172516                    MOV     @#172516,R1             ; TEST MMR3
8171 035222  012637  000004                    MOV     (SP)+,@#4               ;RESTORE VECTOR
8172 035226  020527  000000                    CMP     R5,#0                   ;DID WE TRAP
8173 035232  001401                            BEQ     1$                      ;NO, THEN BRANCH
8174 035234  104002                            ERROR   +2              ;MMU ERROR
8175                                                                   ;YES, GO TO ERROR
8176 035236                            1$:
8179 035236                            TSMMU2:
3180                                   ;       ADDRESS TEST OF PARS,PDRS, AND FP REGS
8181 035236  005067  142524                    CLR     CPEREG                  ;CLEAR CPU ERROR REGISTER
8182 035242  005037  177572                    CLR     @#177572                ;MMU OFF
8183 035246  005037  003030                    CLR     @#FLAG                  ;CLEAR MMU TRAP FLAG
8184 035252  013746  000244                    MOV     @#244,-(SP)             ;SAVE FP VECTOR
8185 035256  013746  000246                    MOV     @#246,-(SP)
8186 035262  013746  000004                    MOV     @#4,-(SP)               ;SAVE TIME OUT VECTOR
8187 035266  012737  000246  000244           MOV     #246,@#244              ;SETUP NEW FP VECTOR
8188 035274  012737  000002  000246           MOV     #2,@#246                ;
8189 035302  012737  137720  000004           MOV     #ADDTRP,@#4             ;SETUP NEW TIME OUT VECTOR
8190 035310  005005                            CLR     R5                      ;CLEAR TIMEOUT FLAG
8191 035312  012700  172200                    MOV     #172200,R0              ;LOAD ALL PARS AND PDRS WITH ZERO
8192 035316  005020                    1$:     CLR     (R0)+                   ;
8193 035320  020027  172400                    CMP     R0,#172400              ;
8194 035324  001374                            BNE     1$                      ;
8195 035326  012700  177600                    MOV     #177600,R0              ;
8196 035332  005020                    2$:     CLR     (R0)+                   ;
8197 035334  020027  177700                    CMP     R0,#177700              ;
8198 035340  001374                            BNE     2$                      ;
8199 035342  170127  000200                    LDFPS   #200                    ;
8200 035346  012700  003052                    MOV     #FLOAT,R0               ;LOAD AC0-AC5 WITH 0
8201 035352  005020                            CLR     (R0)+                   ;
8202 035354  005020                            CLR     (R0)+                   ;
8203 035356  005020                            CLR     (R0)+                   ;
8204 035360  005020                            CLR     (R0)+                   ;
8205 035362  012700  003052                    MOV     #FLOAT,R0               ;
8206 035366  172410                            LDD     (R0),AC0                ;
8207 035370  172510                            LDD     (R0),AC1                ;
8208 035372  172610                            LDD     (R0),AC2                ;
8209 035374  172710                            LDD     (R0),AC3                ;
8210 035376  174004                            STD     AC0,AC4                 ;
8211 035400  174005                            STD     AC0,AC5                 ;
8212 035402  174500                    3$:     DIVD    AC0,AC1                 ;LOAD FEC WITH 4 AND FEA WITH @#3$
8213 035404  170337  003062                    STST    @#FLO                   ;CHECK FEC FOR 4 AND FEA FOR @#3$
8214 035410  012704  003062                    MOV     #FLO,R4                 ;
8215 035414  022427  000004                    CMP     (R4)+,#4                ;
8216 035420  001401                            BEQ     21$                     ;
8217 035422  104002                            ERROR   +2              ;MMU ERROR
8218                                                                   ;
8219 035424  021427  035402            21$:    CMP     (R4),@#3$               ;
8220 035430  001401                            BEQ     22$                     ;
```

MEMORY MANAGEMENT TESTS

```
8221 035432  104002                        ERROR   +2              ;MMU ERROR
8222                                        ;
8223 035434  012704  172200        22$:     MOV     #172200,R4              ;CHECK EACH PAR, PDR FOR 0 THEN
8224 035440  012701  000001                MOV     #1,R1                   ;WRITE A UNIQUE NUMBER TO IT
8225 035444  010102                4$:     MOV     R1,R2                   ;
8226 035446  072227  000010                ASH     #10,R2                  ;
8227 035452  021427  000000                CMP     (R4),#0                 ;
8228 035456  001401                        BEQ     5$                      ;
8229 035460  104002                        ERROR   +2              ;MMU ERROR
8230                                        ;
8231 035462  010224                5$:     MOV     R2,(R4)+                ;
8232 035464  005201                        INC     R1                      ;
8233 035466  020427  172400                CMP     R4,#172400              ;
8234 035472  001364                        BNE     4$                      ;
8235 035474  012704  177600                MOV     #177600,R4              ;
8236 035500  010102                6$:     MOV     R1,R2                   ;
8237 035502  072227  000010                ASH     #10,R2                  ;
8238 035506  021427  000000                CMP     (R4),#0                 ;
8239 035512  001401                        BEQ     7$                      ;
8240 035514  104002                        ERROR   +2              ;MMU ERROR
8241                                        ;
8242 035516  010224                7$:     MOV     R2,(R4)+                ;
8243 035520  005201                        INC     R1                      ;
8244 035522  020427  177700                CMP     R4,#177700              ;
8245 035526  001364                        BNE     6$                      ;
8246 035530  012704  003062                MOV     #FLO,R4                 ;CHECK AC5 FOR ALL ZEROES THEN LOAD A 6
8247 035534  012703  003052                MOV     #FLOAT,R3               ;
8248 035540  174014                        STD     AC0,(R4)                ;
8249 035542  172405                        LDD     AC5,AC0                 ;
8250 035544  174013                        STD     AC0,(R3)                ;
8251 035546  012702  000004                MOV     #4,R2                   ;
8252 035552  022327  000000        8$:     CMP     (R3)+,#0                ;
8253 035556  001401                        BEQ     9$                      ;
8254 035560  104002                        ERROR   +2              ;MMU ERROR
8255                                        ;
8256 035562  005302                9$:     DEC     R2                      ;
8257 035564  001372                        BNE     8$                      ;
8258 035566  012703  003052                MOV     #FLOAT,R3               ;
8259 035572  012713  000006                MOV     #6,(R3)                 ;
8260 035576  172413                        LDD     (R3),AC0                ;
8261 035600  174005                        STD     AC0,AC5                 ;
8262 035602  172404                        LDD     AC4,AC0                 ;CHECK AC4 FOR ALL ZEROES THEN LOAD A 5
8263 035604  174013                        STD     AC0,(R3)                ;
8264 035606  012702  000004                MOV     #4,R2                   ;
8265 035612  022327  000000        10$:    CMP     (R3)+,#0                ;
8266 035616  001401                        BEQ     11$                     ;
8267 035620  104002                        ERROR   +2              ;MMU ERROR
8268                                        ;
8269 035622  005302                11$:    DEC     R2                      ;
8270 035624  001372                        BNE     10$                     ;
8271 035626  012703  003052                MOV     #FLOAT,R3               ;
8272 035632  012713  000005                MOV     #5,(R3)                 ;
8273 035636  172413                        LDD     (R3),AC0                ;
8274 035640  174004                        STD     AC0,AC4                 ;
8275 035642  012702  000004                MOV     #4,R2                   ;CHECK AC0 FOR ALL ZEROES THEN LOAD A 1
8276 035646  022427  000000        12$:    CMP     (R4)+,#0                ;
8277 035652  001401                        BEQ     13$                     ;
```

H12


```
8278 035654  104002                    ERROR   +2              ;MMU ERROR
8279                                    ;
8280 035656  005302          13$:      DEC     R2              ;
8281 035660  001372                    BNE     12$             ;
8282 035662  012713  000001            MOV     #1,(R3)         ;
8283 035666  172413                    LDD     (R3),AC0        ;
8284 035670  012704  003062            MOV     #FLO,R4         ;CHECK AC1 FOR ALL ZEROES THEN LOAD A 2
8285 035674  012702  000004            MOV     #4,R2           ;
8286 035700  174114                    STD     AC1,(R4)        ;
8287 035702  022427  000000    14$:    CMP     (R4)+,#0        ;
8288 035706  001401                    BEQ     15$             ;
8289 035710  104002                    ERROR   +2              ;MMU ERROR
8290                                    ;
8291 035712  005302          15$:      DEC     R2              ;
8292 035714  001372                    BNE     14$             ;
8293 035716  012713  000002            MOV     #2,(R3)         ;
8294 035722  172513                    LDD     (R3),AC1        ;
8295 035724  012704  003062            MOV     #FLO,R4         ;CHECK AC2 FOR ALL ZEROES THEN LOAD A 3
8296 035730  012702  000004            MOV     #4,R2           ;
8297 035734  174214                    STD     AC2,(R4)        ;
8298 035736  022427  000000    16$:    CMP     (R4)+,#0        ;
8299 035742  001401                    BEQ     17$             ;
8300 035744  104002                    ERROR   +2              ;MMU ERROR
8301                                    ;
8302 035746  005302          17$:      DEC     R2              ;
8303 035750  001372                    BNE     16$             ;
8304 035752  012713  000003            MOV     #3,(R3)         ;
8305 035756  172613                    LDD     (R3),AC2        ;
8306 035760  012704  003062            MOV     #FLO,R4         ;CHECK AC3 FOR ALL ZEROES THEN LOAD A 4
8307 035764  012702  000004            MOV     #4,R2           ;
8308 035770  174314                    STD     AC3,(R4)        ;
8309 035772  022427  000000    18$:    CMP     (R4)+,#0        ;
8310 035776  001401                    BEQ     19$             ;
8311 036000  104002                    ERROR   +2              ;MMU ERROR
8312                                    ;
8313 036002  005302          19$:      DEC     R2              ;
8314 036004  001372                    BNE     18$             ;
8315 036006  012713  000004            MOV     #4,(R3)         ;
8316 036012  172713                    LDD     (R3),AC3        ;
8317 036014  012704  003062            MOV     #FLO,R4         ;CHECK FPS FOR 100204 THEN LOAD IT WITH 200
8318 036020  170214                    STFPS   (R4)            ;
8319 036022  022714  100204            CMP     #100204,(R4)    ;
8320 036026  001401                    BEQ     20$             ;
8321 036030  104002                    ERROR   +2              ;MMU ERROR
8322                                    ;
8323 036032  170127  000200    20$:    LDFPS   #200            ;
8324 036036  012704  172200            MOV     #172200,R4      ;CHECK PDR, PAR FOR UNIQUE NUMBERS
8325 036042  012701  000001            MOV     #1,R1           ;
8326 036046  010102          23$:      MOV     R1,R2           ;
8327 036050  072227  000010            ASH     #10,R2          ;
8328 036054  022402                    CMP     (R4)+,R2        ;
8329 036056  001401                    BEQ     24$             ;
8330 036060  104002                    ERROR   +2              ;MMU ERROR
8331                                    ;
8332 036062  005201          24$:      INC     R1              ;
8333 036064  020427  172400            CMP     R4,#172400      ;
8334 036070  001366                    BNE     23$             ;
```

MEMORY MANAGEMENT TESTS

```
8335 036072  012704  177600            MOV     #177600,R4              ;
8336 036076  010102              25$:   MOV     R1,R2                   ;
8337 036100  072227  000010            ASH     #10,R2                  ;
8338 036104  022402                    CMP     (R4)+,R2                ;
8339 036106  001401                    BEQ     26$                     ;
8340 036110  104002                    ERROR   +2                      ;MMU ERROR
8341                                                                   ;
8342 036112  005201              26$:   INC     R1                      ;
8343 036114  020427  177700            CMP     R4,#177700              ;
8344 036120  001366                    BNE     25$                     ;
8345 036122  012701  003052            MOV     #FLOAT,R1               ;CHECK AC5 FOR #6
8346 036126  012704  003062            MOV     #FLO,R4                 ;
8347 036132  174014                    STD     AC0,(R4)                ;
8348 036134  172405                    LDD     AC5,AC0                 ;
8349 036136  174011                    STD     AC0,(R1)                ;
8350 036140  022127  000006            CMP     (R1)+,#6                ;
8351 036144  001401                    BEQ     27$                     ;
8352 036146  104002                    ERROR   +2                      ;MMU ERROR
8353                                                                   ;
8354 036150  012703  000003      27$:  MOV     #3,R3                   ;
8355 036154  022127  000000      28$:  CMP     (R1)+,#0                ;
8356 036160  001401                    BEQ     29$                     ;
8357 036162  104002                    ERROR   +2                      ;MMU ERROR
8358                                                                   ;
8359 036164  005303              29$:   DEC     R3                      ;
8360 036166  001372                    BNE     28$                     ;
8361 036170  012701  003052            MOV     #FLOAT,R1               ;CHECK AC4 FOR #5
8362 036174  172404                    LDD     AC4,AC0                 ;
8363 036176  174011                    STD     AC0,(R1)                ;
8364 036200  022127  000005            CMP     (R1)+,#5                ;
8365 036204  001401                    BEQ     30$                     ;
8366 036206  104002                    ERROR   +2                      ;MMU ERROR
8367                                                                   ;
8368 036210  012703  000003      30$:  MOV     #3,R3                   ;
8369 036214  022127  000000      31$:  CMP     (R1)+,#0                ;
8370 036220  001401                    BEQ     32$                     ;
8371 036222  104002                    ERROR   +2                      ;MMU ERROR
8372                                                                   ;
8373 036224  005303              32$:   DEC     R3                      ;
8374 036226  001372                    BNE     31$                     ;
8375 036230  022427  000001            CMP     (R4)+,#1                ;CHECK AC0 FOR #1
8376 036234  001401                    BEQ     33$                     ;
8377 036236  104002                    ERROR   +2                      ;MMU ERROR
8378                                                                   ;
8379 036240  012703  000003      33$:  MOV     #3,R3                   ;
8380 036244  022427  000000      34$:  CMP     (R4)+,#0                ;
8381 036250  001401                    BEQ     35$                     ;
8382 036252  104002                    ERROR   +2                      ;MMU ERROR
8383                                                                   ;
8384 036254  005303              35$:   DEC     R3                      ;
8385 036256  001372                    BNE     34$                     ;
8386 036260  012701  003052            MOV     #FLOAT,R1               ;CHECK AC1 FOR #2
8387 036264  174111                    STD     AC1,(R1)                ;
8388 036266  022127  000002            CMP     (R1)+,#2                ;
8389 036272  001401                    BEQ     36$                     ;
8390 036274  104002                    ERROR   +2                      ;MMU ERROR
8391                                                                   ;
```

J12

MEMORY MANAGEMENT TESTS

```
8392 036276  012703  000003          36$:    MOV     #3,R3                   ;
8393 036302  022127  000000          37$:    CMP     (R1)+,#0                ;
8394 036306  001401                         BEQ     38$                     ;
8395 036310  104002                         ERROR   +2              ;MMU ERROR
8396                                                                ;
8397 036312  005303                 38$:    DEC     R3                      ;
8398 036314  001372                         BNE     37$                     ;
8399 036316  012701  003052                 MOV     #FLOAT,R1               ;CHECK AC2 FOR #3
8400 036322  174211                         STD     AC2,(R1)                ;
8401 036324  022127  000003                 CMP     (R1)+,#3                ;
8402 036330  001401                         BEQ     39$                     ;
8403 036332  104002                         ERROR   +2              ;MMU ERROR
8404                                                                ;
8405 036334  012703  000003          39$:    MOV     #3,R3                   ;
8406 036340  022127  000000          40$:    CMP     (R1)+,#0                ;
8407 036344  001401                         BEQ     41$                     ;
8408 036346  104002                         ERROR   +2              ;MMU ERROR
8409                                                                ;
8410 036350  005303                 41$:    DEC     R3                      ;
8411 036352  001372                         BNE     40$                     ;
8412 036354  012701  003052                 MOV     #FLOAT,R1               ;CHECK AC3 FOR #4
8413 036360  174311                         STD     AC3,(R1)                ;
8414 036362  022127  000004                 CMP     (R1)+,#4                ;
8415 036366  001401                         BEQ     42$                     ;
8416 036370  104002                         ERROR   +2              ;MMU ERROR
8417                                                                ;
8418 036372  012703  000003          42$:    MOV     #3,R3                   ;
8419 036376  022127  000000          43$:    CMP     (R1)+,#0                ;
8420 036402  001401                         BEQ     44$                     ;
8421 036404  104002                         ERROR   +2              ;MMU ERROR
8422                                                                ;
8423 036406  005303                 44$:    DEC     R3                      ;
8424 036410  001372                         BNE     43$                     ;
8425 036412  020527  000000                 CMP     R5,#0                   ;IS TIME OUT FLAG 0
8426 036416  001401                         BEQ     45$                     ;YES GO ON
8427 036420  104002                         ERROR   +2              ;MMU ERROR
8428                                                                ;NO GO TO ERROR
8429 036422  012637  000004          45$:    MOV     (SP)+,@#4               ;RESTORE TIME OUT VECTOR
8430 036426  012637  000246                 MOV     (SP)+,@#246             ;RESTORE FP VECTOR
8431 036432  012637  000244                 MOV     (SP)+,@#244             ;
8432
8435 036436                         TSMMU3:
8436                                ;       WRITE ALL PARS/PDRS WITH ONES THEN ZEROS
8437 036436  005037  177572                 CLR     @#177572                ;MMU OFF
8438 036442  005037  003030                 CLR     @#FLAG                  ;CLEAR MMU ABORT FLAG
8439 036446  012703  172200                 MOV     #172200,R3              ;LOAD ALL PARS AND PDRS WITH ONES
8440 036452  012723  177777          1$:    MOV     #177777,(R3)+           ;
8441 036456  020327  172400                 CMP     R3,#172400              ;
8442 036462  001373                         BNE     1$                      ;
8443 036464  012703  177600                 MOV     #177600,R3              ;
8444 036470  012723  177777          2$:    MOV     #177777,(R3)+           ;
8445 036474  020327  177700                 CMP     R3,#177700              ;
8446 036500  001373                         BNE     2$                      ;
8447 036502  012703  172200                 MOV     #172200,R3              ;CHECK SPDRS FOR ONES
8448 036506  022327  177416          3$:    CMP     (R3)+,#177416           ;
8449 036512  001401                         BEQ     4$                      ;
8450 036514  104002                         ERROR   +2              ;MMU ERROR
```

MEMORY MANAGEMENT TESTS

```
8451                                                    ;
8452 036516  020327  172240        4$:   CMP   R3,#172240           ;
8453 036522  001371                      BNE   3$                    ;
8454 036524  022327  177777        5$:   CMP   (R3)+,#177777        ;CHECK SPARS FOR ONES
8455 036530  001401                      BEQ   6$                    ;
8456 036532  104002                      ERROR +2                    ;MMU ERROR
8457                                                    ;
8458 036534  020327  172300        6$:   CMP   R3,#172300           ;
8459 036540  001371                      BNE   5$                    ;
8460 036542  022327  177416        7$:   CMP   (R3)+,#177416        ;CHECK KPDRS FOR ONES
8461 036546  001401                      BEQ   8$                    ;
8462 036550  104002                      ERROR +2                    ;MMU ERROR
8463                                                    ;
8464 036552  020327  172340        8$:   CMP   R3,#172340           ;
8465 036556  001371                      BNE   7$                    ;
8466 036560  022327  177777        9$:   CMP   (R3)+,#177777        ;CHECK KPARS FOR ONES
8467 036564  001401                      BEQ   10$                   ;
8468 036566  104002                      ERROR +2                    ;MMU ERROR
8469                                                    ;
8470 036570  020327  172400        10$:  CMP   R3,#172400           ;
8471 036574  001371                      BNE   9$                    ;
8472 036576  012703  177600              MOV   #177600,R3            ;CHECK UPDRS FOR ONES
8473 036602  022327  177416        11$:  CMP   (R3)+,#177416        ;
8474 036606  001401                      BEQ   12$                   ;
8475 036610  104002                      ERROR +2                    ;MMU ERROR
8476                                                    ;
8477 036612  020327  177640        12$:  CMP   R3,#177640           ;
8478 036616  001371                      BNE   11$                   ;
8479 036620  022327  177777        13$:  CMP   (R3)+,#177777        ;CHECK UPARS FOR ONES
8480 036624  001401                      BEQ   14$                   ;
8481 036626  104002                      ERROR +2                    ;MMU ERROR
8482                                                    ;
8483 036630  020327  177700        14$:  CMP   R3,#177700           ;
8484 036634  001371                      BNE   13$                   ;
8485 036636  012703  172200              MOV   #172200,R3            ;LOAD ALL PARS AND PDRS WITH ZEROES
8486 036642  012723  000000        15$:  MOV   #0,(R3)+              ;
8487 036646  020327  172400              CMP   R3,#172400            ;
8488 036652  001373                      BNE   15$                   ;
8489 036654  012703  177600              MOV   #177600,R3            ;
8490 036660  012723  000000        16$:  MOV   #0,(R3)+              ;
8491 036664  020327  177700              CMP   R3,#177700            ;
8492 036670  001373                      BNE   16$                   ;
8493 036672  012703  172200              MOV   #172200,R3            ;CHECK ALL PARS AND PDRS FOR ZEROES
8494 036676  022327  000000        17$:  CMP   (R3)+,#0              ;
8495 036702  001401                      BEQ   18$                   ;
8496 036704  104002                      ERROR +2                    ;MMU ERROR
8497                                                    ;
8498 036706  020327  172400        18$:  CMP   R3,#172400           ;
8499 036712  001371                      BNE   17$                   ;
8500 036714  012703  177600              MOV   #177600,R3            ;
8501 036720  022327  000000        19$:  CMP   (R3)+,#0              ;
8502 036724  001401                      BEQ   20$                   ;
8503 036726  104002                      ERROR +2                    ;MMU ERROR
8504                                                    ;
8505 036730  020327  177700        20$:  CMP   R3,#177700           ;
8506 036734  001371                      BNE   19$                   ;
8507
```

MEMORY MANAGEMENT TESTS

```
8510 036736                                TSMMU4:
8511                                       ;       TEST FOR ADJACENT SHORTS IN PARS/PDRS
8512 036736  005037  177572                        CLR     @#177572                ;MMU OFF
8513 036742  005067  144062                        CLR     FLAG                    ;CLEAR MMU ABORT FLAG
8514 036746  012700  172200                        MOV     #172200,R0              ;LOAD SPDRS WITH ALTERNATING PATTERN
8515 036752  012720  052404              1$:       MOV     #52404,(R0)+            ;
8516 036756  012720  125012                        MOV     #125012,(R0)+           ;
8517 036762  020027  172240                        CMP     R0,#172240              ;
8518 036766  001371                                BNE     1$                      ;
8519 036770  012720  125252              2$:       MOV     #125252,(R0)+           ;LOAD SPARS WITH ALTERNATING PATTERN
8520 036774  012720  052525                        MOV     #52525,(R0)+            ;
8521 037000  020027  172300                        CMP     R0,#172300              ;
8522 037004  001371                                BNE     2$                      ;
8523 037006  012720  052404              3$:       MOV     #52404,(R0)+            ;LOAD KPDRS WITH ALTERNATING PATTERN
8524 037012  012720  125012                        MOV     #125012,(R0)+           ;
8525 037016  020027  172340                        CMP     R0,#172340              ;
8526 037022  001371                                BNE     3$                      ;
8527 037024  012720  125252              4$:       MOV     #125252,(R0)+           ;LOAD KPARS WITH ALTERNATING PATTERN
8528 037030  012720  052525                        MOV     #52525,(R0)+            ;
8529 037034  020027  172400                        CMP     R0,#172400              ;
8530 037040  001371                                BNE     4$                      ;
8531 037042  012700  177600                        MOV     #177600,R0              ;LOAD UPDRS WITH ALTERNATING PATTERN
8532 037046  012720  052404              5$:       MOV     #52404,(R0)+            ;
8533 037052  012720  125012                        MOV     #125012,(R0)+           ;
8534 037056  020027  177640                        CMP     R0,#177640              ;
8535 037062  001371                                BNE     5$                      ;
8536 037064  012720  125252              6$:       MOV     #125252,(R0)+           ;LOAD UPARS WITH ALTERNATING PATTERN
8537 037070  012720  052525                        MOV     #52525,(R0)+            ;
8538 037074  020027  177700                        CMP     R0,#177700              ;
8539 037100  001371                                BNE     6$                      ;
8540                                       ;
8541 037102  012703  172200                        MOV     #172200,R3              ;CHECK SPDRS
8542 037106  022327  052404              7$:       CMP     (R3)+,#52404            ;
8543 037112  001401                                BEQ     8$                      ;
8544 037114  104002                                ERROR   +2              ;MMU ERROR
8545                                               ;
8546 037116  022327  125012              8$:       CMP     (R3)+,#125012           ;
8547 037122  001401                                BEQ     9$                      ;
8548 037124  104002                                ERROR   +2              ;MMU ERROR
8549                                               ;
8550 037126  020327  172240              9$:       CMP     R3,#172240              ;
8551 037132  001365                                BNE     7$                      ;
8552 037134  022327  125252              10$:      CMP     (R3)+,#125252           ;CHECK SPARS
8553 037140  001401                                BEQ     11$                     ;
8554 037142  104002                                ERROR   +2              ;MMU ERROR
8555                                               ;
8556 037144  022327  052525              11$:      CMP     (R3)+,#52525            ;
8557 037150  001401                                BEQ     12$                     ;
8558 037152  104002                                ERROR   +2              ;MMU ERROR
8559                                               ;
8560 037154  020327  172300              12$:      CMP     R3,#172300              ;
8561 037160  001365                                BNE     10$                     ;
8562 037162  022327  052404              13$:      CMP     (R3)+,#52404            ;CHECK KPDRS
8563 037166  001401                                BEQ     14$                     ;
8564 037170  104002                                ERROR   +2              ;MMU ERROR
8565                                               ;
8566 037172  022327  125012              14$:      CMP     (R3)+,#125012           ;
```

MEMORY MANAGEMENT TEST

```
8567 037176  001401                        BEQ     15$                     ;
8568 037200  104002                        ERROR   +2                      ;MMU ERROR
8569                                                                       ;
8570 037202  020327  172340        15$:    CMP     R3,#172340              ;
8571 037206  001365                        BNE     13$                     ;
8572 037210  022327  125252        16$:    CMP     (R3)+,#125252           ;CHECK KPARS
8573 037214  001401                        BEQ     17$                     ;
8574 037216  104002                        ERROR   +2                      ;MMU ERROR
8575                                                                       ;
8576 037220  022327  052525        17$:    CMP     (R3)+,#52525            ;
8577 037224  001401                        BEQ     18$                     ;
8578 037226  104002                        ERROR   +2                      ;MMU ERROR
8579                                                                       ;
8580 037230  020327  172400        18$:    CMP     R3,#172400              ;
8581 037234  001365                        BNE     16$                     ;
8582 037236  012703  177600                MOV     #177600,R3              ;CHECK UPDRS
8583 037242  022327  052404        19$:    CMP     (R3)+,#52404            ;
8584 037246  001401                        BEQ     20$                     ;
8585 037250  104002                        ERROR   +2                      ;MMU ERROR
8586                                                                       ;
8587 037252  022327  125012        20$:    CMP     (R3)+,#125012           ;
8588 037256  001401                        BEQ     21$                     ;
8589 037260  104002                        ERROR   +2                      ;MMU ERROR
8590                                                                       ;
8591 037262  020327  177640        21$:    CMP     R3,#177640              ;
8592 037266  001365                        BNE     19$                     ;
8593 037270  022327  125252        22$:    CMP     (R3)+,#125252           ;CHECK UPARS
8594 037274  001401                        BEQ     23$                     ;
8595 037276  104002                        ERROR   +2                      ;MMU ERROR
8596                                                                       ;
8597 037300  022327  052525        23$:    CMP     (R3)+,#52525            ;
8598 037304  001401                        BEQ     24$                     ;
8599 037306  104002                        ERROR   +2                      ;MMU ERROR
8600                                                                       ;
8601 037310  020327  177700        24$:    CMP     R3,#177700              ;
8602 037314  001365                        BNE     22$                     ;
8603                                  ;
8604                                  ;REVERSE ALTERNATING PATTERN
8605                                  ;
8606 037316  012700  172200                MOV     #172200,R0              ;LOAD SPDRS WITH REVERSE PATTERN
8607 037322  012720  125012        25$:    MOV     #125012,(R0)+           ;
8608 037326  012720  052404                MOV     #52404,(R0)+            ;
8609 037332  020027  172240                CMP     R0,#172240              ;
8610 037336  001371                        BNE     25$                     ;
8611 037340  012720  052525        26$:    MOV     #52525,(R0)+            ;LOAD SPARS WITH REVERSE PATTERN
8612 037344  012720  125252                MOV     #125252,(R0)+           ;
8613 037350  020027  172300                CMP     R0,#172300              ;
8614 037354  001371                        BNE     26$                     ;
8615 037356  012720  125012        27$:    MOV     #125012,(R0)+           ;LOAD KPDRS WITH REVERSE PATTERN
8616 037362  012720  052404                MOV     #52404,(R0)+            ;
8617 037366  020027  172340                CMP     R0,#172340              ;
8618 037372  001371                        BNE     27$                     ;
8619 037374  012720  052525        28$:    MOV     #52525,(R0)+            ;LOAD KPARS WITH REVERSE PATTERN
8620 037400  012720  125252                MOV     #125252,(R0)+           ;
8621 037404  020027  172400                CMP     R0,#172400              ;
8622 037410  001371                        BNE     28$                     ;
8623 037412  012700  177600                MOV     #177600,R0              ;LOAD UPDRS WITH REVERSE PATTERN
```

MEMORY MANAGEMENT TESTS

```
8624 037416  012720  125012          29$:    MOV     #125012,(R0)+           ;
8625 037422  012720  052404                  MOV     #52404,(R0)+            ;
8626 037426  020027  177640                  CMP     R0,#177640              ;
8627 037432  001371                          BNE     29$                     ;
8628 037434  012720  052525          30$:    MOV     #52525,(R0)+            ;LOAD UPARS WITH REVERSE PATTERN
8629 037440  012720  125252                  MOV     #125252,(R0)+           ;
8630 037444  020027  177700                  CMP     R0,#177700              ;
8631 037450  001371                          BNE     30$                     ;
8632                                 ;
8633 037452  012703  172200                  MOV     #172200,R3              ;CHECK SPDRS
8634 037456  022327  125012          31$:    CMP     (R3)+,#125012           ;
8635 037462  001401                          BEQ     32$                     ;
8636 037464  104002                          ERROR   +2              ;MMU ERROR
8637                                                                 ;
8638 037466  022327  052404          32$:    CMP     (R3)+,#52404            ;
8639 037472  001401                          BEQ     33$                     ;
8640 037474  104002                          ERROR   +2              ;MMU ERROR
8641                                                                 ;
8642 037476  020327  172240          33$:    CMP     R3,#172240              ;
8643 037502  001365                          BNE     31$                     ;
8644 037504  022327  052525          34$:    CMP     (R3)+,#52525            ;CHECK SPARS
8645 037510  001401                          BEQ     35$                     ;
8646 037512  104002                          ERROR   +2              ;MMU ERROR
8647                                                                 ;
8648 037514  022327  125252          35$:    CMP     (R3)+,#125252           ;
8649 037520  001401                          BEQ     36$                     ;
8650 037522  104002                          ERROR   +2              ;MMU ERROR
8651                                                                 ;
8652 037524  020327  172300          36$:    CMP     R3,#172300              ;
8653 037530  001365                          BNE     34$                     ;
8654 037532  022327  125012          37$:    CMP     (R3)+,#125012           ;CHECK KPDRS
8655 037536  001401                          BEQ     38$                     ;
8656 037540  104002                          ERROR   +2              ;MMU ERROR
8657                                                                 ;
8658 037542  022327  052404          38$:    CMP     (R3)+,#52404            ;
8659 037546  001401                          BEQ     39$                     ;
8660 037550  104002                          ERROR   +2              ;MMU ERROR
8661                                                                 ;
8662 037552  020327  172340          39$:    CMP     R3,#172340              ;
8663 037556  001365                          BNE     37$                     ;
8664 037560  022327  052525          40$:    CMP     (R3)+,#52525            ;CHECK KPARS
8665 037564  001401                          BEQ     41$                     ;
8666 037566  104002                          ERROR   +2              ;MMU ERROR
8667                                                                 ;
8668 037570  022327  125252          41$:    CMP     (R3)+,#125252           ;
8669 037574  001401                          BEQ     42$                     ;
8670 037576  104002                          ERROR   +2              ;MMU ERROR
8671                                                                 ;
8672 037600  020327  172400          42$:    CMP     R3,#172400              ;
8673 037604  001365                          BNE     40$                     ;
8674 037606  012703  177600                  MOV     #177600,R3              ;CHECK UPDRS
8675 037612  022327  125012          43$:    CMP     (R3)+,#125012           ;
8676 037616  001401                          BEQ     44$                     ;
8677 037620  104002                          ERROR   +2              ;MMU ERROR
8678                                                                 ;
8679 037622  022327  052404          44$:    CMP     (R3)+,#52404            ;
8680 037626  001401                          BEQ     45$                     ;
```

MEMORY MANAGEMENT TESTS

```
8681 037630  104002                          ERROR   +2              ;MMU ERROR
8682                                                                 ;
8683 037632  020327  177640          45$:    CMP     R3,#177640             ;
8684 037636  001365                          BNE     43$                    ;
8685 037640  022327  052525          46$:    CMP     (R3)+,#52525           ;CHECK UPARS
8686 037644  001401                          BEQ     47$                    ;
8687 037646  104002                          ERROR   +2              ;MMU ERROR
8688                                                                 ;
8689 037650  022327  125252          47$:    CMP     (R3)+,#125252          ;
8690 037654  001401                          BEQ     48$                    ;
8691 037656  104002                          ERROR   +2              ;MMU ERROR
8692                                                                 ;
8693 037660  020327  177700          48$:    CMP     R3,#177700             ;
8694 037664  001365                          BNE     46$                    ;
8695
8698 037666                          TSMMU5:
8699                                 ;       TEST MMR0 ABORT BITS
8700 037666  012737  160000  177572          MOV     #160000,@#177572       ;LOAD MMR0<15:13>=111
8701 037674  005067  143130                  CLR     FLAG                   ;CLEAR MMU ABORT FLAG
8702 037700  013700  177572                  MOV     @#SR0,R0               ;SAVE SR0 IN R0
8703 037704  042700  000176                  BIC     #176,R0                ;CLEAR UNDEFINED BITS FROM SR0
8704 037710  020027  160000                  CMP     R0,#160000             ;CHECK MMR0
8705 037714  001401                          BEQ     1$                     ;
8706 037716  104002                          ERROR   +2              ;MMU ERROR
8707                                                                 ;
8708 037720  005037  177572          1$:     CLR     @#177572               ;LOAD MMR0=0
8709 037724  013700  177572                  MOV     @#SR0,R0               ;SAVE SR0 IN R0
8710 037730  042700  000176                  BIC     #176,R0                ;CLEAR UNDEFINED BITS FROM SR0
8711 037734  020027  000000                  CMP     R0,#0                  ;CHECK MMR0
8712 037740  001401                          BEQ     2$                     ;
8713 037742  104002                          ERROR   +2              ;MMU ERROR
8714                                                                 ;
8715 037744  012737  120000  177572  2$:     MOV     #120000,@#177572       ;LOAD MMR0<15:13>=101
8716 037752  013700  177572                  MOV     @#SR0,R0               ;SAVE SR0 IN R0
8717 037756  042700  000176                  BIC     #176,R0                ;CLEAR UNDEFINED BITS FROM SR0.
8718 037762  020027  120000                  CMP     R0,#120000             ;CHECK MMR0
8719 037766  001401                          BEQ     3$                     ;
8720 037770  104002                          ERROR   +2              ;MMU ERROR
8721                                                                 ;
8722 037772  012737  040000  177572  3$:     MOV     #40000,@#177572        ;LOAD MMR0<15:13>=010
8723 040000  013700  177572                  MOV     @#SR0,R0               ;SAVE SR0 IN R0
8724 040004  042700  000176                  BIC     #176,R0                ;CLEAR UNDEFINED BITS FROM SR0.
8725 040010  020027  040000                  CMP     R0,#40000              ;CHECK MMR0
8726 040014  001401                          BEQ     4$                     ;
8727 040016  104002                          ERROR   +2              ;MMU ERROR
8728 040020                          4$:
8731 040020                          TSMMU6:
8732                                 ;       TEST MMR3 BITS 5-0
8733 040020  005037  177572                  CLR     @#177572               ;MMU OFF
8734 040024  005067  143000                  CLR     FLAG                   ;CLEAR MMU ABORT FLAG
8735 040030  012737  000077  172516          MOV     #77,@#172516           ;LOAD MMR3<5:0>=77
8736 040036  023727  172516  000077          CMP     @#172516,#77           ;CHECK MMR3
8737 040044  001401                          BEQ     1$                     ;
8738 040046  104002                          ERROR   +2              ;MMU ERROR
8739 040050  005037  172516          1$:     CLR     @#172516               ;LOAD MMR3<5:0>=0
8740 040054  023727  172516  000000          CMP     @#172516,#0            ;CHECK MMR3
8741 040062  001401                          BEQ     2$                     ;
```

MEMORY MANAGEMENT TESTS

```
 8742 040064  104002                          ERROR   ;2                ;MMU ERROR
 8743 040066  012737  000052  172516  2$:     MOV     #52,@#172516              ;LOAD MMR3<5:0>=52
 8744 040074  023727  172516  000052          CMP     @#172516,#52              ;CHECK MMR3
 8745 040102  001401                          BEQ     3$                        ;
 8746 040104  104002                          ERROR   ;2                ;MMU ERROR
 8747 040106  012737  000025  172516  3$:     MOV     #25,@#172516              ;LOAD MMR3<5:0>=25
 8748 040114  023727  172516  000025          CMP     @#172516,#25              ;CHECK MMR3
 8749 040122  001401                          BEQ     4$                        ;
 8750 040124  104002                          ERROR   ;2                ;MMU ERROR
 8751 040126                          4$:
 8754 040126                          TSMM6A:
 8755                                 ;       TEST MFPI (MOVE FROM PREVIOUS INST SPACE)
 8756 040126  005037  177572                  CLR     @#177572                  ;MMU OFF
 8757 040132  005037  003030                  CLR     @#FLAG                    ;CLEAR MMU ABORT FLAG
 8758 040136  012737  140000  177776          MOV     #140000,@#177776          ;POINT TO USER SPACE
 8759 040144  012706  001000                  MOV     #STBOT,SP                 ;INIT THE USER STACK POINTER
 8760 040150  010637  003040                  MOV     R6,@#SAVUSE               ;SAVE USER SP
 8761 040154  012737  040000  177776          MOV     #40000,@#177776           ;POINT TO SUPERVISOR SPACE
 8762 040162  012706  001000                  MOV     #STBOT,SP                 ;INIT THE SUPERVISOR STACK POINTER
 8763 040166  010637  003036                  MOV     R6,@#SAVSUP               ;SAVE SUPERVISOR SP
 8764 040172  012737  030000  177776          MOV     #30000,@#177776           ;SETUP PSW
 8765 040200  004767  077346                  JSR     PC,MMU                    ;INIT MMU
 8766 040204  012737  000027  172516          MOV     #27,@#172516              ;SETUP MMR3
 8767 040212  013746  000244                  MOV     @#244,-(SP)               ;SAVE DATA AT TEST LOCATION
 8768 040216  012746  177777                  MOV     #177777,-(SP)             ;PUT KNOWN DATA ON TOP OF STACK
 8769 040222  012737  135072  000244          MOV     #135072,@#244             ;SETUP DATA AT TEST LOCATION
 8770 040230  012767  077400  137362          MOV     #77400,UDPDR0             ;SETUP UDPDR0 TO ABORT
 8771 040236  012703  000244                  MOV     #244,R3                   ;SETUP POINTER TO TEST LOCATION
 8772 040242  005237  177572                  INC     @#177572                  ;TURN MMU ON
 8773 040246  006523                          MFPI    (R3)+                     ; TEST INSTRUCTION
 8774 040250  022737  030010  177776          CMP     #30010,@#177776           ;IS PSW CORRECT
 8775 040256  001401                          BEQ     1$                        ;YES GO ON
 8776 040260  104002                          ERROR   ;2                ;MMU ERROR
 8777                                                                   ;NO GO TO ERROR
 8778 040262  005037  177572          1$:     CLR     @#177572                  ;TURN MMU OFF
 8779 040266  012737  140000  177776          MOV     #140000,@#177776          ;POINT TO USER SPACE
 8780 040274  020637  003040                  CMP     R6,@#SAVUSE               ;IS USER SP CORRECT
 8781 040300  001401                          BEQ     100$                      ;YES GO ON
 8782 040302  104002                          ERROR   ;2                ;MMU ERROR
 8783                                                                   ;NO GO TO ERROR
 8784 040304  012737  040000  177776  100$:   MOV     #40000,@#177776           ;POINT TO SUPERVISOR SPACE
 8785 040312  020637  003036                  CMP     R6,@#SAVSUP               ;IS SUPERVISOR SP CORRECT
 8786 040316  001401                          BEQ     200$                      ;YES GO ON
 8787 040320  104002                          ERROR   ;2                ;MMU ERROR
 8788                                                                   ;NO GO TO ERROR
 8789 040322  023727  000244  135072  200$:   CMP     @#244,#135072             ;IS TEST DATA OK
 8790 040330  001401                          BEQ     2$                        ;YES GO ON
 8791 040332  104002                          ERROR   ;2                ;MMU ERROR
 8792                                                                   ;NO GO TO ERROR
 8793 040334  020327  000246          2$:     CMP     R3,#246                   ;IS R3 CORRECT
 8794 040340  001401                          BEQ     3$                        ;YES GO ON
 8795 040342  104002                          ERROR   ;2                ;MMU ERROR
 8796                                                                   ;NO GO TO ERROR
 8797 040344  005037  177776          3$:     CLR     @#177776                  ;SET PSW TO KERNEL MODE
 8798 040350  022627  135072                  CMP     (SP)+,#135072             ;IS KERNEL STACK CORRECT
 8799 040354  001401                          BEQ     4$                        ;YES GO ON
 8800 040356  104002                          ERROR   ;2                ;MMU ERROR
```

MEMORY MANAGEMENT TESTS

```
8801
8802 040360  021627  177777          4$      CMP     (SP),#177777            ;NO GO TO ERROR
8803 040364  001401                          BEQ     5$                      ;IS STACK CORRECT
8804 040366  104002                          ERROR   .+2                     ;YES GO ON
8805                                                                         ;MMU ERROR
8806 040370  012737  030017  177776  5$:     MOV     #30017,@#177776         ;NO GO TO ERROR
8807 040376  012737  173621  000244          MOV     #173621,@#244           ;SETUP PSW
8808 040404  012701  000244                  MOV     #244,R1                 ;SETUP TEST LOCATION
8809 040410  005237  177572                  INC     @#177572                ;SETUP R1
8810 040414  006511                          MFPI    (R1)                    ;TURN MMU ON
8811 040416  022737  030011  177776          CMP     #30011,@#177776         ;TEST INSTRUCTION
8812 040424  001401                          BEQ     300$                    ;IS PSW CORRECT
8813 040426  104002                          ERROR   .+2                     ;YES GO ON
8814                                                                         ;MMU ERROR
8815 040430  005037  177572          300$:   CLR     @#177572                ;NO GO TO ERROR
8816 040434  023727  000244  173621          CMP     @#244,#173621           ;TURN MMU OFF
8817 040442  001401                          BEQ     301$                    ;IS TEST LOCATION CORRECT
8818 040444  104002                          ERROR   .+2                     ;YES GO ON
8819                                                                         ;MMU ERROR
8820 040446  020127  000244          301$:   CMP     R1,#244                 ;NO GO TO ERROR
8821 040452  001401                          BEQ     302$                    ;IS R1 CORRECT
8822 040454  104002                          ERROR   .+2                     ;YES GO ON
8823                                                                         ;MMU ERROR
8824 040456  005037  177776          302$:   CLR     @#177776                ;NO GO TO ERROR
8825 040462  022627  173621                  CMP     (SP)+,#173621           ;SET PSW TO KERNEL MODE
8826 040466  001401                          BEQ     303$                    ;IS STACK CORRECT
8827 040470  104002                          ERROR   .+2                     ;YES GO ON
8828                                                                         ;MMU ERROR
8829 040472  021627  177777          303$:   CMP     (SP),#177777            ;NO GO TO ERROR
8830 040476  001401                          BEQ     304$                    ;IS STACK CORRECT
8831 040500  104002                          ERROR   .+2                     ;YES GO ON
8832                                                                         ;MMU ERROR
8833 040502  005003                  304$:   CLR     R3                      ;NO GO TO ERROR
8834 040504  005237  177572                  INC     @#177572                ;SETUP SOURCE FOR NEXT TEST
8835 040510  006503                          MFPI    R3                      ;TURN MMU ON
8836 040512  022737  000004  177776          CMP     #4,@#177776             ; TEST INSTRUCTION
8837 040520  001401                          BEQ     6$                      ;IS PSW CORRECT
8838 040522  104002                          ERROR   .+2                     ;YES GO ON
8839                                                                         ;MMU ERROR
8840 040524  005037  177572          6$:     CLR     @#177572                ;NO GO TO ERROR
8841 040530  020327  000000                  CMP     R3,#0                   ;TURN MMU OFF
8842 040534  001401                          BEQ     7$                      ;IS R3 CORRECT
8843 040536  104002                          ERROR   .+2                     ;YES GO ON
8844                                                                         ;MMU ERROR
8845 040540  022627  000000          7$:     CMP     (SP)+,#0                ;NO GO TO ERROR
8846 040544  001401                          BEQ     8$                      ;IS STACK CORRECT
8847 040546  104002                          ERROR   .+2                     ;YES GO ON
8848                                                                         ;MMU ERROR
8849 040550  022627  177777          8$:     CMP     (SP)+,#177777           ;NO GO TO ERROR
8850 040554  001401                          BEQ     9$                      ;IS STACK CORRECT
8851 040556  104002                          ERROR   .+2                     ;YES GO ON
8852                                                                         ;MMU ERROR
8853 040560  012637  000244          9$:     MOV     (SP)+,@#244             ;NO GO TO ERROR
8854                                                                         ;RESTORE TEST LOCATION
8855
8858 040564                          ;TSMM16B:
8859                                 ;       TEST MFPD (MOVE FROM PREVIOUS DATA SPACE)
```

E13

MEMORY MANAGEMENT TESTS

```
8860 040564  005037  177572             CLR     @#177572                ;MMU OFF
8861 040570  005037  003030             CLR     @#FLAG                  ;CLEAR MMU ABORT FLAG
8862 040574  012737  140000  177776     MOV     #140000,@#177776        ;POINT TO USER SPACE
8863 040602  010637  003040             MOV     R6,@#SAVUSE             ;SAVE USER SP
8864 040606  012737  040000  177776     MOV     #40000,@#177776         ;POINT TO SUPERVISOR SPACE
8865 040614  010637  003036             MOV     R6,@#SAVSUP             ;SAVE SUPERVISOR SP
8866 040620  012737  030000  177776     MOV     #30000,@#177776         ;SETUP PSW
8867 040626  004767  076720             JSR     PC,MMU                  ;INIT MMU
8868 040632  012737  000027  172516     MOV     #27,@#172516            ;SETUP MMR3
8869 040640  013746  000244             MOV     @#244,-(SP)             ;SAVE DATA AT TEST LOCATION
8870 040644  012746  177777             MOV     #177777,-(SP)           ;PUT KNOWN DATA ON TOP OF STACK
8871 040650  012737  157002  000244     MOV     #157002,@#244           ;SETUP DATA AT TEST LOCATION
8872 040656  012767  077400  136714     MOV     #77400,UIPDR0           ;SETUP UIPDR0 TO ABORT
8873 040664  012703  000244             MOV     #244,R3                 ;SETUP POINTER TO TEST LOCATION
8874 040670  005237  177572             INC     @#177572                ;TURN MMU ON
8875 040674  106523                     MFPD    (R3)+                   ; TEST INSTRUCTION
8876 040676  022737  030010  177776     CMP     #30010,@#177776         ;IS PSW CORRECT
8877 040704  001401                     BEQ     1$                      ;YES GO ON
8878 040706  104002                     ERROR   .+2                 ;MMU ERROR
8879                                                                 ;NO GO TO ERROR
8880 040710  005037  177572       1$:   CLR     @#177572                ;TURN MMU OFF
8881 040714  012737  140000  177776     MOV     #140000,@#177776        ;POINT TO USER SPACE
8882 040722  020637  003040             CMP     R6,@#SAVUSE             ;IS USER SP CORRECT
8883 040726  001401                     BEQ     100$                    ;YES GO ON
8884 040730  104002                     ERROR   .+2                 ;MMU ERROR
8885                                                                 ;NO GO TO ERROR
8886 040732  012737  040000  177776 100$:   MOV     #40000,@#177776         ;POINT TO SUPERVISOR SPACE
8887 040740  020637  003036             CMP     R6,@#SAVSUP             ;IS SUPERVISOR SP CORRECT
8888 040744  001401                     BEQ     200$                    ;YES GO ON
8889 040746  104002                     ERROR   .+2                 ;MMU ERROR
8890                                                                 ;NO GO TO ERROR
8891 040750  023727  000244  157002 200$:   CMP     @#244,#157002           ;IS TEST DATA OK
8892 040756  001401                     BEQ     2$                      ;YES GO ON
8893 040760  104002                     ERROR   .+2                 ;MMU ERROR
8894                                                                 ;NO GO TO ERROR
8895 040762  020327  000246         2$:   CMP     R3,#246                 ;IS R3 CORRECT
8896 040766  001401                     BEQ     3$                      ;YES GO ON
8897 040770  104002                     ERROR   .+2                 ;MMU ERROR
8898                                                                 ;NO GO TO ERROR
8899 040772  005037  177776         3$:   CLR     @#177776                ;SET PSW TO KERNEL MODE
8900 040776  022627  157002             CMP     (SP)+,#157002           ;IS KERNEL STACK CORRECT
8901 041002  001401                     BEQ     4$                      ;YES GO ON
8902 041004  104002                     ERROR   .+2                 ;MMU ERROR
8903                                                                 ;NO GO TO ERROR
8904 041006  021627  177777         4$:   CMP     (SP),#177777            ;IS STACK CORRECT
8905 041012  001401                     BEQ     5$                      ;YES GO ON
8906 041014  104002                     ERROR   .+2                 ;MMU ERROR
8907                                                                 ;NO GO TO ERROR
8908 041016  012737  030017  177776 5$:   MOV     #30017,@#177776         ;SETUP PSW
8909 041024  012737  103456  000244     MOV     #103456,@#244           ;SETUP TEST LOCATION
8910 041032  012701  000244             MOV     #244,R1                 ;SETUP R1
8911 041036  005237  177572             INC     @#177572                ;TURN MMU ON
8912 041042  106511                     MFPD    (R1)                    ;TEST INSTRUCTION
8913 041044  022737  030011  177776     CMP     #30011,@#177776         ;IS PSW CORRECT
8914 041052  001401                     BEQ     300$                    ;YES GO ON
8915 041054  104002                     ERROR   .+2                 ;MMU ERROR
8916                                                                 ;NO GO TO ERROR
```

F13

MEMORY MANAGEMENT TESTS

```
8917 041056  005037  177572          300$:   CLR     @#177572                ;TURN MMU OFF
8918 041062  023727  000244  103456          CMP     @#244,#103456           ;IS TEST LOCATION CORRECT
8919 041070  001401                          BEQ     301$                    ;YES GO ON
8920 041072  104002                          ERROR   .+2                     ;MMU ERROR
8921                                                                         ;NO GO TO ERROR
8922 041074  020127  000244          301$:   CMP     R1,#244                 ;IS R1 CORRECT
8923 041100  001401                          BEQ     302$                    ;YES GO ON
8924 041102  104002                          ERROR   .+2                     ;MMU ERROR
8925                                                                         ;NO GO TO ERROR
8926 041104  005037  177776          302$:   CLR     @#177776                ;SET PSW TO KERNEL MODE
8927 041110  022627  103456                  CMP     (SP)+,#103456           ;IS STACK CORRECT
8928 041114  001401                          BEQ     303$                    ;YES GO ON
8929 041116  104002                          ERROR   .+2                     ;MMU ERROR
8930                                                                         ;NO GO TO ERROR
8931 041120  021627  177777          303$:   CMP     (SP),#177777            ;IS STACK CORRECT
8932 041124  001401                          BEQ     304$                    ;YES GO ON
8933 041126  104002                          ERROR   .+2                     ;MMU ERROR
8934                                                                         ;NO GO TO ERROR
8935 041130  012737  030017  177776  304$:   MOV     #30017,@#177776         ;SETUP PSW
8936 041136  012737  113672  000244          MOV     #113672,@#244           ;SETUP TEST LOCATION
8937 041144  012701  000246                  MOV     #246,R1                 ;SETUP R1
8938 041150  005237  177572                  INC     @#177572                ;TURN MMU ON
8939 041154  106541                          MFPD    -(R1)                   ;TEST INSTRUCTION
8940 041156  022737  030011  177776          CMP     #30011,@#177776         ;IS PSW CORRECT
8941 041164  001401                          BEQ     400$                    ;YES GO ON
8942 041166  104002                          ERROR   .+2                     ;MMU ERROR
8943                                                                         ;NO GO TO ERROR
8944 041170  005037  177572          400$:   CLR     @#177572                ;TURN MMU OFF
8945 041174  023727  000244  113672          CMP     @#244,#113672           ;IS TEST LOCATION CORRECT
8946 041202  001401                          BEQ     401$                    ;YES GO ON
8947 041204  104002                          ERROR   .+2                     ;MMU ERROR
8948                                                                         ;NO GO TO ERROR
8949 041206  020127  000244          401$:   CMP     R1,#244                 ;IS R1 CORRECT
8950 041212  001401                          BEQ     402$                    ;YES GO ON
8951 041214  104002                          ERROR   .+2                     ;MMU ERROR
8952                                                                         ;NO GO TO ERROR
8953 041216  005037  177776          402$:   CLR     @#177776                ;SET PSW TO KERNEL MODE
8954 041222  022627  113672                  CMP     (SP)+,#113672           ;IS STACK CORRECT
8955 041226  001401                          BEQ     403$                    ;YES GO ON
8956 041230  104002                          ERROR   .+2                     ;MMU ERROR
8957                                                                         ;NO GO TO ERROR
8958 041232  021627  177777          403$:   CMP     (SP),#177777            ;IS STACK CORRECT
8959 041236  001401                          BEQ     404$                    ;YES GO ON
8960 041240  104002                          ERROR   .+2                     ;MMU ERROR
8961                                                                         ;NO GO TO ERROR
8962 041242  005003                  404$:   CLR     R3                      ;SETUP SOURCE FOR NEXT TEST
8963 041244  005237  177572                  INC     @#177572                ;TURN MMU ON
8964 041250  106503                          MFPD    R3                      ; TEST INSTRUCTION
8965 041252  022737  000004  177776          CMP     #4,@#177776             ;IS PSW CORRECT
8966 041260  001401                          BEQ     6$                      ;YES GO ON
8967 041262  104002                          ERROR   .+2                     ;MMU ERROR
8968                                                                         ;NO GO TO ERROR
8969 041264  005037  177572          6$:     CLR     @#177572                ;TURN MMU OFF
8970 041270  020327  000000                  CMP     R3,#0                   ;IS R3 CORRECT
8971 041274  001401                          BEQ     7$                      ;YES GO ON
8972 041276  104002                          ERROR   .+2                     ;MMU ERROR
8973                                                                         ;NO GO TO ERROR
```

MEMORY MANAGEMENT TESTS

```
8974 041300  022627  000000          7$:     CMP     (SP)+,#0                ;IS STACK CORRECT
8975 041304  001401                          BEQ     8$                      ;YES GO ON
8976 041306  104002                          ERROR   .+2             ;MMU ERROR
8977                                                                 ;NO GO TO ERROR
8978 041310  022627  177777          8$:     CMP     (SP)+,#177777           ;IS STACK CORRECT
8979 041314  001401                          BEQ     9$                      ;YES GO ON
8980 041316  104002                          ERROR   .+2             ;MMU ERROR
8981                                                                 ;NO GO TO ERROR
8982 041320  012637  000244          9$:     MOV     (SP)+,@#244             ;RESTORE TEST LOCATION
8983
8984                                         ;
8987 041324                          TSMM6C:
8988                                 ;       TEST MTPI (MOVE TO PREVIOUS INSTRUCTION SPACE)
8989 041324  005037  177572                  CLR     @#177572                ;MMU OFF
8990 041330  005037  003030                  CLR     @#FLAG                  ;CLEAR MMU ABORT FLAG
8991 041334  012737  140000  177776          MOV     #140000,@#177776        ;POINT TO USER SPACE
8992 041342  010637  003040                  MOV     R6,@#SAVUSE             ;SAVE USER SP
8993 041346  012737  040000  177776          MOV     #40000,@#177776         ;POINT TO SUPERVISOR SPACE
8994 041354  010637  003036                  MOV     R6,@#SAVSUP             ;SAVE SUPERVISOR SP
8995 041360  012737  030000  177776          MOV     #30000,@#177776         ;SETUP PSW
8996 041366  004767  076160                  JSR     PC,MMU                  ;INIT MMU
8997 041372  012737  000027  172516          MOV     #27,@#172516            ;SETUP MMR3
8998 041400  013746  000244                  MOV     @#244,-(SP)             ;SAVE DATA AT TEST LOCATION
8999 041404  012746  177777                  MOV     #177777,-(SP)           ;PUT KNOWN DATA ON STACK
9000 041410  012746  120413                  MOV     #120413,-(SP)           ;PUT TEST DATA ON STACK
9001 041414  012737  177777  000244          MOV     #177777,@#244           ;PUT KNOWN DATA AT TEST LOCATION
9002 041422  012767  077400  136170          MOV     #77400,UDPDR0           ;SETUP UDPDR0 TO ABORT
9003 041430  012703  000244                  MOV     #244,R3                 ;SETUP POINTER TO TEST LOCATION
9004 041434  005237  177572                  INC     @#177572                ;TURN MMU ON
9005 041440  006623                          MTPI    (R3)+                   ; TEST INSTRUCTION
9006 041442  022737  030010  177776          CMP     #30010,@#177776         ;IS PSW CORRECT
9007 041450  001401                          BEQ     1$                      ;YES GO ON
9008 041452  104002                          ERROR   .+2             ;MMU ERROR
9009                                                                 ;NO GO TO ERROR
9010 041454  005037  177572          1$:     CLR     @#177572                ;TURN MMU OFF
9011 041460  012737  140000  177776          MOV     #140000,@#177776        ;POINT TO USER SPACE
9012 041466  020637  003040                  CMP     R6,@#SAVUSE             ;IS USER SP CORRECT
9013 041472  001401                          BEQ     100$                    ;YES GO ON
9014 041474  104002                          ERROR   .+2             ;MMU ERROR
9015                                                                 ;NO GO TO ERROR
9016 041476  012737  040000  177776  100$:   MOV     #40000,@#177776         ;POINT TO SUPERVISOR SPACE
9017 041504  020637  003036                  CMP     R6,@#SAVSUP             ;IS SUPERVISOR SP CORRECT
9018 041510  001401                          BEQ     200$                    ;YES GO ON
9019 041512  104002                          ERROR   .+2             ;MMU ERROR
9020                                                                 ;NO GO TO ERROR
9021 041514  023727  000244  120413  200$:   CMP     @#244,#120413           ;IS TEST LOCATION CORRECT
9022 041522  001401                          BEQ     2$                      ;YES GO ON
9023 041524  104002                          ERROR   .+2             ;MMU ERROR
9024                                                                 ;NO GO TO ERROR
9025 041526  020327  000246          2$:     CMP     R3,#246                 ;IS R3 CORRECT
9026 041532  001401                          BEQ     3$                      ;YES GO ON
9027 041534  104002                          ERROR   .+2             ;MMU ERROR
9028                                                                 ;NO GO TO ERROR
9029 041536  005037  177776          3$:     CLR     @#177776                ;SET PSW TO KERNEL MODE
9030 041542  021627  177777                  CMP     (SP),#177777            ;IS KERNEL STACK CORRECT
9031 041546  001401                          BEQ     4$                      ;YES GO ON
9032 041550  104002                          ERROR   .+2             ;MMU ERROR
```

MEMORY MANAGEMENT TESTS

```
                                                                                ;NO GO TO ERROR
9033
9034 041552  012737  030017  177776  4$:     MOV     #30017,@#177776         ;SETUP PSW
9035 041560  012746  145121                  MOV     #145121,-(SP)           ;SETUP TEST DATA
9036 041564  012701  000244                  MOV     #244,R1                 ;SETUP R1
9037 041570  005237  177572                  INC     @#177572                ;TURN MMU ON
9038 041574  006611                          MTPI    (R1)                    ;TEST INSTRUCTION
9039 041576  022737  030011  177776          CMP     #30011,@#177776         ;IS PSW CORRECT
9040 041604  001401                          BEQ     300$                    ;YES GO ON
9041 041606  104002                          ERROR   +2                      ;MMU ERROR
                                                                                ;NO GO TO ERROR
9042
9043 041610  005037  177572          300$:   CLR     @#177572                ;TURN MMU OFF
9044 041614  023727  000244  145121          CMP     @#244,#145121           ;IS TEST LOCATION CORRECT
9045 041622  001401                          BEQ     301$                    ;YES GO ON
9046 041624  104002                          ERROR   +2                      ;MMU ERROR
                                                                                ;NO GO TO ERROR
9047
9048 041626  020127  000244          301$:   CMP     R1,#244                 ;IS R1 CORRECT
9049 041632  001401                          BEQ     302$                    ;YES GO ON
9050 041634  104002                          ERROR   +2                      ;MMU ERROR
                                                                                ;NO GO TO ERROR
9051
9052 041636  005037  177776          302$:   CLR     @#177776                ;SET PSW TO KERNEL MODE
9053 041642  021627  177777                  CMP     (SP),#177777            ;IS STACK CORRECT
9054 041646  001401                          BEQ     304$                    ;YES GO ON
9055 041650  104002                          ERROR   +2                      ;MMU ERROR
                                                                                ;NO GO TO ERROR
9056
9057 041652  012737  030017  177776  304$:   MOV     #30017,@#177776         ;SETUP PSW
9058 041660  012746  122347                  MOV     #122347,-(SP)           ;SETUP TEST DATA
9059 041664  012701  000246                  MOV     #246,R1                 ;SETUP R1
9060 041670  005237  177572                  INC     @#177572                ;TURN MMU ON
9061 041674  006641                          MTPI    -(R1)                   ;TEST INSTRUCTION
9062 041676  022737  030011  177776          CMP     #30011,@#177776         ;IS PSW CORRECT
9063 041704  001401                          BEQ     400$                    ;YES GO ON
9064 041706  104002                          ERROR   +2                      ;MMU ERROR
                                                                                ;NO GO TO ERROR
9065
9066 041710  005037  177572          400$:   CLR     @#177572                ;TURN MMU OFF
9067 041714  023727  000244  122347          CMP     @#244,#122347           ;IS TEST LOCATION CORRECT
9068 041722  001401                          BEQ     401$                    ;YES GO ON
9069 041724  104002                          ERROR   +2                      ;MMU ERROR
                                                                                ;NO GO TO ERROR
9070
9071 041726  020127  000244          401$:   CMP     R1,#244                 ;IS R1 CORRECT
9072 041732  001401                          BEQ     402$                    ;YES GO ON
9073 041734  104002                          ERROR   +2                      ;MMU ERROR
                                                                                ;NO GO TO ERROR
9074
9075 041736  005037  177776          402$:   CLR     @#177776                ;SET PSW TO KERNEL MODE
9076 041742  021627  177777                  CMP     (SP),#177777            ;IS STACK CORRECT
9077 041746  001401                          BEQ     404$                    ;YES GO ON
9078 041750  104002                          ERROR   +2                      ;MMU ERROR
                                                                                ;NO GO TO ERROR
9079
9080 041752  005046                  404$:   CLR     -(SP)                   ;SETUP STACK FOR NEXT TEST
9081 041754  005237  177572                  INC     @#177572                ;TURN MMU ON
9082 041760  006603                          MTPI    R3                      ; TEST INSTRUCTION
9083 041762  022737  000004  177776          CMP     #4,@#177776             ;IS PSW CORRECT
9084 041770  001401                          BEQ     5$                      ;YES GO ON
9085 041772  104002                          ERROR   +2                      ;MMU ERROR
                                                                                ;NO GO TO ERROR
9086
9087 041774  005037  177572          5$:     CLR     @#177572                ;TURN MMU OFF
9088 042000  020327  000000                  CMP     R3,#0                   ;IS R3 CORRECT
9089 042004  001401                          BEQ     6$                      ;YES GO ON
```

```
9090 042006  104002                          ERROR   +2              ;MMU ERROR
9091                                                                 ;NO GO TO ERROR
9092 042010  022627  177777          6$:     CMP     (SP)+,#177777           ;IS STACK CORRECT
9093 042014  001401                          BEQ     7$              ;YES GO ON
9094 042016  104002                          ERROR   +2              ;MMU ERROR
9095                                                                 ;NO GO TO ERROR
9096 042020  012637  000244          7$:     MOV     (SP)+,@#244             ;RESTORE TEST LOCATION
9097
9098                                         ;
9101 042024                                  TSMM6D:
9102                                         ;       TEST MTPD (MOVE TO PREVIOUS DATA SPACE)
9103 042024  005037  177572                  CLR     @#177572                ;MMU OFF
9104 042030  005037  003030                  CLR     @#FLAG                  ;CLEAR MMU ABORT FLAG
9105 042034  012737  140000  177776          MOV     #140000,@#177776        ;POINT TO USER SPACE
9106 042042  010637  003040                  MOV     R6,@#SAVUSE             ;SAVE USER SP
9107 042046  012737  040000  177776          MOV     #40000,@#177776         ;POINT TO SUPERVISOR SPACE
9108 042054  010637  003036                  MOV     R6,@#SAVSUP             ;SAVE SUPERVISOR SP
9109 042060  012737  030000  177776          MOV     #30000,@#177776         ;SETUP PSW
9110 042066  004767  075460                  JSR     PC,MMU                  ;INIT MMU
9111 042072  012737  000027  172516          MOV     #27,@#172516            ;SETUP MMR3
9112 042100  013746  000244                  MOV     @#244,-(SP)             ;SAVE DATA AT TEST LOCATION
9113 042104  012746  177777                  MOV     #177777,-(SP)           ;PUT KNOWN DATA ON STACK
9114 042110  012746  100004                  MOV     #100004,-(SP)           ;PUT TEST DATA ON STACK
9115 042114  012737  177777  000244          MOV     #177777,@#244           ;PUT KNOWN DATA AT TEST LOCATION
9116 042122  012767  077400  135450          MOV     #77400,UIPDR0           ;SETUP UIPDR0 TO ABORT
9117 042130  012703  000244                  MOV     #244,R3                 ;SETUP POINTER TO TEST LOCATION
9118 042134  005237  177572                  INC     @#177572                ;TURN MMU ON
9119 042140  106623                          MTPD    (R3)+                   ; TEST INSTRUCTION
9120 042142  022737  030010  177776          CMP     #30010,@#177776         ;IS PSW CORRECT
9121 042150  001401                          BEQ     1$              ;YES GO ON
9122 042152  104002                          ERROR   +2              ;MMU ERROR
9123                                                                 ;NO GO TO ERROR
9124 042154  005037  177572          1$:     CLR     @#177572                ;TURN MMU OFF
9125 042160  012737  140000  177776          MOV     #140000,@#177776        ;POINT TO USER SPACE
9126 042166  020637  003040                  CMP     R6,@#SAVUSE             ;IS USER SP CORRECT
9127 042172  001401                          BEQ     100$            ;YES GO ON
9128 042174  104002                          ERROR   +2              ;MMU ERROR
9129                                                                 ;NO GO TO ERROR
9130 042176  012737  040000  177776  100$:   MOV     #40000,@#177776         ;POINT TO SUPERVISOR SPACE
9131 042204  020637  003036                  CMP     R6,@#SAVSUP             ;IS SUPERVISOR SP CORRECT
9132 042210  001401                          BEQ     200$            ;YES GO ON
9133 042212  104002                          ERROR   +2              ;MMU ERROR
9134                                                                 ;NO GO TO ERROR
9135 042214  023727  000244  100004  200$:   CMP     @#244,#100004           ;IS TEST LOCATION CORRECT
9136 042222  001401                          BEQ     2$              ;YES GO ON
9137 042224  104002                          ERROR   +2              ;MMU ERROR
9138                                                                 ;NO GO TO ERROR
9139 042226  020327  000246          2$:     CMP     R3,#246                 ;IS R3 CORRECT
9140 042232  001401                          BEQ     3$              ;YES GO ON
9141 042234  104002                          ERROR   +2              ;MMU ERROR
9142                                                                 ;NO GO TO ERROR
9143 042236  005037  177776          3$:     CLR     @#177776                ;SET PSW TO KERNEL MODE
9144 042242  021627  177777                  CMP     (SP),#177777            ;IS KERNEL STACK CORRECT
9145 042246  001401                          BEQ     4$              ;YES GO ON
9146 042250  104002                          ERROR   +2              ;MMU ERROR
9147                                                                 ;NO GO TO ERROR
9148 042252  012737  030017  177776  4$:     MOV     #30017,@#177776         ;SETUP PSW
```

J13

MEMORY MANAGEMENT TESTS

```
 9149 042260  012746  100737                    MOV     #100737,-(SP)          ;SETUP TEST DATA
 9150 042264  012701  000244                    MOV     #244,R1                ;SETUP R1
 9151 042270  005237  177572                    INC     @#177572               ;TURN MMU ON
 9152 042274  106611                            MTPD    (R1)                   ;TEST INSTRUCTION
 9153 042276  022737  030011  177776            CMP     #30011,@#177776        ;IS PSW CORRECT
 9154 042304  001401                            BEQ     300$                   ;YES GO ON
 9155 042306  104002                            ERROR   .+2                 ;MMU ERROR
                                                                            ;NO GO TO ERROR
 9156
 9157 042310  005037  177572          300$:     CLR     @#177572               ;TURN MMU OFF
 9158 042314  023727  000244  100737            CMP     @#244,#100737          ;IS TEST LOCATION CORRECT
 9159 042322  001401                            BEQ     301$                   ;YES GO ON
 9160 042324  104002                            ERROR   .+2                 ;MMU ERROR
                                                                            ;NO GO TO ERROR
 9161
 9162 042326  020127  000244          301$:     CMP     R1,#244                ;IS R1 CORRECT
 9163 042332  001401                            BEQ     302$                   ;YES GO ON
 9164 042334  104002                            ERROR   .+2                 ;MMU ERROR
                                                                            ;NO GO TO ERROR
 9165
 9166 042336  005037  177776          302$:     CLR     @#177776               ;SET PSW TO KERNEL MODE
 9167 042342  021627  177777                    CMP     (SP),#177777           ;IS STACK CORRECT
 9168 042346  001401                            BEQ     304$                   ;YES GO ON
 9169 042350  104002                            ERROR   .+2                 ;MMU ERROR
                                                                            ;NO GO TO ERROR
 9170
 9171 042352  012737  030017  177776  304$:     MOV     #30017,@#177776        ;SETUP PSW
 9172 042360  012746  156711                    MOV     #156711,-(SP)          ;SETUP TEST DATA
 9173 042364  012701  000246                    MOV     #246,R1                ;SETUP R1
 9174 042370  005237  177572                    INC     @#177572               ;TURN MMU ON
 9175 042374  106641                            MTPD    -(R1)                  ;TEST INSTRUCTION
 9176 042376  022737  030011  177776            CMP     #30011,@#177776        ;IS PSW CORRECT
 9177 042404  001401                            BEQ     400$                   ;YES GO ON
 9178 042406  104002                            ERROR   .+2                 ;MMU ERROR
                                                                            ;NO GO TO ERROR
 9179
 9180 042410  005037  177572          400$:     CLR     @#177572               ;TURN MMU OFF
 9181 042414  023727  000244  156711            CMP     @#244,#156711          ;IS TEST LOCATION CORRECT
 9182 042422  001401                            BEQ     401$                   ;YES GO ON
 9183 042424  104002                            ERROR   .+2                 ;MMU ERROR
                                                                            ;NO GO TO ERROR
 9184
 9185 042426  020127  000244          401$:     CMP     R1,#244                ;IS R1 CORRECT
 9186 042432  001401                            BEQ     402$                   ;YES GO ON
 9187 042434  104002                            ERROR   .+2                 ;MMU ERROR
                                                                            ;NO GO TO ERROR
 9188
 9189 042436  005037  177776          402$:     CLR     @#177776               ;SET PSW TO KERNEL MODE
 9190 042442  021627  177777                    CMP     (SP),#177777           ;IS STACK CORRECT
 9191 042446  001401                            BEQ     404$                   ;YES GO ON
 9192 042450  104002                            ERROR   .+2                 ;MMU ERROR
                                                                            ;NO GO TO ERROR
 9193
 9194 042452  005046                  404$:     CLR     -(SP)                  ;SETUP STACK FOR NEXT TEST
 9195 042454  005237  177572                    INC     @#177572               ;TURN MMU ON
 9196 042460  106603                            MTPD    R3                     ; TEST INSTRUCTION
 9197 042462  022737  000004  177776            CMP     #4,@#177776            ;IS PSW CORRECT
 9198 042470  001401                            BEQ     5$                     ;YES GO ON
 9199 042472  104002                            ERROR   .+2                 ;MMU ERROR
                                                                            ;NO GO TO ERROR
 9200
 9201 042474  005037  177572          5$:       CLR     @#177572               ;TURN MMU OFF
 9202 042500  020327  000000                    CMP     R3,#0                  ;IS R3 CORRECT
 9203 042504  001401                            BEQ     6$                     ;YES GO ON
 9204 042506  104002                            ERROR   .+2                 ;MMU ERROR
 9205                                                                       ;NO GO TO ERROR
```

MEMORY MANAGEMENT TESTS

```
9206 042510  022627  177777          6$:     CMP     (SP)+,#177777           ;IS STACK CORRECT
9207 042514  001401                          BEQ     7$                      ;YES GO ON
9208 042516  104002                          ERROR   +2              ;MMU ERROR
9209                                                                 ;NO GO TO ERROR
9210 042520  012637  000244          7$:     MOV     (SP)+,@#244             ;RESTORE TEST LOCATION
9211
9212                                         ;
9215 042524                          TSMMU7:
9216                                 ;       TEST NON RESIDENT ABORT
9217 042524  005037  177572                  CLR     @#177572                ;MMU OFF
9218 042530  005067  140274                  CLR     FLAG                    ;CLEAR MMU ABORT FLAG
9219 042534  013746  000214                  MOV     @#214,-(SP)             ;SAVE DATA AT TEST LOCATIONS
9220 042540  013746  000216                  MOV     @#216, (SP)             ;
9221 042544  005067  140272                  CLR     SAVMR0                  ;CLEAR STATUS REGS SAVE AREAS
9222 042550  005067  140270                  CLR     SAVMR1                  ;
9223 042554  005067  140266                  CLR     SAVMR2                  ;
9224 042560  004767  074766                  JSR     PC,MMU                  ;INIT MMU
9225 042564  012737  030000  177776          MOV     #30000,@#177776         ;SETUP PSW
9226 042572  012702  000200                  MOV     #200,R2                 ;
9227 042576  012737  077400  177600          MOV     #77400,@#177600         ;SETUP FOR AN ABORT
9228 042604  004767  000164                  JSR     PC,TS7                  ;CAUSE AN ABORT TO OCCUR AND
9229                                                                         ;THEN CHECK IF ABORT FLAG REGISTERED
9230                                                                         ;THIS EVENT AND CHECK IF STATUS REGS
9231                                                                         ;CONTAINED EXPECTED VALUES.
9232                                                                         ;IF NO ABORT OCCURRED THEN GO TO ERROR
9233                                                                         ;OTHERWISE CONTINUE.
9234 042610  012737  077404  177600          MOV     #77404,@#177600         ;SETUP FOR AN ABORT
9235 042616  004767  000152                  JSR     PC,TS7                  ;CAUSE AN ABORT TO OCCUR AND
9236                                                                         ;THEN CHECK IF ABORT FLAG REGISTERED
9237                                                                         ;THIS EVENT AND CHECK IF STATUS REGS
9238                                                                         ;CONTAINED EXPECTED VALUES.
9239                                                                         ;IF NO ABORT OCCURRED THEN GO TO ERROR
9240                                                                         ;OTHERWISE CONTINUE.
9241 042622  012701  000220                  MOV     #220,R1                 ;
9242 042626  004767  074720                  JSR     PC,MMU                  ;INIT MMU
9243 042632  005003                          CLR     R3                      ;SETUP MMR1 EXPECTED DATA
9244 042634  012767  000001  140166          MOV     #1,FLAG                 ;SETUP FLAG FOR AN ABORT
9245 042642  012737  000001  177572          MOV     #1,@#177572             ;TURN MMU ON
9246 042650  012737  100000  177776          MOV     #100000,@#177776        ;SETUP PSW FOR AN ABORT (ILLEGAL MODE)
9247 042656  012241                          MOV     (R2)+,-(R1)             ;CAUSE AN ABORT
9248 042660  004767  000220                  JSR     PC,TSM7                 ;CHECK IF AN ABORT OCCURRED BY
9249                                                                         ;CHECKING ABORT FLAG AND STATUS REGS
9250                                                                         ;IF NO ABORT OCCURRED THEN GO TO ERROR
9251                                                                         ;OTHERWISE CONTINUE.
9252 042664  005067  140152                  CLR     SAVMR0                  ;CLEAR STATUS REGS SAVE AREAS
9253 042670  005067  140150                  CLR     SAVMR1                  ;
9254 042674  005067  140146                  CLR     SAVMR2                  ;
9255 042700  012703  000022                  MOV     #22,R3                  ;SETUP MMR1 EXPECTED DATA
9256 042704  012767  000001  140116          MOV     #1,FLAG                 ;SETUP FLAG FOR AN ABORT
9257 042712  012737  000001  177572          MOV     #1,@#177572             ;TURN MMU ON
9258 042720  012737  020000  177776          MOV     #20000,@#177776         ;SETUP PSW FOR AN ABORT (ILLEGAL MODE)
9259 042726  006522                          MFPI    (R2)+                   ;CAUSE AN ABORT
9260 042730  004767  000150                  JSR     PC,TSM7                 ;CHECK IF AN ABORT OCCURRED BY
9261                                                                         ;CHECKING ABORT FLAG AND STATUS REGS
9262                                                                         ;IF NO ABORT OCCURRED THEN GO TO ERROR
9263                                                                         ;OTHERWISE CONTINUE.
9264 042734  012737  030000  177776          MOV     #30000,@#177776         ;SETUP PSW
```

L13

MEMORY MANAGEMENT TESTS

```
9265 042742  012737  077400  177600          MOV     #77400,@#177600         ;SETUP FOR AN ABORT
9266 042750  005037  177572                  CLR     @#177572                ;MMU OFF
9267 042754  006522                          MFPI    (R2)+                   ;TRY TO CAUSE AN ABORT
9268 042756  012603                          MOV     (SP)+,R3                ;POP THE STACK
9269 042760  012637  000216                  MOV     (SP)+,@#216             ;RESTORE DATA AT TEST LOCATIONS
9270 042764  012637  000214                  MOV     (SP)+,@#214             ;
9271
9272 042770  000167  000154                  JMP     TS7FIN
9273                                 ;
9274                                 ;ROUTINE TO CAUSE AND CHECK  NONRESIDENT ABORTS
9275                                 ;
9276 042774  012767  000001  140026  TS7:    MOV     #1,FLAG                 ;SETUP FOR AN ABORT
9277 043002  012737  000001  177572          MOV     #1,@#177572             ;TURN MMU ON
9278 043010  010701                          MOV     R7,R1                   ;SAVE PC
9279 043012  006522                          MFPI    (R2)+                   ;CAUSE AN ABORT
9280 043014  022767  000000  140006          CMP     #0,FLAG                 ;DID AN ABORT OCCUR
9281 043022  001401                          BEQ     OK7                     ;IF YES GO ON
9282 043024  104002                          ERROR   .+2             ;MMU ERROR
9283                                                                 ;IF NO GO TO ERROR
9284 043026  105067  140010          OK7:    CLRB    SAVMR0                  ;SETUP EXPECTED DATA
9285 043032  022767  100000  140002          CMP     #100000,SAVMR0          ; TEST MMR0 FOR EXPECTED VALUE
9286 043040  001401                          BEQ     OKA7                    ;IF OK THEN CONTINUE
9287 043042  104002                          ERROR   .+2             ;MMU ERROR
9288                                                                 ;NOT OK THEN GO TO ERROR
9289 043044  026727  137774  000022  OKA7:   CMP     SAVMR1,#22              ; TEST MMR1 FOR EXPECTED VALUE
9290 043052  001401                          BEQ     OKAY7                   ;IF OK THEN CONTINUE
9291 043054  104002                          ERROR   .+2             ;MMU ERROR
9292                                                                 ;NOT OK THEN GO TO ERROR
9293 043056  026701  137764          OKAY7:  CMP     SAVMR2,R1               ; TEST MMR2 FOR EXPECTED VALUE
9294 043062  001401                          BEQ     OKAY7A                  ;IF OK THEN CONTINUE
9295 043064  104002                          ERROR   .+2             ;MMU ERROR
9296                                                                 ;NOT OK THEN GO TO ERROR
9297 043066  005067  137750          OKAY7A: CLR     SAVMR0                  ;CLEAR STATUS REGS SAVE AREAS
9298 043072  005067  137746                  CLR     SAVMR1                  ;
9299 043076  005067  137744                  CLR     SAVMR2                  ;
9300 043102  000207                          RTS     PC                      ;RETURN
9301                                 ;
9302                                 ;ROUTINE TO CHECK IF A NONRESIDENT ABORT OCCURRED
9303                                 ;
9304 043104  022767  000000  137716  TSM7:   CMP     #0,FLAG                 ;DID AN ABORT OCCUR
9305 043112  001401                          BEQ     TSMA                    ;IF YES GO ON
9306 043114  104002                          ERROR   .+2             ;MMU ERROR
9307                                                                 ;IF NO THEN GO TO ERROR
9308 043116  042737  040377  003042  TSMA:   BIC     #40377,@#SAVMR0         ;SETUP EXPECTED DATA
9309 043124  022767  100000  137710          CMP     #100000,SAVMR0          ; TEST MMR0 FOR EXPECTED VALUE
9310 043132  001401                          BEQ     TSMB                    ;IF OK THEN CONTINUE
9311 043134  104002                          ERROR   .+2             ;MMU ERROR
9312                                                                 ;IF NO THEN GO TO ERROR
9313 043136  020367  137702          TSMB:   CMP     R3,SAVMR1               ; TEST MMR1 FOR EXPECTED VALUE
9314 043142  001401                          BEQ     TSMC                    ;IF OK THEN CONTINUE
9315 043144  104002                          ERROR   .+2             ;MMU ERROR
9316                                                                 ;IF NOT OK THEN GO TO ERROR
9317 043146  000207                  TSMC:   RTS     PC                      ;RETURN
9318                                 ;
9319 043150  000240                  TS7FIN: NOP
9322 043152                          TSMMU8:
9323                                 ;       TEST READ ONLY ABORTS
```

MEMORY MANAGEMENT TESTS

```
 9324 043152  005037  177572                    CLR     @#177572                ;MMU OFF
 9325 043156  005067  137646                    CLR     FLAG                    ;CLEAR MMU ABORT FLAG
 9326 043162  013746  000244                    MOV     @#244, (SP)             ;SAVE DATA AT TEST LOCATIONS
 9327 043166  013746  000246                    MOV     @#246,-(SP)             ;
 9328 043172  005067  137644                    CLR     SAVMR0                  ;CLEAR STATUS REGS SAVE AREAS
 9329 043176  005067  137642                    CLR     SAVMR1                  ;
 9330 043202  005067  137640                    CLR     SAVMR2                  ;
 9331 043206  004767  074340                    JSR     PC,MMU                  ;INIT MMU
 9332 043212  012737  030000  177776            MOV     #30000,@#177776         ;SETUP PSW
 9333 043220  012702  000244                    MOV     #244,R2                 ;
 9334 043224  012737  077402  177600            MOV     #77402,@#177600         ;SETUP FOR AN ABORT
 9335 043232  012746  000246                    MOV     #246,-(SP)              ;PUSH DATA ONTO THE STACK
 9336 043236  012767  000001  137564            MOV     #1,FLAG                 ;SETUP FLAG FOR AN ABORT
 9337 043244  012737  000001  177572            MOV     #1,@#177572             ;TURN MMU ON
 9338 043252  010701                            MOV     R7,R1                   ;SAVE PC
 9339 043254  006622                            MTPI    (R2)+                   ;CAUSE ABORT
 9340 043256  022767  000000  137544            CMP     #0,FLAG                 ;DID ABORT OCCUR
 9341 043264  001401                            BEQ     1$                      ;IF YES THEN GO ON
 9342 043266  104002                            ERROR   +2                  ;MMU ERROR
 9343                                                                       ;IF NO THEN GO TO ERROR
 9344 043270  105067  137546          1$:       CLRB    SAVMR0                  ;SETUP EXPECTED DATA
 9345 043274  022767  020000  137540            CMP     #20000,SAVMR0           ; TEST MMR0 FOR EXPECTED VALUE
 9346 043302  001401                            BEQ     2$                      ;IF OK THEN CONTINUE
 9347 043304  104002                            ERROR   +2                  ;MMU ERROR
 9348                                                                       ,OTHERWISE GO TO ERROR
 9349 043306  022767  011026  137530  2$:       CMP     #11026,SAVMR1           ; TEST MMR1 FOR EXPECTED VALUE
 9350 043314  001401                            BEQ     3$                      ;IF OK THEN CONTINUE
 9351 043316  104002                            ERROR   +2                  ;MMU ERROR
 9352                                                                       ;OTHERWISE GO TO ERROR
 9353 043320  020167  137522          3$:       CMP     R1,SAVMR2               ; TEST MMR2 FOR EXPECTED VALUE
 9354 043324  001401                            BEQ     4$                      ;IF OK THEN CONTINUE
 9355 043326  104002                            ERROR   +2                  ;MMU ERROR
 9356                                                                       ;OTHERWISE GO TO ERROR
 9357 043330  012737  030000  177776  4$:       MOV     #30000,@#177776         ;SETUP PSW
 9358 043336  012746  000002                    MOV     #2,-(SP)                ;PUSH DATA ONTO STACK
 9359 043342  006622                            MTPI    (R2)+                   ;TRY TO CAUSE ABORT
 9360 043344  012637  000246                    MOV     (SP)+,@#246             ;RESTORE DATA AT TEST LOCATIONS
 9361 043350  012637  000244                    MOV     (SP)+,@#244             ;
 9362
 9365 043354                          TSMMU9:
 9366                                 ;         TEST PAGE LENGTH ERROR ABORTS
 9367 043354  005037  177572                    CLR     @#177572                ;MMU OFF
 9368 043360  005067  137444                    CLR     FLAG                    ;CLEAR MMU ABORT FLAG
 9369 043364  005067  137452                    CLR     SAVMR0                  ;CLEAR STATUS REGS SAVE AREAS
 9370 043370  005067  137450                    CLR     SAVMR1                  ;
 9371 043374  005067  137446                    CLR     SAVMR2                  ;
 9372 043400  012737  030000  177776            MOV     #30000,@#177776         ;SETUP PSW
 9373 043406  004767  074140                    JSR     PC,MMU                  ;INIT MMU
 9374 043412  012703  043674                    MOV     #PLF0,R3                ;LET R3, R1, AND R2 POINT TO THE
 9375 043416  012701  043746                    MOV     #BN0,R1                 ;UPWARD EXPANSION TABLES
 9376 043422  012702  044016                    MOV     #ABORT0,R2              ;
 9377 043426  012737  000026  172516            MOV     #26,@#172516            ;DISABLE USER DATA SPACE
 9378 043434  004767  000050                    JSR     PC,TSM9                 ;TURN MMU ON
 9379                                                                       ;DO RELOCATIONS FOR THE DIFFERENT
 9380                                                                       ;VALUES OF THE PAGE LENGTH FIELD AND
 9381                                                                       ;BLOCK NUMBER.  IF AN ABORT OCCURS
 9382                                                                       ;CHECK TO SEE IF IT WAS SUPPOSED TO.
```

MEMORY MANAGEMENT TESTS

```
                                                                        ;AND IF YES CHECK ABORT FLAG AND
                                                                        ;STATUS REGISTERS.
043440  012703  044066                  MOV     #PLF1,R3                ;LET R3, R1, AND R2 POINT TO THE
043444  012701  044136                  MOV     #BN1,R1                 ;DOWNWARD EXPANSION TABLES
043450  012702  044204                  MOV     #ABORT7,R2              ;
043454  004767  000030                  JSR     PC,TSM9                 ;TURN MMU ON
                                                                        ;DO RELOCATIONS FOR THE DIFFERENT
                                                                        ;VALUES OF THE PAGE LENGTH FIELD AND
                                                                        ;BLOCK NUMBER.  IF AN ABORT OCCURS
                                                                        ;CHECK TO SEE IF IT WAS SUPPOSED TO,
                                                                        ;AND IF YES CHECK ABORT FLAG AND
                                                                        ;STATUS REGISTERS.
043460  005037  177572                  CLR     @#177572                ;MMU OFF
043464  012703  044076                  MOV     #PLF1+10,R3             ;POINT TO A VALUE WHICH SHOULD CAUSE
043470  012701  044146                  MOV     #BN1+10,R1              ;AN ABORT IF MMU IS ON.
043474  011337  177600                  MOV     (R3),@#177600           ;SETUP UIPDR0
043500  006521                          MFPI    (R1)+                   ;DO A RELOCATION
043502  012605                          MOV     (SP)+,R5                ;POP THE STACK

043504  000167  000542                  JMP     TS9FIN

                                ;
                                ;ROUTINE TO CAUSE AND CHECK PAGE LENGTH ERROR ABORTS
                                ;
043510  012337  177600          TSM9:   MOV     (R3)+,@#177600          ;SETUP UIPDR0
043514  010100                          MOV     R1,R0                   ;SAVE A COPY OF R1
043516  012767  000001  137304          MOV     #1,FLAG                 ;SETUP FOR AN ABORT
043524  012737  000001  177572          MOV     #1,@#177572             ;TURN MMU ON
043532  010704                          MOV     R7,R4                   ;SAVE PC
043534  006530                          MFPI    @(R0)+                  ;DO A RELOCATION OPERATION
043536  021227  000000                  CMP     (R2),#0                 ;WAS AN ABORT SUPPOSED TO OCCUR
043542  001007                          BNE     2$                      ;IF YES GO TO 2$
043544  012605                          MOV     (SP)+,R5                ;POP THE STACK
043546  022767  000001  137254          CMP     #1,FLAG                 ;DID AN ABORT OCCUR
043554  001401                          BEQ     1$                      ;NO GO ON
043556  104002                          ERROR   +2              ;MMU ERROR
                                                                ;YES GO TO ERROR
043560  000425                  1$:     BR      6$                      ;
043562  022767  000000  137240  2$:     CMP     #0,FLAG                 ;DID AN ABORT OCCUR
043570  001401                          BEQ     3$                      ;YES GO ON
043572  104002                          ERROR   +2              ;MMU ERROR
                                                                ,NO GO TO ERROR
043574  105067  137242          3$:     CLRB    SAVMR0                  ;SETUP EXPECTED DATA
043600  022767  040000  137234          CMP     #40000,SAVMR0           ; TEST MMR0 FOR EXPECTED VALUE
043606  001401                          BEQ     4$                      ;IF OK THEN CONTINUE
043610  104002                          ERROR   +2              ;MMU ERROR
                                                                ;NOT OK THEN GO TO ERROR
043612  022767  000020  137224  4$:     CMP     #20,SAVMR1              ; TEST MMR1 FOR EXPECTED VALUE
043620  001401                          BEQ     5$                      ;IF OK THEN CONTINUE
043622  104002                          ERROR   +2              ;MMU ERROR
                                                                ;NOT OK THEN GO TO ERROR
043624  020467  137216          5$:     CMP     R4,SAVMR2               ; TEST MMR2 FOR EXPECTED VALUE
043630  001401                          BEQ     6$                      ;IF OK THEN CONTINUE
043632  104002                          ERROR   +2              ;MMU ERROR
                                                                ;NOT OK THEN GO TO ERROR
043634  005067  137170          6$:     CLR     FLAG                    ;CLEAR MMU ABORT FLAG
043640  005067  137176                  CLR     SAVMR0                  ;CLEAR STATUS REGS SAVE AREAS
043644  005067  137174                  CLR     SAVMR1                  ;
```

MEMORY MANAGEMENT TESTS

```
9440 043650  005067  137172                CLR     SAVMR2                  ;
9441 043654  005201                        INC     R1                      ;POINT TO NEXT ENTRY
9442 043656  005201                        INC     R1                      ;
9443 043660  005202                        INC     R2                      ;
9444 043662  005202                        INC     R2                      ;
9445 043664  021327  000777                CMP     (R3),#777               ;HAVE ALL ENTRIES BEEN TRIED
9446 043670  001307                        BNE     TSM9                    ;NO REPEAT
9447 043672  000207                        RTS     PC                      ;YES RETURN
9448                                ;
9449                                ;UPWARD EXPANSION TABLES
9450                                ;
9451 043674  070006                PLF0:   .WORD   70006
9452 043676  070006                        .WORD   70006
9453 043700  070006                        .WORD   70006
9454 043702  013406                        .WORD   13406
9455 043704  020006                        .WORD   20006
9456 043706  004006                        .WORD   04006
9457 043710  040006                        .WORD   40006
9458 043712  070006                        .WORD   70006
9459 043714  024006                        .WORD   24006
9460 043716  004006                        .WORD   04006
9461 043720  014006                        .WORD   14006
9462 043722  012006                        .WORD   12006
9463 043724  002006                        .WORD   02006
9464 043726  001406                        .WORD   01406
9465 043730  004006                        .WORD   04006
9466 043732  002006                        .WORD   02006
9467 043734  000406                        .WORD   00406
9468 043736  007406                        .WORD   07406
9469 043740  001006                        .WORD   01006
9470 043742  003406                        .WORD   03406
9471 043744  000777                        .WORD   777
9472 043746  013000                BNO:    .WORD   013000
9473 043750  016000                        .WORD   016000
9474 043752  017000                        .WORD   017000
9475 043754  002700                        .WORD   002700
9476 043756  014000                        .WORD   014000
9477 043760  002000                        .WORD   002000
9478 043762  004000                        .WORD   004000
9479 043764  007000                        .WORD   007000
9480 043766  002000                        .WORD   002000
9481 043770  000700                        .WORD   000700
9482 043772  004000                        .WORD   004000
9483 043774  001000                        .WORD   001000
9484 043776  000300                        .WORD   000300
9485 044000  000400                        .WORD   000400
9486 044002  001400                        .WORD   001400
9487 044004  000600                        .WORD   000600
9488 044006  000200                        .WORD   000200
9489 044010  001700                        .WORD   001700
9490 044012  000300                        .WORD   000300
9491 044014  000700                        .WORD   000700
9492 044016  000000                ABORT0: .WORD   0
9493 044020  000000                        .WORD   0
9494 044022  000001                        .WORD   1
9495 044024  000000                        .WORD   0
9496 044026  000001                        .WORD   1
```

MEMORY MANAGEMENT TESTS

```
9497 044070  000001                    .WORD   1
9498 044032  000000                    .WORD   0
9499 044034  000000                    .WORD   0
9500 044036  000000                    .WORD   0
9501 044040  000000                    .WORD   0
9502 044042  000001                    .WORD   1
9503 044044  000000                    .WORD   0
9504 044046  000000                    .WORD   0
9505 044050  000001                    .WORD   1
9506 044052  000001                    .WORD   1
9507 044054  000001                    .WORD   1
9508 044056  000001                    .WORD   1
9509 044060  000000                    .WORD   0
9510 044062  000001                    .WORD   1
9511 044064  000000                    .WORD   0
9512                           ;
9513                           ;DOWNWARD EXPANSION TABLES
9514                           ;
9515 044066  000416            PLF1:   .WORD   00416
9516 044070  020016                    .WORD   20016
9517 044072  024016                    .WORD   24016
9518 044074  034016                    .WORD   34016
9519 044076  074016                    .WORD   74016
9520 044100  040016                    .WORD   40016
9521 044102  020016                    .WORD   20016
9522 044104  000016                    .WORD   00016
9523 044106  030016                    .WORD   30016
9524 044110  010016                    .WORD   10016
9525 044112  014016                    .WORD   14016
9526 044114  004016                    .WORD   04016
9527 044116  002016                    .WORD   02016
9528 044120  000416                    .WORD   00416
9529 044122  000016                    .WORD   00016
9530 044124  003416                    .WORD   03416
9531 044126  001016                    .WORD   01016
9532 044130  001416                    .WORD   01416
9533 044132  000416                    .WORD   00416
9534 044134  000777                    .WORD   777
9535 044136  000100            BN1:    .WORD   000100
9536 044140  010000                    .WORD   010000
9537 044142  006000                    .WORD   006000
9538 044144  016000                    .WORD   016000
9539 044146  016000                    .WORD   016000
9540 044150  004000                    .WORD   004000
9541 044152  000000                    .WORD   000000
9542 044154  000000                    .WORD   000000
9543 044156  004000                    .WORD   004000
9544 044160  004000                    .WORD   004000
9545 044162  004000                    .WORD   004000
9546 044164  000000                    .WORD   000000
9547 044166  000300                    .WORD   000300
9548 044170  000000                    .WORD   000000
9549 044172  000400                    .WORD   000400
9550 044174  001000                    .WORD   001000
9551 044176  000100                    .WORD   000100
9552 044200  000400                    .WORD   000400
9553 044202  000200                    .WORD   000200
```

MEMORY MANAGEMENT TESTS

```
 9554 044204  000000                    ABORT7: .WORD   0
 9555 044206  000000                            .WORD   0
 9556 044210  000000                            .WORD   0
 9557 044212  000000                            .WORD   0
 9558 044214  000001                            .WORD   1
 9559 044216  000001                            .WORD   1
 9560 044220  000001                            .WORD   1
 9561 044222  000000                            .WORD   0
 9562 044224  000001                            .WORD   1
 9563 044226  000000                            .WORD   0
 9564 044230  000000                            .WORD   0
 9565 044232  000001                            .WORD   1
 9566 044234  000001                            .WORD   1
 9567 044236  000001                            .WORD   1
 9568 044240  000000                            .WORD   0
 9569 044242  000000                            .WORD   0
 9570 044244  000001                            .WORD   1
 9571 044246  000000                            .WORD   0
 9572 044250  000000                            .WORD   0
 9573                                   ;
 9574 044252  000240                    TS9FIN: NOP
 9577 044254                            TSMM10:
 9578                                   ;       FUNCTIONAL TEST OF BITS <6:1> OF MMR0
 9579 044254  005037  177572                    CLR     @#177572                ;MMU OFF
 9580 044260  005067  136544                    CLR     FLAG                    ;CLEAR MMU ABORT FLAG
 9581 044264  005067  136552                    CLR     SAVMR0                  ;CLEAR STATUS REGS SAVE AREAS
 9582 044270  005067  136550                    CLR     SAVMR1                  ;
 9583 044274  005067  136546                    CLR     SAVMR2                  ;
 9584 044300  004767  073246                    JSR     PC,MMU                  ;INIT MMU
 9585 044304  005037  177776                    CLR     @#177776                ;INIT PSW- PREVIOUS MODE = KERNAL
 9586 044310  012702  020200                    MOV     #20200,R2               ;
 9587 044314  012737  077400  172302            MOV     #77400,@#172302         ;SETUP KIPDR1 TO ABORT
 9588 044322  012767  000001  136500            MOV     #1,FLAG                 ;SETUP FLAG FOR AN ABORT
 9589 044330  012737  000001  177572            MOV     #1,@#177572             ;TURN MMU ON
 9590 044336  010701                            MOV     R7,R1                   ;SAVE PC
 9591 044340  006522                            MFPI    (R2)+                   ;DO A RELOCATION VIA KIPAR1
 9592 044342  012704  100003                    MOV     #100003,R4              ;SETUP EXPECTED DATA
 9593 044346  004767  000202                    JSR     PC,TS10                 ;CHECK IF AN ABORT OCCURRED AND
                                                                                ;IF YES CHECK BITS <6:1> OF MMR0.
 9594
 9595 044352  012737  030000  177776            MOV     #30000,@#177776         ;INIT PSW: PREVIOUS MODE = USER
 9596 044360  004767  073166                    JSR     PC,MMU                  ;INIT MMU
 9597 044364  012737  077400  177636            MOV     #77400,@#177636         ;SETUP UDPDR7 TO ABORT
 9598 044372  012702  160000                    MOV     #160000,R2              ;
 9599 044376  012767  000001  136424            MOV     #1,FLAG                 ;SETUP FLAG FOR AN ABORT
 9600 044404  012737  000001  177572            MOV     #1,@#177572             ;TURN MMU ON
 9601 044412  010701                            MOV     R7,R1                   ;SAVE PC
 9602 044414  106522                            MFPD    (R2)+                   ;DO A RELOCATION VIA UDPAR7
 9603 044416  012704  100177                    MOV     #100177,R4              ;SETUP EXPECTED DATA
 9604 044422  004767  000126                    JSR     PC,TS10                 ;CHECK IF AN ABORT OCCURRED AND
                                                                                ;IF YES CHECK BITS <6:1> OF MMR0.
 9605
 9606 044426  012737  010000  177776            MOV     #10000,@#177776         ;INIT PSW: PREVIOUS MODE= SUPERVISOR
 9607 044434  004767  073112                    JSR     PC,MMU                  ;INIT MMU
 9608 044440  012737  077400  172212            MOV     #77400,@#172212         ;SETUP SIPDR5 TO ABORT
 9609 044446  012702  120000                    MOV     #120000,R2              ;ACCESS PAGE 05
 9610 044452  012767  000001  136350            MOV     #1,FLAG                 ;SETUP FLAG FOR AN ABORT
 9611 044460  012737  000001  177572            MOV     #1,@#177572             ;TURN MMU ON
 9612 044466  010701                            MOV     R7,R1                   ;SAVE PC
```

E14

MEMORY MANAGEMENT TESTS

```
                                        MFPI    (R2)+                   ;DO A RELOCATION VIA SIPAR5
                                        MOV     #100053,R4              ;SETUP EXPECTED DATA:ABORT, PAGE 05
                                        JSR     PC,TS10                 ;CHECK IF AN ABORT OCCURRED AND
                                                                        ;IF YES CHECK BITS <6:1> OF MMR0.

                                ;
                                ;TEST THAT ILLEGAL MODE CAUSES MMU ABORT
                                ;
                                        MOV     #20000,@#177776         ;INIT PSW:SET ILLEGAL PREVIOUS MODE
                                        JSR     PC,MMU                  ;INIT MMU
                                        MOV     #40000,R2               ;SET UP ACCESS TO PAGE 2
                                        MOV     #1,FLAG                 ;SETUP FLAG FOR AN ABORT
                                        MOV     #1,@#177572             ;TURN MMU ON
                                        MOV     R7,R1                   ;SAVE PC
                                        MFPD    (R2)+                   ;DO A RELOCATION
                                        MOV     #100105,R4              ;SETUP EXPECTED DATA:ABORT, ILLEGAL
                                                                        ; PROCESSOR MODE, PAGE 02
                                        JSR     PC,TS10                 ;CHECK IF AN ABORT OCCURRED AND
                                                                        ;IF YES CHECK BITS <6:1> OF MMR0.

                                        JMP     T10FIN

                                ;
                                ;ROUTINE TO CHECK IF A MMU ABORT OCCURRED AND IF STATUS REG MMR0
                                ;CONTAINS EXPECTED DATA
                                ;
TS10:   CMP     @#0,FLAG                                                ;DID AN ABORT OCCUR
        BEQ     1$                                                      ;YES GO ON
        ERROR   +2                      ;MMU ERROR
                                        ;NO GO TO ERRC
                                1$:     CMP     R4,SAVMR0               ; TEST      OR EXPECTED DATA
        BEQ     2$                                                      ;OK GO on
        ERROR   +2                      ;MMU ERROR
                                        ;NO GO TO ERROR
                                2$:     CMP     @#22,SAVMR1             ; TEST MMR1 FOR EXPECTED DATA
        BEQ     3$                                                      ;OK GO ON
        ERROR   +2                      ;MMU ERROR
                                        ;NO GO TO ERROR
                                3$:     CMP     R1,SAVMR2               ; TEST MMR2 FOR EXPECTED DATA
        BEQ     4$                                                      ;OK GO ON
        ERROR   +2                      ;MMU ERROR
                                        ;NO GO TO ERROR
                                4$:     CLR     SAVMR0                  ;CLEAR MMU STATUS REGS SAVE AREAS
                                        CLR     SAVMR1                  ;
                                        CLR     SAVMR2                  ;
                                        RTS     PC                      ;RETURN

                                ;
                                T10FIN:
                                TSMM11:
                                ;       TEST DATA SPACE BITS MMR3
                                        CLR     @#177572                ;MMU OFF
                                        CLR     FLAG                    ;CLEAR MMU ABORT FLAG
                                        MOV     #30000,@#177776         ;SETUP PSW
                                        MOV     #26,R1                  ;SETUP FIRST MMR3 VALUE
                                        MOV     #177610,R3              ;POINT TO UIPDR4
                                        MOV     #21,R4                  ;SETUP SECOND MMR3 VALUE
                                        JSR     PC,TS11                 ; TEST ENABLE USER DATA SPACE BIT
                                        MOV     #0,@#177776             ;SETUP PSW
                                        MOV     #23,R1                  ;SETUP FIRST MMR3 VALUE
```

F14

MEMORY MANAGEMENT TESTS

```
 9672 044706  012703  172310                MOV     #172310,R3              ;POINT TO KIPDR4
 9673 044712  012704  000024                MOV     #24,R4                  ;SETUP SECOND MMR3 VALUE
 9674 044716  004767  000032                JSR     PC,TS11                 ; TEST ENABLE KERNEL DATA SPACE BIT
 9675 044722  012737  010000  177776        MOV     #10000,@#177776         ;SETUP PSW
 9676 044730  012701  000025                MOV     #25,R1                  ;SETUP FIRST MMR3 VALUE
 9677 044734  012703  172210                MOV     #172210,R3              ;POINT TO SIPDR4
 9678 044740  012704  000022                MOV     #22,R4                  ;SETUP SECOND MMR3 VALUE
 9679 044744  004767  000004                JSR     PC,TS11                 ; TEST ENABLE SUPERVISOR DATA SPACE BIT
 9680
 9681 044750  000167  000120                JMP     T11FIN
 9682                                    ;
 9683                                    ;ROUTINE TO TEST ENABLE DATA SPACE BITS OF MMR3
 9684                                    ;
 9685 044754  004767  072572        TS11:   JSR     PC,MMU                  ;INIT MMU
 9686 044760  010137  172516                MOV     R1,@#172516             ;DISABLE DATA SPACE OF MODE UNDER TEST
 9687 044764  012713  077400                MOV     #77400,(R3)             ;SETUP IPDR TO ABORT
 9688 044770  012702  100000                MOV     #100000,R2              ;
 9689 044774  012767  000001  136026        MOV     #1,FLAG                 ;SETUP FLAG FOR AN ABORT
 9690 045002  012737  000001  177572        MOV     #1,@#177572             ;MMU ON
 9691 045010  106522                        MFPD    (R2)+                   ;DO A RELOCATION
 9692 045012  022767  000000  136010        CMP     #0,FLAG                 ;DID AN ABORT OCCUR
 9693 045020  001401                        BEQ     1$                      ;YES GO ON
 9694 045022  104002                        ERROR   +2                  ;MMU ERROR
                                          ;NO GO TO ERROR
 9695
 9696 045024  010437  172516        1$:     MOV     R4,@#172516             ;ENABLE DATA SPACE OF MODE UNDER TEST
 9697 045030  012702  100000                MOV     #100000,R2              ;
 9698 045034  012767  000001  135766        MOV     #1,FLAG                 ;SETUP FLAG FOR AN ABORT
 9699 045042  012737  000001  177572        MOV     #1,@#177572             ;MMU ON
 9700 045050  106522                        MFPD    (R2)+                   ;DO A RELOCATION
 9701 045052  005726                        TST     (SP)+                   ;POP THE STACK
 9702 045054  022767  000001  135746        CMP     #1,FLAG                 ;DID AN ABORT OCCUR
 9703 045062  001401                        BEQ     2$                      ;NO GO ON
 9704 045064  104002                        ERROR   +2                  ;MMU ERROR
 9705                                                                   ;YES GO TO ERROR
 9706 045066  005067  135736        2$:     CLR     FLAG                    ;CLEAR MMU ABORT FLAG
 9707 045072  000207                        RTS     PC                      ;RETURN
 9708                                    ;
 9709 045074                        T11FIN:
 9712 045074                        TSMM12:
 9713                                    ;       MMR1 FUNCTIONAL TEST
 9714 045074  005037  177572                CLR     @#177572                ;MMU OFF
 9715 045100  005067  135724                CLR     FLAG                    ;CLEAR MMU ABORT FLAG
 9716 045104  005067  135734                CLR     SAVMR1                  ;CLEAR STATUS REG SAVE AREA
 9717 045110  004767  072436                JSR     PC,MMU                  ;INIT MMU
 9718 045114  012737  030000  177776        MOV     #30000,@#177776         ;INIT PSW
 9719 045122  012704  100200                MOV     #100200,R4              ;SETUP TEST LOCATIONS
 9720 045126  010401                        MOV     R4,R1                   ;
 9721 045130  012705  100101                MOV     #100101,R5              ;
 9722 045134  010502                        MOV     R5,R2                   ;
 9723 045136  012737  000020  172516        MOV     #20,@#172516            ;INIT MMR3
 9724 045144  012737  077402  172310        MOV     #77402,@#172310         ;SETUP KIPDR4 TO ABORT
 9725 045152  012703  006414                MOV     #6414,R3                ;SETUP EXPECTED DATA FOR MMR1
 9726 045156  012767  000001  135644        MOV     #1,FLAG                 ;SETUP FLAG FOR AN ABORT
 9727 045164  012737  000001  177572        MOV     #1,@#177572             ;TURN MMU ON
Z 9728 045172  010767  135614                MOV     R7,SLOC00               ;SAVE PC
 9729 045176  112425                        MOVB    (R4)+,(R5)+             ;DO A RELOCATION
 9730 045200  004767  000206                JSR     PC,TS12                 ;CHECK IF AN ABORT OCCURRED AND IF
```

MEMORY MANAGEMENT TESTS

```
    9731
    9732 045204  012703  175011                MOV     #175011,R3      ;YES IF MMR1 EQUALS EXPECTED DATA
    9733 045210  012767  000001  135612        MOV     #1,FLAG         ;SETUP EXPECTED DATA FOR MMR1
    9734 045216  012737  000001  177572        MOV     #1,@#177572     ;SETUP FLAG FOR AN ABORT
Z   9735 045224  010767  135562                MOV     R7,SLOC00       ;TURN MMU ON
    9736 045230  112142                        MOVB    (R1)+,-(R2)     ;SAVE PC
    9737 045232  004767  000154                JSR     PC,TS12         ;DO A RELOCATION
    9738                                                               ;CHECK IF AN ABORT OCCURRED AND IF
    9739 045236  012703  006771                MOV     #6771,R3        ;YES IF MMR1 EQUALS EXPECTED DATA
    9740 045242  012767  000001  135560        MOV     #1,FLAG         ;SETUP EXPECTED DATA FOR MMR1
    9741 045250  012737  000001  177572        MOV     #1,@#177572     ;SETUP FLAG FOR AN ABORT
Z   9742 045256  010767  135530                MOV     R7,SLOC00       ;TURN MMU ON
    9743 045262  114125                        MOVB    -(R1),(R5)+     ;SAVE PC
    9744 045264  004767  000122                JSR     PC,TS12         ;DO A RELOCATION
    9745                                                               ;CHECK IF AN ABORT OCCURRED AND IF
    9746 045270  012703  006411                MOV     #6411,R3        ;YES IF MMR1 EQUALS EXPECTED DATA
    9747 045274  012767  000001  135526        MOV     #1,FLAG         ;SETUP EXPECTED DATA FOR MMR1
    9748 045302  012737  000001  177572        MOV     #1,@#177572     ;SETUP FLAG FOR AN ABORT
Z   9749 045310  010767  135476                MOV     R7,SLOC00       ;TURN MMU ON
    9750 045314  112125                        MOVB    (R1)+,(R5)+     ;SAVE PC
    9751 045316  004767  000070                JSR     PC,TS12         ;DO A RELOCATION
    9752                                                               ;CHECK IF AN ABORT OCCURRED AND IF
    9753 045322  012703  171025                MOV     #171025,R3      ;YES IF MMR1 EQUALS EXPECTED DATA
    9754 045326  012767  000001  135474        MOV     #1,FLAG         ;SETUP EXPECTED DATA FOR MMR1
    9755 045334  012737  000001  177572        MOV     #1,@#177572     ;SETUP FLAG FOR AN ABORT
Z   9756 045342  010767  135444                MOV     R7,SLOC00       ;T'URN MMU ON
    9757 045346  012542                        MOV     (R5)+,-(R2)     ;  .VE PC
    9758 045350  004767  000036                JSR     PC,TS12         ;DO A RELOCATION
    9759                                                               ;CHECK IF AN ABORT OCCURRED AND IF
    9760 045354  012703  012762                MOV     #12762,R3       ;YES IF MMR1 EQUALS EXPECTED DATA
    9761 045360  012767  000001  135442        MOV     #1,FLAG         ;SETUP EXPECTED DATA FOR MMR1
    9762 045366  012737  000001  177572        MOV     #1,@#177572     ;SETUP FLAG FOR AN ABORT
Z   9763 045374  010767  135412                MOV     R7,SLOC00       ;TURN MMU ON
    9764 045400  014225                        MOV     -(R2),(R5)+     ;SAVE PC
    9765 045402  004767  000004                JSR     PC,TS12         ;DO A RELOCATION
    9766                                                               ;CHECK IF AN ABORT OCCURRED AND IF
    9767                                                               ;YES IF MMR1 EQUALS EXPECTED DATA
    9768 045406  000167  000046                JMP     T12FIN
    9769                        ;
    9770                        ;ROUTINE TO CHECK IF AN ABORT OCCURRED AND IF MMR1 EQUALS EXPECTED DATA
    9771                        ;
    9772 045412  022767  000000  135410  TS12:  CMP     #0,FLAG                 ;DID AN ABORT OCCUR
    9773 045420  001401                        BEQ     1$                      ;YES GO ON
    9774 045422  104002                        ERROR   +2              ;MMU ERROR
    9775                                                               ;NO GO TO ERROR
    9776 045424  020367  135414          1$:   CMP     R3,SAVMR1               ; TEST MMR1 FOR EXPECTED DATA
    9777 045430  001401                        BEQ     2$                      ;OK GO ON
    9778 045432  104002                        ERROR   +2              ;MMU ERROR
    9779                                                               ;NO GO TO ERROR
    9780 045434  026767  135352  135404  2$:   CMP     SLOC00,SAVMR2           ; TEST MMR2 FOR EXPECTED DATA
    9781 045442  001401                        BEQ     3$                      ;OK GO ON
    9782 045444  104002                        ERROR   +2              ;MMU ERROR
    9783                                                               ;NO GO TO ERROR
    9784 045446  005067  135372          3$:   CLR     SAVMR1                  ;CLEAR STATUS REG SAVE AREA
    9785 045452  005067  135370                CLR     SAVMR2                  ;
    9786 045456  000207                        RTS     PC                      ;RETURN
    9787                        ;
```

MEMORY MANAGEMENT TESTS

```
 9788 045460  000240                    T12FIN: NOP
 9798 045462                             TSMM13:
 9799                                    ;       ADDER RELOCATION TEST PART A
 9800                                    ;;*****************************************************************
                                         ;(NEED 16 BITS OF MEMORY ADDRESSING)
 9801 045462  005037  177572                     CLR     @#177572                ;MMU OFF
 9802 045466  005067  135336                     CLR     FLAG                    ;CLEAR MMU ABORT FLAG
 9803 045472  005037  177776                     CLR     @#177776                ;INIT PSW
 9804 045476  004767  072050                     JSR     PC,MMU                  ;INIT MMU
 9805 045502  012737  000020  172516             MOV     #20,@#172516            ;INIT MMR3
 9806 045510  012703  045664                     MOV     #PARAD1,R3              ;SETUP PARS WITH TEST VALUES
 9807 045514  012701  045716                     MOV     #PARVA1,R1              ;
 9808 045520  012133              1$:    MOV     (R1)+,@(R3)+            ;
 9809 045522  021127  000333                     CMP     (R1),#333               ;
 9810 045526  001374                             BNE     1$                      ;
 9811 045530  012703  046002                     MOV     #PHY1,R3                ;SET POINTERS TO ADDER PART A
 9812 045534  012701  045750                     MOV     #VIR1,R1                ; TEST TABLES.
 9813 045540  012702  046034                     MOV     #MODE1,R2               ;
 9814 045544  012237  177776              2$:    MOV     (R2)+,@#177776          ;INIT PSW
 9815 045550  013305                             MOV     @(R3)+,R5               ;SAVE DATA AT PHYSICAL ADDRESS
 9816 045552  012737  000001  177572             MOV     #1,@#177572             ;TURN MMU ON
 9817 045560  006531                             MFPI    @(R1)+                  ;SAVE DATA AT RELOCATED VIRTUAL ADDRESS
 9818 045562  012604                             MOV     (SP)+,R4                ;
 9819 045564  005037  177572                     CLR     @#177572                ;TURN MMU OFF
 9820 045570  020504                             CMP     R5,R4                   ;IS DATA EQUAL TO EXPECTED
 9821 045572  001401                             BEQ     3$                      ;YES GO ON
 9822 045574  104002                             ERROR   +2              ;MMU ERROR
                                                                         ;NO IT IS AN ADDER ERROR
 9823
 9824 045576  021327  000111              3$:    CMP     (R3),#111               ;ARE WE READY TO TEST DATA SPACE
 9825 045602  001360                             BNE     2$                      ;NO GO TO 2$
 9826 045604  005203                             INC     R3                      ;POINT TO DATA SPACE VALUES
 9827 045606  005203                             INC     R3                      ;
 9828 045610  005201                             INC     R1                      ;
 9829 045612  005201                             INC     R1                      ;
 9830 045614  005202                             INC     R2                      ;
 9831 045616  005202                             INC     R2                      ;
 9832 045620  012237  177776                     MOV     (R2)+,@#177776          ;INIT PSW
 9833 045624  013305                             MOV     @(R3)+,R5               ;SAVE DATA AT PHYSICAL ADDRESS
 9834 045626  012737  000027  172516             MOV     #27,@#172516            ;INIT MMR3
 9835 045634  012737  000001  177572             MOV     #1,@#177572             ;TURN MMU ON
 9836 045642  106531                             MFPD    @(R1)+                  ;SAVE DATA AT RELOCATED VIRTUAL ADDRESS
 9837 045644  012604                             MOV     (SP)+,R4                ;POP THE STACK
 9838 045646  005037  177572                     CLR     @#177572                ;TURN MMU OFF
 9839 045652  020504                             CMP     R5,R4                   ;IS DATA EQUAL TO EXPECTED
 9840 045654  001401                             BEQ     4$                      ;YES GO ON
 9841 045656  104002                             ERROR   +2              ;MMU ERROR
                                                                         ;NO IT IS AN ADDER ERROR
 9842
 9843 045660                      4$:
 9844 045660  000167  000202                     JMP     T13FIN
 9845                                    ;
 9846                                    ;ADDER TEST PART A TABLES
 9847                                    ;
 9848 045664  172240              PARAD1: .WORD   172240
 9849 045666  177642                             .WORD   177642
 9850 045670  172252                             .WORD   172252
 9851 045672  177640                             .WORD   177640
 9852 045674  172242                             .WORD   172242
```

MEMORY MANAGEMENT TESTS

```
 9853 045676  172254                        .WORD   172254
 9854 045700  177652                        .WORD   177652
 9855 045702  177644                        .WORD   177644
 9856 045704  172246                        .WORD   172246
 9857 045706  177654                        .WORD   177654
 9858 045710  172250                        .WORD   172250
 9859 045712  177660                        .WORD   177660
 9860 045714  000333                        .WORD   333
 9861 045716  000000              PARVA1:   .WORD   000000
 9862 045720  000010                        .WORD   000010
 9863 045722  177777                        .WORD   177777
 9864 045724  177601                        .WORD   177601
 9865 045726  000010                        .WORD   000010
 9866 045730  000052                        .WORD   000052
 9867 045732  000070                        .WORD   000070
 9868 045734  000010                        .WORD   000010
 9869 045736  000010                        .WORD   000010
 9870 045740  000060                        .WORD   000060
 9871 045742  000000                        .WORD   000000
 9872 045744  000010                        .WORD   000010
 9873 045746  000333                        .WORD   333
 9874 045750  000000              VIR1:     .WORD   000000
 9875 045752  025000                        .WORD   025000
 9876 045754  135224                        .WORD   135224
 9877 045756  017700                        .WORD   017700
 9878 045760  033000                        .WORD   033000
 9879 045762  145252                        .WORD   145252
 9880 045764  121000                        .WORD   121000
 9881 045766  043000                        .WORD   043000
 9882 045770  075000                        .WORD   075000
 9883 045772  142000                        .WORD   142000
 9884 045774  117700                        .WORD   117700
 9885 045776  000111                        .WORD   111
 9886 046000  007000                        .WORD   007000
 9887 046002  000000              PHY1:     .WORD   000000
 9888 046004  006000                        .WORD   006000
 9889 046006  015124                        .WORD   015124
 9890 046010  000000                        .WORD   000000
 9891 046012  014000                        .WORD   014000
 9892 046014  012452                        .WORD   012452
 9893 046016  010000                        .WORD   010000
 9894 046020  004000                        .WORD   004000
 9895 046022  016000                        .WORD   016000
 9896 046024  010000                        .WORD   010000
 9897 046026  017700                        .WORD   017700
 9898 046030  000111                        .WORD   111
 9899 046032  010000                        .WORD   010000
 9900 046034  010000              MODE1:    .WORD   010000
 9901 046036  030000                        .WORD   030000
 9902 046040  010000                        .WORD   010000
 9903 046042  030000                        .WORD   030000
 9904 046044  010000                        .WORD   010000
 9905 046046  010000                        .WORD   010000
 9906 046050  030000                        .WORD   030000
 9907 046052  030000                        .WORD   030000
 9908 046054  010000                        .WORD   010000
 9909 046056  030000                        .WORD   030000
```

MEMORY MANAGEMENT TESTS

```
 9910 046060  010000                          .WORD   010000
 9911 046062  000111                          .WORD   111
 9912 046064  030000                          .WORD   030000
 9913                                ;
 9914 046066                         T13FIN:
 9925 046066                         TS1822:
 9926                                ;       TEST 22/18 BIT ADDRESS OPTION
 9927                                ;;*****************************************************************
                                     ;CHECK THE SOFTWARE SWITCH REGISTER TO DETERMINE IF THIS IS A 22 BIT OR AN
                                     ;18 BIT ADDRESS SYSTEM. BIT 08 IN THE SWR=1 INDICATES AN 18 BIT SYSTEM.
                                     ;IF WE'RE IN A 22 BIT SYSTEM WE CAN PERFORM SOME EXTRA TESTS.
 9928 046066  032777  000400  133044          BIT     #BIT08,@SWR             ;IS BIT 08 SET?
 9929 046074  001405                          BEQ     100$                    ;BRANCH IF ITS NOT
 9930 046076  062737  000001  001204          ADD     #1,@#$TESTN             ;KEEP TEST NUMBERS IN ORDER
 9931                                                                         ;ADD 1 FOR THE TESTS WE'RE SKIPPING
 9932 046104  000167  001422                  JMP     T14FIN                  ;SKIP OVER THESE TESTS IF IT IS
 9933                                ;
 9934                                ;IF THIS IS A 22 BIT SYSTEM CHECK THE 22/18 BIT ADDRESS OPTION
 9935                                ;
 9936                                ;TO TEST 22 BIT ADDRESSING WE DO THE FOLLOWING:
 9937                                ;       A. ENABLE 22 BIT ADDRESSING MODE
 9938                                ;       B. CLEAR ADDRESS 0
 9939                                ;       C. WRITE PHYSICAL ADDRESS 17000000 WITH ALL ONES
 9940                                ;       D. CHECK ADDRESS 0
 9941                                ;IF ADDRESS 0 IS UNCHANGED (=0) OR A TIME OUT OCCURRED, IT INDICATES
 9942                                ;22 BIT MODE IS FUNCTIONING.
 9943                                ;IF ADDRESS 0 =177777 IT INDICATES THAT 22 BIT MODE IS NOT FUNCTIONING.
 9944
 9945 046110  013767  000004  134674  100$:   MOV     @#4,SLOC00              ;SAVE VECTOR
 9946 046116  013767  000006  134670          MOV     @#6,SLOC01              ;SAVE VECTOR
 9947 046124  012737  046204  000004          MOV     #1$,@#4                 ;SET VECTOR FOR NXM TRAP
 9948 046132  012737  000340  000006          MOV     #340,@#6                ;
 9949 046140  012767  000020  124350          MOV     #20,SR3                 ;ENABLE 22 BIT MODE ADDRESSING
 9950 046146  012767  170000  124200          MOV     #170000,KIPAR6          ;SET KIPAR6 FOR 1920 1924KW ADDR RANGE
 9951 046154  012767  000001  131410          MOV     #1,SR0                  ;ENABLE MMU
 9952 046162  005067  131612                  CLR     0                       ;CLEAR ADDR 0
 9953 046166  012737  177777  140000          MOV     #177777,@#140000        ;MOVE ALL ONES TO ADDR 17000000 VIA KIPAR6
 9954                                                                         ;A TIME OUT ERROR OR
 9955 046174  005767  131600                  TST     0                       ;ADDR 0 REMAINING CLEAR INDICATES
 9956                                                                         ;THAT 22 BIT ADDRESS MODE IS WORKING AND
 9957                                                                         ;THAT SOME FURTHER TESTS SHOULD BE PERFORMED
 9958 046200  001405                          BEQ     2$                      ;IF ADDR 0 =177777
 9959                                                                         ;ERROR! 22 BIT ADDRESS MODE BAD
 9960 046202  104002                          ERROR   +2              ;MMU ERROR
 9961 046204  012706  001000          1$:     MOV     #STBOT,SP               ;GOT HERE AS A RESULT OF NXM TRAP
 9962                                                                         ;CLEAN UP THE STACK
 9963 046210  005037  177766                  CLR     @#177766                ;CLEAR CPU ERROR REGISTER
 9964 046214  012737  046254  000004  2$:     MOV     #3$,@#4                 ;SET UP VECTOR FOR NXM TRAP
 9965 046222  042767  000020  124266          BIC     #BIT04,SR3              ;SET 18 BIT ADDRESSING MODE IN SR3
 9966 046230  012767  170000  124116          MOV     #170000,KIPAR6          ;SET KIPAR6 SO THAT BITS 18 21 SHOULD
 9967                                                                         ;BE ASSERTED IF 22 BIT ADR WAS ENABLED
 9968 046236  012737  177777  140000          MOV     #177777,@#140000        ;TRY TO WRITE ADDR 17000000 VIA KIPAR6
 9969                                                                         ;ADDR 0 SHOULD = 177777. A TIME OUT
 9970 046244  022737  177777  000000          CMP     #177777,@#0             ;OR ADDR 0 = ZERO INDICATES AN ERROR
 9971 046252  001403                          BEQ     4$                      ;GO TO NEXT TEST IF ADDR 0=177777
 9972 046254  005067  131312          3$:     CLR     SR0                     ;DISABLE MMU BEFORE ERROR.
 9973                                                                         ;ERROR! 18 BIT ADDR OPTION IS N.G.
```

MEMORY MANAGEMENT TESTS

```
 9974 046260  104002                          ERROR   +2              ;MMU ERROR
 9975                                 ;
 9976                                 ;TEST ADDRESS BITS 18 THRU 21
 9977                                 ;
 9978
 9979 046262  052767  000020  124226  4$:     BIS     #BIT04,SR3             ;ENABLE 22 BIT ADDRESSING MODE
 9980 046270  012767  046332  131506          MOV     #5$,4                  ;SET UP FOR NXM TRAP
 9981 046276  005067  131476                  CLR     0                      ;CLEAR ADDRESS 0
 9982 046302  012767  010000  124044          MOV     #10000,KIPAR6          ;TEST ADDRESS BIT 18
 9983 046310  012737  177777  140000          MOV     #177777,@#140000       ;WRITE ALL ONES TO ADDR 1000000
 9984 046316  005767  131456                  TST     0                      ;TEST ADDRESS 0. SHOULD = ZERO
 9985 046322  001403                          BEQ     5$                     ;BRANCH IF ADDRESS 0=0
 9986 046324  005067  131242                  CLR     SR0                    ;DISABLE MMU BEFORE ERROR
 9987 046330  104002                          ERROR   +2              ;MMU ERROR
 9988                                         ;ERROR! BIT 18 DID NOT ASSERT
 9989 046332  012737  046374  000004  5$:     MOV     #6$,@#4                ;SET UP FOR NXM TRAP
 9990 046340  005067  131434                  CLR     0                      ;CLEAR ADDR 0
 9991 046344  012767  020000  124002          MOV     #20000,KIPAR6          ;TEST ADDRESS BIT 19
 9992 046352  012737  177777  140000          MOV     #177777,@#140000       ;WRITE ALL ONES TO ADDR 2000000
 9993 046360  005767  131414                  TST     0                      ;TEST ADDR 0. SHOULD- ZERO
 9994 046364  001403                          BEQ     6$                     ;BRANCH IF ADDRESS 0=0
 9995 046366  005067  131200                  CLR     SR0                    ;DISABLE MMU BEFORE ERROR
 9996 046372  104002                          ERROR   +2              ;MMU ERROR
 9997                                         ;ERROR! BIT 19 DID NOT ASSERT
 9998 046374  012737  046436  000004  6$:     MOV     #7$,@#4                ;SET UP FOR NXM TRAP
 9999 046402  005067  131372                  CLR     0                      ;CLEAR ADDR 0
10000 046406  012767  040000  123740          MOV     #40000,KIPAR6          ;TEST ADDRESS BIT 20
10001 046414  012737  177777  140000          MOV     #177777,@#140000       ;WRITE ALL ONES TO ADDR 4000000
10002 046422  005767  131352                  TST     0                      ;TEST ADDR 0. SHOULD =0
10003 046426  001403                          BEQ     7$                     ;BRANCH IF ADDRESS 0 =0
10004 046430  005067  131136                  CLR     SR0                    ;DISABLE MMU BEFORE ERROR
10005 046434  104002                          ERROR   +2              ;MMU ERROR
10006                                         ;ERROR! BIT 20 DID NOT ASSERT
10007 046436  012737  046500  000004  7$:     MOV     #8$,@#4                ;SET UP FOR NXM
10008 046444  005067  131330                  CLR     0                      ;CLEAR ADDRESS 0
10009 046450  012767  100000  123676          MOV     #100000,KIPAR6         ;TEST ADDRESS BIT 21
10010 046456  012737  177777  140000          MOV     #177777,@#140000       ;WRITE ALL ONES AT ADDR 10000000
10011 046464  005767  131310                  TST     0                      ;CHECK ADDRESS 0. SHOULD = 0
10012 046470  001403                          BEQ     8$                     ;BRANCH IF ADDR 0 = 0
10013 046472  005067  131074                  CLR     SR0                    ;DISABLE MMU    BEFORE ERROP
10014 046476  104002                          ERROR   +2              ;MMU ERROR
10015                                         ;ERROR! ADDR BIT 21 DID NOT ASSERT
10016 046500  005067  131066          8$:     CLR     SR0                    ;DISABLE MMU
10017 046504  005037  177766                  CLR     @#177766               ;CLEAR CPU ERROR REGISTER
10018 046510  012706  001000                  MOV     #STBOT,R6              ;RESET STACK POINTER
10019 046514  013737  003012  000004          MOV     @#SLOC00,@#4           ;RESTORE VECTORS
10020 046522  013737  003014  000006          MOV     @#SLOC01,@#6           ;
10021
10030 046530                          TSMM14:
10031                                 ;       ADDER RELOCATION TEST PART B
10032                                 ;;*****************************************************************
                                      ;(NEED 22 BITS OF MEMORY ADDRESSING)
10033 046530  005037  177572                  CLR     @#SR0                  ;TURN OFF MMU.
10034 046534  005067  131226                  CLR     CPEREG                 ;CLEAR THE CPU ERROR REGISTER
10035 046540  013737  000004  003012          MOV     @#4,@#SLOC00           ;SAVE LOC 4 IN SLOC00.
10036 046546  013737  000006  003014          MOV     @#6,@#SLOC01           ;SAVE LOC 6 IN SLOC01.
10037 046554  012737  047244  000004          MOV     #NXMTRP,@#4            ;SET UP FOR TIMEOUT TRAP
```

MEMORY MANAGEMENT TESTS

```
10038 046562  012737  000340  000006          MOV     #340,@#6                ;SET UP FOR TIMEOUT TRAP
10039 046570  005037  172340                  CLR     @#KIPAR0                ;SET KER PARO FOR 1ST 4KW OF MEMORY.
10040 046574  012767  077406  123476          MOV     #77406,KIPDR0           ;SET KER PDR FOR 4KW R/W ACCESS
10041 046602  012737  177500  172354          MOV     #177500,@#KIPAR6        ;SET UP KERNEL PAGE ADDR REG 6
10042                                                                         ;FOR HIGHEST 4K WORDS OF NON I/O
10043                                                                         ;FOR 2 MEG WORDS OF MEMORY.
10044 046610  012767  077406  123476          MOV     #77406,KIPDR6           ;SET KER PDR6 FOR 4KW R/W ACCESS.
10045 046616  012737  000020  172516          MOV     #20,@#SR3               ;ENABLE 22 BIT ADDRESSING.
10046 046624  012737  000001  177572          MOV     #1,@#SR0                ;TURN ON THE MMU.
10047 046632  005737  147776                  TST     @#147776                ;ATTEMPT TO ADDR. LAST MEMORY ADDR. $$$
10048                       ;*************WILL TRAP TO 4 IF 2 MEG WORDS OF MEMORY NOT AVAILABLE*********
10049 046636  013737  003012  000004          MOV     @#SLOC00,@#4            ;RESTORE LOC 4
10050 046644  013737  003014  000006          MOV     @#SLOC01,@#6            ;RESTORE LOC 6
10051 046652  005037  177572                  CLR     @#177572                ;MMU OFF
10052 046656  005037  003030                  CLR     @#FLAG                  ;CLEAR MMU ABORT FLAG
10053 046662  004767  070664                  JSR     PC,MMU                  ;INIT MMU
10054 046666  012737  010000  177776          MOV     #10000,@#177776         ;INIT PSW
10055 046674  012737  000020  172516          MOV     #20,@#172516            ;INIT MMR3
10056 046702  052737  001000  177746          BIS     #1000,@#177746          ;TURN CACHE TEST FEATURE ON
10057 046710  012704  047446                  MOV     #PARVA3,R4              ;SET POINTERS TO INIT TABLE
10058 046714  012701  047500                  MOV     #VIR3,R1                ;
10059 046720  012437  172246          1$:     MOV     (R4)+,@#172246          ;INIT SIPAR3
10060 046724  012737  000001  177572          MOV     #1,@#177572             ;TURN MMU ON
10061 046732  012746  125252                  MOV     #125252,-(SP)           ;PUSH BACKGROUND DATA ON TO THE STACK
10062 046736  006671  000000                  MTPI    @0(R1)                  ;WRITE DATA TO PHYSICAL ADDRESS
10063 046742  006531                          MFPI    @(R1)+                  ;WRITE DATA AT PHYSICAL ADDRESS TO STACK
10064 046744  022726  125252                  CMP     #125252,(SP)+           ;IS DATA EQUAL TO EXPECTED
10065 046750  001403                          BEQ     2$                      ;YES GO ON
10066 046752  005037  177572                  CLR     @#177572                ;TURN MMU OFF
10067 046756  104002                          ERROR   +2                      ;MMU ERROR
10068                                                                         ;NOT EQUAL GO TO ERROR
10069 046760  005037  177572          2$:     CLR     @#177572                ;TURN MMU OFF
10070 046764  021427  000333                  CMP     (R4),#333               ;ARE WE DONE
10071 046770  001353                          BNE     1$                      ;NO GO TO 1$
10072 046772  012704  047330                  MOV     #PARVA2,R4              ;SET POINTERS TO PAR INIT TABLES
10073 046776  012701  047276                  MOV     #PARAD2,R1              ;
10074 047002  012431                  3$:     MOV     (R4)+,@(R1)+            ;INIT PARS
10075 047004  021127  000333                  CMP     (R1),#333               ;ARE WE DONE
10076 047010  001374                          BNE     3$                      ;NO, GO TO 3$
10077 047012  012704  047414                  MOV     #MODE2,R4               ;SET POINTERS TO ADDER PART B TABLES
10078 047016  012701  047362                  MOV     #VIR2,R1                ;
10079 047022  012702  047446                  MOV     #PARVA3,R2              ;
10080 047026  012703  047500                  MOV     #VIR3,R3                ;
10081 047032  004767  000076          4$:     JSR     PC,TS14                 ;WRITE DATA TO PHYSICAL ADDRESS AND THEN
10082                                                                         ;CHECK IF DATA AT PHYSICAL ADDRESS IS
10083                                                                         ;EQUAL TO EXPECTED AND IF NOT DETERMINE
10084                                                                         ;IF IT IS AN ADDER ERROR OR A MEMORY ERROR
10085 047036  021127  000111                  CMP     (R1),#111               ;HAVE WE DONE ALL THE 22 BIT MODE I SPACE
10086                                                                         ;CASES
10087 047042  001373                          BNE     4$                      ;NO GO TO 4$
10088 047044  005201                          INC     R1                      ;POINT TO 22 BIT MODE D SPACE CASE
10089 047046  005201                          INC     R1                      ;
10090 047050  005204                          INC     R4                      ;
10091 047052  005204                          INC     R4                      ;
10092 047054  012737  000027  172516          MOV     #27,@#172516            ;INIT MMR3
10093 047062  012437  177776                  MOV     (R4)+,@#177776          ;INIT PSW
10094 047066  012746  052525                  MOV     #52525,-(SP)            ;PUSH DATA ONTO STACK
```

MEMORY MANAGEMENT TESTS

```
10095 047072  012737  000001  177572          MOV     #1,@#177572             ;TURN MMU ON
10096 047100  106631                           MTPD    @(R1)+                  ;WRITE DATA TO PHYSICAL ADDRESS
10097 047102  005037  177572                   CLR     @#177572                ;TURN MMU OFF
10098 047106  012737  000020  172516          MOV     #20,@#172516            ;INIT MMR3
10099 047114  004767  000040                   JSR     PC,T14                  ;CHECK IF DATA AT PHYSICAL ADDRESS IS EQUAL
10100                                                                          ;TO EXPECTED AND IF NOT DETERMINE IF IT
10101                                                                          ;IS AN ADDER ERROR OR A MEMORY ERROR
10102 047120  005037  172516                   CLR     @#172516                ;INIT MMR3 FOR 18 BIT MODE
10103 047124  004767  000004                   JSR     PC,TS14                 ;WRITE DATA TO PHYSICAL ADDRESS AND THEN
10104                                                                          ;CHECK IF DATA AT PHYSICAL ADDRESS IS
10105                                                                          ;EQUAL TO EXPECTED AND IF NOT DETERMINE IF
10106                                                                          ;IT IS AN ADDER ERROR OR A MEMORY ERROR
10107
10108 047130  000167  000376                   JMP     T14FIN
10109                                  ;
10110                                  ;ROUTINE TO WRITE DATA TO PHYSICAL ADDRESS AND TO CHECK IF DATA AT
10111                                  ;PHYSICAL ADDRESS IS EQUAL TO EXPECTED AND IF NOT DETERMINE IF IT IS
10112                                  ;AN ADDER ERROR OR A MEMORY ERROR
10113                                  ;
10114 047134  012437  177776          TS14:   MOV     (R4)+,@#177776          ;INIT PSW
10115 047140  012737  000001  177572          MOV     #1,@#177572             ;TURN MMU ON
10116 047146  012746  052525                   MOV     #52525,-(SP)            ;WRITE DATA ONTO STACK
10117 047152  006631                           MTPI    @(R1)+                  ;WRITE DATA TO PHYSICAL ADDRESS VIA STACK
10118 047154  005037  177572                   CLR     @#177572                ;TURN MMU OFF
10119 047160  012737  010000  177776  T14:    MOV     #10000,@#177776         ;INIT PSW
10120 047166  012237  172246                   MOV     (R2)+,@#172246          ;INIT SIPAR3
10121 047172  012737  000001  177572          MOV     #1,@#177572             ;TURN MMU ON
10122 047200  006573  000000                   MFPI    @0(R3)                  ;DO RELOCATION
10123 047204  022726  052525                   CMP     #52525,(SP)+            ;IS DATA EQUAL TO EXPECTED
10124 047210  001410                           BEQ     2$                      ;YES GO ON
10125 047212  006573  000000                   MFPI    @0(R3)                  ;WHAT TYPE OF ERROR IS IT
10126 047216  022726  125252                   CMP     #125252,(SP)+           ;
10127 047222  001402                           BEQ     1$                      ;
10128 047224  104002                           ERROR   .+2                     ;MMU ERROR
10129                                                                          ;IT IS A MEMORY ERROR
10130 047226  000401                           BR      2$                      ;
10131 047230  104002                  1$:      ERROR   .+2                     ;MMU ERROR
10132                                                                          ;IT IS AN ADDER ERROR
10133 047232  005037  177572          2$:     CLR     @#177572                ;TURN MMU OFF
10134 047236  005203                           INC     R3                      ;
10135 047240  005203                           INC     R3                      ;
10136 047242  000207                           RTS     PC                      ;RETURN
10137                                  ;
10138                                  ;NON-EXISTANT MEMORY TRAP ROUTINE
10139                                  ;
10140 047244  005037  177572          NXMTRP: CLR     @#SR0                   ;TURN OFF MMU.
10141 047250  012716  047532                   MOV     #T14FIN,(SP)            ;SET UP STACK WITH RETURN ADDR.
10142 047254  013737  003012  000004          MOV     @#SLOC00,@#4            ;RESTORE LOC 4
10143 047262  013737  003014  000006          MOV     @#SLOC01,@#6            ;RESTORE LOC 6
10144 047270  005037  177766                   CLR     @#177766                ;CLEAR TIME OUT INDICATION FROM
10145                                                                          ;CPU ERROR REGISTER.
10146 047274  000006                           RTT                             ;RETURN FROM TRAP; GO TO NEXT TEST.
10147
10148                                  ;ADDER TEST PART B TABLES
10149                                  ;
10150 047276  177646                  PARAD2: .WORD   177646
10151 047300  177650                          .WORD   177650
```

MEMORY MANAGEMENT TESTS

```
10152 047302  177652                    .WORD   177652
10153 047304  172240                    .WORD   172240
10154 047306  177640                    .WORD   177640
10155 047310  177642                    .WORD   177642
10156 047312  172244                    .WORD   172244
10157 047314  177644                    .WORD   177644
10158 047316  172252                    .WORD   172252
10159 047320  172352                    .WORD   172352
10160 047322  177662                    .WORD   177662
10161 047324  172242                    .WORD   172242
10162 047326  000333                    .WORD   333
10163 047330  157700            PARVA2: .WORD   157700
10164 047332  137700                    .WORD   137700
10165 047334  077700                    .WORD   077700
10166 047336  176777                    .WORD   176777
10167 047340  007600                    .WORD   007600
10168 047342  167700                     WORD   167700
10169 047344  175700                    .WORD   175700
10170 047346  177425                    .WORD   177425
10171 047350  177220                    .WORD   177220
10172 047352  173700                    .WORD   173700
10173 047354  176700                    .WORD   176700
10174 047356  077400                    .WORD   077400
10175 047360  000333                    .WORD   333
10176 047362  070000            VIR2:   .WORD   070000
10177 047364  110000                    .WORD   110000
10178 047366  130000                    .WORD   130000
10179 047370  000000                    .WORD   000000
10180 047372  000000                    .WORD   000000
10181 047374  030000                    .WORD   030000
10182 047376  050000                    .WORD   050000
10183 047400  052524                    .WORD   052524
10184 047402  136000                    .WORD   136000
10185 047404  130000                    .WORD   130000
10186 047406  000111                    .WORD   111
10187 047410  030000                    .WORD   030000
10188 047412  030000                    .WORD   030000
10189 047414  030000            MODE2:  .WORD   030000
10190 047416  030000                    .WORD   030000
10191 047420  030000                    .WORD   030000
10192 047422  010000                    .WORD   010000
10193 047424  030000                    .WORD   030000
10194 047426  030000                    .WORD   030000
10195 047430  010000                    .WORD   010000
10196 047432  030000                    .WORD   030000
10197 047434  010000                    .WORD   010000
10198 047436  000000                    .WORD   000000
10199 047440  000111                    .WORD   111
10200 047442  030000                    .WORD   030000
10201 047444  010000                    .WORD   010000
10202 047446  160000            PARVA3: .WORD   160000
10203 047450  140000                    .WORD   140000
10204 047452  100000                    .WORD   100000
10205 047454  176770                    .WORD   176770
10206 047456  007600                    .WORD   007600
10207 047460  170000                    .WORD   170000
10208 047462  176000                    .WORD   176000
```

MEMORY MANAGEMENT TESTS

```
10209 047464  177552                          .WORD   177552
10210 047466  177400                          .WORD   177400
10211 047470  174000                          .WORD   174000
10212 047472  177000                          .WORD   177000
10213 047474  007500                          .WORD   007500
10214 047476  000333                          .WORD   333
10215 047500  060000                  VIR3:   .WORD   060000
10216 047502  060000                          .WORD   060000
10217 047504  060000                          .WORD   060000
10218 047506  060700                          .WORD   060700
10219 047510  060000                          .WORD   060000
10220 047512  060000                          .WORD   060000
10221 047514  060000                          .WORD   060000
10222 047516  060024                          .WORD   060024
10223 047520  060000                          .WORD   060000
10224 047522  060000                          .WORD   060000
10225 047524  060000                          .WORD   060000
10226 047526  060000                          .WORD   060000
10227 047530  000333                          .WORD   333
10228
10229
10230                                 ;
10231 047532                          T14FIN:
10244
10245                                 ;       TEST NON-EXISTANT MEMORY TRAP
10246                                 ;;*************************************************************
                                      ;WE ARE ASSUMING THAT THE NON-EXISTANT MEMORY TIME OUT
                                      ;FEATURE IS WORKING SINCE WE CAN'T GUARANTEE THAT
                                      ;THE SYSTEM BEING TESTED HAS A NON-EXISTANT MEMORY LOCATION.
                                      ;AT THIS TIME WE WILL ATTEMPT TO TEST THE NXM FUNCTION
10247 047532  004767  070014                  JSR     PC,MMU                  ;INIT THE MMU
10248 047536  012737  177400  172354          MOV     #177400,@#KIPAR6        ;SET KIPAR6 TO RELOCATE TO HIGHEST MEMORY
10249 047544  016767  130234  133240          MOV     4,SLOC00                ;SAVE VECTOR
10250 047552  016767  000026  130224          MOV     2$,4                    ;LOAD VEC WITH ADDR OF TRAP HANDLER
10251 047560  052767  000001  130004          BIS     #BIT00,SR0              ;TURN ON THE MMU
10252 047566  005067  130174                  CLR     CPEREG                  ;CLEAR THE CPU ERROR REGISTER
10253 047572  005067  130200                  CLR     PS                      ;CLEAR THE PSW
10254 047576  005737  157776                  TST     @#157776                ;ACCESS PHYSICAL ADDR 17757776
10255 047602  000415                  1$:     BR      NXMFIN                  ;IF IT DOESN'T TRAP WE'LL ASSUME
10256                                                                         ;THAT THIS IA A 4 MEGABYTE SYSTEM
10257                                                                         ;AND GO TO THE NEXT TEST
10258 047604  022767  000040  130154  2$:     CMP     #BIT05,CPEREG           ;IS CPU ERROR REGISTER CORRECT?
10259 047612  001401                          BEQ     3$                      ;
10260 047614  104002                          ERROR   +2              ;MMU ERROR
10261 047616  022726  047602          3$:     CMP     #1$,(SP)+               ;IS CONTENTS OF STACK CORRECT?
10262 047622  001401                          BEQ     4$                      ;
10263 047624  104002                          ERROR   +2              ;MMU ERROR
10264 047626  022726  000000          4$:     CMP     #0,(SP)+                ;IS CONTENTS OF STACK CORRECT?
10265 047632  001401                          BEQ     NXMFIN                  ;
10266 047634  104002                          ERROR   +2              ;MMU ERROR
10267 047636  005067  127730          NXMFIN: CLR     SR0                     ;TURN OFF THE MMU
10268 047642  005067  130120                  CLR     CPEREG                  ;CLEAR THE CPU ERROR REGISTER
10269 047646  016767  133140  130130          MOV     SLOC00,4                ;RESTORE THE VECTOR
10270
10272
10274 047654                          TSMM15:
10275                                 ;       PAGE WRITTEN BIT TEST
```

MEMORY MANAGEMENT TESTS

```
10276 047654  005037  177572                CLR     @#177572                ;MMU OFF
10277 047660  005067  133144                CLR     FLAG                    ;CLEAR MMU ABORT FLAG
10278 047664  004767  067662                JSR     PC,MMU                  ;INIT MMU
10279 047670  005037  177776                CLR     @#177776                ;INIT PSW
10280 047674  012704  172300                MOV     #172300,R4              ;SET POINTER TO KPDRS
10281 047700  004767  000114                JSR     PC,TS15                 ;DO RELOCATIONS AND TEST KIPDRS FOR
10282                                                                         ;PAGE WRITTEN BIT BEING SET AND IF
10283                                                                         ;NOT SET GO TO ERROR
10284 047704  004767  000160                JSR     PC,T15                  ;DO RELOCATIONS AND TEST KDPDRS FOR
10285                                                                         ;PAGE WRITTEN BIT BEING SET AND IF NOT
10286                                                                         ;SET GO TO ERROR
10287 047710  012737  050000  177776        MOV     #50000,@#177776         ;INIT PSW
10288 047716  012704  172200                MOV     #172200,R4              ;SET POINTER TO SPDRS
10289 047722  004767  000072                JSR     PC,TS15                 ;DO RELOCATIONS AND TEST SIPDRS FOR
10290                                                                         ;PAGE WRITTEN BIT BEING SET AND IF NOT
10291                                                                         ;SET GO TO ERROR
10292 047726  004767  000136                JSR     PC,T15                  ;DO RELOCATIONS AND TEST SDPDRS FOR
10293                                                                         ;PAGE WRITTEN BIT BEING SET AND IF NOT
10294                                                                         ;SET GO TO ERROR
10295 047732  005037  177776                CLR     @#177776                ;INIT PSW TO A KNOWN STATE
10296 047736  012737  170000  177776        MOV     #170000,@#177776        ;INIT PSW
10297 047744  012704  177600                MOV     #177600,R4              ;SET POINTER TO UPDRS
10298 047750  004767  000044                JSR     PC,TS15                 ;DO RELOCATIONS AND TEST UIPDRS FOR
10299                                                                         ;PAGE WRITTEN BIT BEING SET AND IF
10300                                                                         ;NOT SET GO TO ERROR
10301 047754  004767  000110                JSR     PC,T15                  ;DO RELOCATIONS AND TEST UDPDRS FOR
10302                                                                         ;PAGE WRITTEN BIT BEING SET AND IF NOT
10303                                                                         ;SET GO TO ERROR
10304 047760  005037  177776                CLR     @#177776                ;INIT PSW TO A KNOWN STATE
10305 047764  012704  172300                MOV     #172300,R4              ;SET POINTER TO KPDRS
10306 047770  004767  000152                JSR     PC,T15A                 ;EXPLICITLY WRITE TO KPDRS AND TEST
10307                                                                         ;FOR PAGE WRITTEN BIT BEING CLEARED
10308                                                                         ;AND IF NOT CLEARED GO TO ERROR
10309 047774  012704  172200                MOV     #172200,R4              ;SET POINTER TO SPDRS
10310 050000  004767  000142                JSR     PC,T15A                 ;EXPLICITLY WRITE TO SPDRS AND TEST
10311                                                                         ;FOR PAGE WRITTEN BIT BEING CLEARED
10312                                                                         ;AND IF NOT CLEARED GO TO ERROR
10313 050004  012704  177600                MOV     #177600,R4              ;SET POINTER TO UPDRS
10314 050010  004767  000132                JSR     PC,T15A                 ;EXPLICITLY WRITE TO UPDRS AND TEST
10315                                                                         ;FOR PAGE WRITTEN BIT BEING CLEARED
10316                                                                         ;AND IF NOT CLEARED GO TO ERROR
10317
10318 050014  000167  000154                JMP     T15FIN
10319                                 ;
10320                                 ;ROUTINE TO DO RELOCATIONS AND TEST IPDRS FOR PAGE WRITTEN BIT BEING
10321                                 ;SET AND IF NOT SET REPORT AN ERROR
10322                                 ;
10323 050020  005001                TS15:   CLR     R1                      ;SET POINTER TO VIRTUAL ADDRESS
10324 050022  012737  000020  172516        MOV     #20,@#172516            ;INIT MMR3
10325 050030  012737  000001  177572  1$:   MOV     #1,@#177572             ;TURN MMU ON
10326 050036  011111                        MOV     (R1),(R1)               ;DO A RELOCATION
10327 050040  005037  177572                CLR     @#177572                ;TURN MMU OFF
10328 050044  022427  077506                CMP     (R4)+,#77506            ;IS DATA EQUAL TO EXPECTED
10329 050050  001401                        BEQ     2$                      ;OK GO ON
10330 050052  104002                        ERROR   +2              ;MMU ERROR
                                                                    ;NO GO TO ERROR
10331
10332 050054  062701  020000          2$:   ADD     #20000,R1               ;POINT TO NEXT VIRTUAL ADDRESS
```

MEMORY MANAGEMENT TESTS

```
10333 050060  020127  160000                CMP     R1,#160000              ;ARE WE DONE
10334 050064  001361                        BNE     1$                      ;NO GO TO 1$
10335 050066  000207                        RTS     PC                      ;RETURN
10336                                 ;
10337                                 ;ROUTINE TO DO RELOCATIONS AND TEST DPDRS FOR PAGE WRITTEN BIT BEING SET
10338                                 ;AND IF NOT SET REPORT AN ERROR
10339                                 ;
10340 050070  005001                  T15:  CLR     R1                      ;SET POINTER TO VIRTUAL ADDRESS
10341 050072  062704  000002                ADD     #2,R4                   ;POINT TO FIRST DPDR
10342 050076  012737  000027  172516        MOV     #27,@#172516            ;INIT MMR3
10343 050104  012737  000001  177572  1$:   MOV     #1,@#177572             ;TURN MMU ON
10344 050112  011146                        MOV     (R1),-(SP)              ;PUSH DATA ONTO THE STACK
10345 050114  106611                        MTPD    (R1)                    ;DO A RELOCATION
10346 050116  005037  177572                CLR     @#177572                ;TURN MMU OFF
10347 050122  022427  077506                CMP     (R4)+,#77506            ;IS DATA EQUAL TO EXPECTED
10348 050126  001401                        BEQ     2$                      ;OK GO ON
10349 050130  104002                        ERROR   +2              ;MMU ERROR
10350                                                               ;NO GO TO ERROR
10351 050132  062701  020000          2$:   ADD     #20000,R1               ;POINT TO NEXT VIRTUAL ADDRESS
10352 050136  020127  160000                CMP     R1,#160000              ;ARE WE DONE
10353 050142  001360                        BNE     1$                      ;NO GO TO 1$
10354 050144  000207                        RTS     PC                      ;RETURN
10355                                 ;
10356                                 ;ROUTINE TO EXPLICITLY WRITE TO PDRS AND TEST PAGE WRITTEN BIT FOR BEING
10357                                 ;CLEARED AND IF NOT CLEARED REPORT AN ERROR
10358                                 ;
10359 050146  005002                  T15A: CLR     R2                      ;CLEAR COUNTER
10360 050150  011414                  1$:   MOV     (R4),(R4)               ;DO AN EXPLICIT WRITE TO PDR
10361 050152  022427  077406                CMP     (R4)+,#77406            ;IS DATA EQUAL TO EXPECTED
10362 050156  001401                        BEQ     2$                      ;OK GO ON
10363 050160  104002                        ERROR   +2              ;MMU ERROR
10364                                                               ;NO GO TO ERROR
10365 050162  005202                  2$:   INC     R2                      ;INCREMENT POINTER
10366 050164  020227  000020                CMP     R2,#20                  ;ARE WE DONE
10367 050170  001367                        BNE     1$                      ;NO GO TO 1$
10368 050172  000207                        RTS     PC                      ;RETURN
10369 050174                          T15FIN:
10370                                 ;
10373 050174                          TSMM16:
10374                                 ;       TEST CSM (CALL SUPERVISOR MODE)
10375 050174  005037  177572                CLR     @#177572                ;MMU OFF
10376 050200  005037  003030                CLR     @#FLAG                  ;CLEAR MMU ABORT FLAG
10377 050204  012704  050524                MOV     #TMM16E,R4              ;INIT R4
10378 050210  004767  067336                JSR     PC,MMU                  ;INIT MMU
10379 050214  012737  000037  172516        MOV     #37,@#172516            ;ENABLE CSM INSTRUCTION
10380 050222  005037  177776                CLR     @#177776                ;SET PS TO KER MODE
10381 050226  013746  000010                MOV     @#10,-(SP)              ;SAVE VECTORS
10382 050232  013746  000014                MOV     @#14,-(SP)              ;
10383 050236  013746  000016                MOV     @#16,-(SP)              ;
10384 050242  012737  050414  000010        MOV     #TMM16B,@#10            ;SETUP NEW VECTORS
10385 050250  012737  000137  000014        MOV     #137,@#14               ;
10386 050256  012737  050534  000016        MOV     #TMM16A,@#16            ;
10387 050264  007014                        .WORD   7014                    ; TEST INSTRUCTION
10388 050266  104002                        ERROR   +2              ;MMU ERROR
10389                                                               ;GO TO ERROR IF NOT TRAPPED
10390 050270  012737  050444  000010  TSM16A: MOV     #TMM16C,@#10            ;SETUP NEW VECTOR
10391 050276  012737  000027  172516        MOV     #27,@#172516            ;DISABLE CSM INSTRUCTION
```

MEMORY MANAGEMENT TESTS

```
10392 050304  012737  140000  177776          MOV     #140000,@#177776        ;SET PS TO USER MODE
10393 050312  007014                          .WORD   7014                    ; TEST INSTRUCTION
10394 050314  104002                          ERROR   .+2             ;MMU ERROR
10395                                                                 ;GO TO ERROR IF NOT TRAPPED
10396 050316  012737  050474  000010  TSM16B: MOV     #TMM16D,@#10            ;SETUP NEW VECTOR
10397 050324  012737  000027  172516          MOV     #27,@#172516            ;DISABLE CSM INSTRUCTION
10398 050332  005037  177776                  CLR     @#177776                ;SET PS TO KER MODE
10399 050336  007014                          .WORD   7014                    ; TEST INSTRUCTION
10400 050340  104002                          ERROR   .+2             ;MMU ERROR
10401                                                                 ;GO TO ERROR IF NOT TRAPPED
10402 050342  012737  000037  172516  TSM16C: MOV     #37,@#172516            ;ENACLE CSM INSTRUCTION
10403 050350  012737  040000  177776          MOV     #40000,@#177776         ;SET PS TO SUP MODE
10404 050356  012706  000700                  MOV     #700,R6                 ;INIT SUP SP
10405 050362  012737  140000  177776          MOV     #140000,@#177776        ;SET PS TO USER MODE
10406 050370  012706  000600                  MOV     #600,R6                 ;INIT USER SP
10407 050374  012737  000014  000010          MOV     #14,@#10                ;SETUP NEW VECTOR
10408 050402  000277                          SCC                             ;SET ALL CC BITS
10409 050404  007024                          .WORD   7024                    ; TEST INSTRUCTION
10410 050406  104002          TSM16D: ERROR   .+2             ;MMU ERROR
10411                                                                 ;GO TO ERROR IF NOT TRAPPED
10412 050410  000167  000504                  JMP     TM16A
10413                         ;
10414                         ;
10415 050414  042737  007777  177776  TMM16B: BIC     #7777,@#177776          ;CLEAR UNWANTED BITS
10416 050422  022737  000000  177776          CMP     #0,@#177776             ;IS PS CORRECT
10417 050430  001401                          BEQ     1$                      ;YES GO ON
10418 050432  104002                          ERROR   .+2             ;MMU ERROR
10419                                                                 ;NO GO TO ERROR
10420 050434  005726                  1$:     TST     (SP)+                   ;CLEAN UP STACK
10421 050436  005726                          TST     (SP)+                   ;
10422 050440  000167  177624                  JMP     TSM16A                  ;CONTINUE TESTING
10423 050444  042737  007777  177776  TMM16C: BIC     #7777,@#177776          ;CLEAR UNWANTED BITS
10424 050452  022737  030000  177776          CMP     #30000,@#177776         ;IS PS CORRECT
10425 050460  001401                          BEQ     1$                      ;YES GO ON
10426 050462  104002                          ERROR   .+2             ;MMU ERROR
10427                                                                 ;NO GO TO ERROR
10428 050464  005726                  1$:     TST     (SP)+                   ;CLEAN UP STACK
10429 050466  005726                          TST     (SP)+                   ;
10430 050470  000167  177622                  JMP     TSM16B                  ;CONTINUE TESTING
10431 050474  042737  007777  177776  TMM16D: BIC     #7777,@#177776          ;CLEAR UNWANTED BITS
10432 050502  022737  000000  177776          CMP     #0,@#177776             ;IS PS CORRECT
10433 050510  001401                          BEQ     1$                      ;YES GO ON
10434 050512  104002                          ERROR   .+2             ;MMU ERROR
10435                                                                 ;NO GO TO ERROR
10436 050514  005726                  1$:     TST     (SP)+                   ;CLEAN UP STACK
10437 050516  005726                          TST     (SP)+                   ;
10438 050520  000167  177616                  JMP     TSM16C                  ;CONTINUE TESTING
10439 050524  156430          TMM16E: .WORD   156430                  ; TEST LOCATION
10440 050526  104002          TMM16F: ERROR   .+2             ;MMU ERROR
10441                                                                 ;GO TO ERROR IF DIDN'T ABORT
10442 050530  000167  000364                  JMP     TM16A
10443 050534  022737  070017  177776  TMM16A: CMP     #70017,@#177776         ;IS PS CORRECT
10444 050542  001401                          BEQ     1$                      ;YES GO ON
10445 050544  104002                          ERROR   .+2             ;MMU ERROR
10446                                                                 ;NO GO TO ERROR
10447 050546  020627  000572          1$:     CMP     R6,#572                 ;IS SP CORRECT
10448 050552  001401                          BEQ     2$                      ;YES GO ON
```

F15

MEMORY MANAGEMENT TESTS

```
10449 050554  104002                          ERROR   +2              ;MMU ERROR
10450                                                                 ;NO GO TO ERROR
10451 050556  020427  050526          2$:     CMP     R4,#TMM16E+2            ;IS R4 CORRECT
10452 050562  001401                          BEQ     3$                      ;YES GO ON
10453 050564  104002                          ERROR   +2              ;MMU ERROR
10454                                                                 ;NO GO TO ERROR
10455 050566  023727  050524  156430  3$:     CMP     @#TMM16E,#156430        ;IS TEST LOCATION OK
10456 050574  001401                          BEQ     4$                      ;YES GO ON
10457 050576  104002                          ERROR   +2              ;MMU ERROR
10458                                                                 ;NO GO TO ERROR
10459 050600  022627  156430          4$:     CMP     (SP)+,#156430           ;IS STACK CORRECT
10460 050604  001401                          BEQ     5$                      ;YES GO ON
10461 050606  104002                          ERROR   +2              ;MMU ERROR
10462                                                                 ;NO GO TO ERROR
10463 050610  022627  050406          5$:     CMP     (SP)+,#TSM16D           ;IS STACK CORRECT
10464 050614  001401                          BEQ     6$                      ;YES GO ON
10465 050616  104002                          ERROR   +2              ;MMU ERROR
10466                                                                 ;NO GO TO ERROR
10467 050620  022627  140000          6$:     CMP     (SP)+,#140000           ;IS STACK CORRECT
10468 050624  001401                          BEQ     7$                      ;YES GO ON
10469 050626  104002                          ERROR   +2              ;MMU ERROR
10470                                                                 ;NO GO TO ERROR
10471 050630  012706  000700          7$:     MOV     #700,R6                 ;RESTORE SUP SP
10472 050634  012737  140000  177776          MOV     #140000,@#177776        ;SET PS TO USER MODE
10473 050642  020627  000600                  CMP     R6,#600                 ;IS USER SP CORRECT
10474 050646  001401                          BEQ     8$                      ;YES GO ON
10475 050650  104002                          ERROR   +2              ;MMU ERROR
10476                                                                 ;NO GO TO ERROR
10477 050652  012767  077400  121320  8$:     MOV     #77400,SIPDR0           ;SETUP SIPDR0 TO ABORT
10478 050660  012737  050526  000016          MOV     #TMM16F,@#16            ;SETUP VECTOR
10479 050666  012737  000001  003030          MOV     #1,@#FLAG               ;SETUP FLAG FOR AN ABORT
10480 050674  012737  000001  177572          MOV     #1,@#177572             ;TURN MMU ON
10481 050702  010701                          MOV     R7,R1                   ;SAVE OLD PC
10482 050704  007014                          .WORD   7014                    ; TEST INSTRUCTION
10483 050706  022737  000000  003030          CMP     #0,@#FLAG               ;DID AN ABORT OCCUR
10484 050714  001401                          BEQ     9$                      ;YES GO ON
10485 050716  104002                          ERROR   +2              ;MMU ERROR
10486                                                                 ;NO GO TO ERROR
10487 050720  023701  003046          9$:     CMP     @#SAVMR2,R1             ;IS MMR2 CORRECT
10488 050724  001401                          BEQ     10$                     ;YES GO ON
10489 050726  104002                          ERROR   +2              ;MMU ERROR
10490                                                                 ;NO GO TO ERROR
10491 050730  023727  003042  100041  10$:     CMP     @#SAVMR0,#100041        ;IS MMR0 CORRECT
10492 050736  001401                          BEQ     11$                     ;YES GO ON
10493 050740  104002                          ERROR   +2              ;MMU ERROR
10494                                                                 ;NO GO TO ERROR
10495 050742  012737  000037  172516  11$:     MOV     #37,@#172516            ;ENABLE CSM
10496 050750  012737  040000  177776          MOV     #40000,@#177776         ;SET PSW TO SUP
10497 050756  012706  000700                  MOV     #700,R6                 ;SETUP SUP SP
10498 050762  012737  140000  177776          MOV     #140000,@#177776        ;SET PSW TO USE
10499 050770  012706  000600                  MOV     #600,R6                 ;SETUP USE SP
10500 050774  012737  000014  000010          MOV     #14,@#10                ;SETUP NEW VECTOR
10501 051002  012737  051024  000016          MOV     #TS16,@#16              ;SETUP NEW VECTOR
10502 051010  000277                          SCC                             ;SET ALL CC BITS
10503 051012  007027                          .WORD   7027                    ;TEST INSTRUCTION
10504 051014  045712                          .WORD   45712
10505 051016  104002          TS16A:  ERROR   +2              ;MMU ERROR
```

MEMORY MANAGEMENT TESTS

```
10506                                                                   ;GO TO ERROR IF DIDN T TRAP
10507 051020  000167  000074                    JMP     TM16A
10508 051024  022737  070017  177776  TS16:     CMP     #70017,@#177776       ;IS PSW CORRECT
10509 051032  001401                            BEQ     200$                  ;YES GO ON
10510 051034  104002                            ERROR   +2                    ;MMU ERROR
10511                                                                   ;NO GO TO ERROR
10512 051036  020627  000572           200$:    CMP     R6,#572               ;IS SP CORRECT
10513 051042  001401                            BEQ     201$                  ;YES GO ON
10514 051044  104002                            ERROR   +2                    ;MMU ERROR
10515                                                                   ;NO GO TO ERROR
10516 051046  022627  045712           201$:    CMP     (SP)+,#45712          ;IS STACK CORRECT
10517 051052  001401                            BEQ     202$                  ;YES GO ON
10518 051054  104002                            ERROR   +2                    ;MMU ERROR
10519                                                                   ;NO GO TO ERROP
10520 051056  022627  051016           202$:    CMP     (SP)+,#TS16A          ;IS STACK CORRECT
10521 051062  001401                            BEQ     203$                  ;YES GO ON
10522 051064  104002                            ERROR   +2                    ;MMU ERROR
10523                                                                   ;NO GO TO ERROR
10524 051066  022627  140000           203$:    CMP     (SP)+,#140000         ;IS STACK CORRECT
10525 051072  001401                            BEQ     204$                  ;YES GO ON
10526 051074  104002                            ERROR   +2                    ;MMU ERROR
10527                                                                   ;NO GO TO ERROR
10528 051076  012706  000700           204$:    MOV     #700,R6               ;RESTORE SUP SP
10529 051102  012737  140000  177776             MOV     #140000,@#177776      ;SET PSW TO USER MODE
10530 051110  020627  000600                     CMP     R6,#600               ;IS USER SP CORRECT
10531 051114  001401                            BEQ     TM16A                 ;YES GO ON
10532 051116  104002                            ERROR   +2                    ;MMU ERROR
10533                                                                   ;NO GO TO ERROR
10534 051120  005037  177776           TM16A:   CLR     @#177776              ;SET PS TO KER MODE
10535 051124  012637  000016                     MOV     (SP)+,@#16            ;RESTORE VECTORS
10536 051130  012637  000014                     MOV     (SP)+,@#14            ;
10537 051134  012637  000010                     MOV     (SP)+,@#10            ;
10538
10542                                   ;*****************************************************************************
10543                                   .SBTTL  FLOATING POINT TESTS
10544                                   ;*****************************************************************************
10545                                   ;                       BEGIN FLOATING POINT TESTING
10546                                   ;*****************************************************************************
10547                                   ;*****************************************************************************
10560 051140                            MBT1:
10561
10562                                   ;       FPP REGISTER BIT TESTS
10563                                   ;;*************************************************************
                                        ;R5=FPP POINTER
                                        ;R1=TEMPORARY COUNTER
                                        ;R2=POINTER TO EXPECTED DATA
                                        ;R3=POINTER TO RECEIVED DATA
                                        ;R4=ODD/EVEN COUNTER
10564 051140  170011                            SETD
10565 051142  005005                    MBT2:   CLR     R5                    ;SETUP FPP ACC POINTER
10566 051144  012702  003100                     MOV     #BTEXP,R2             ;POINT TO TEST DATA
10567 051150  012703  003110                     MOV     #BTRES,R3             ;POINT TO RECEIVED DATA
10568 051154  170400                    MBT2A:  CLRD    AC0                   ;SETUP FPP REGISTER VALUES
10569 051156  174012                            STD     AC0,(R2)              ;CLEAR EXPECTED VALUE
10570 051160  005004                            CLR     R4
10571 051162  170400                    BTGO:   CLRD    AC0                   ;SETUP FPP REGISTER VALUES
10572 051164  170401                            CLRD    AC1
```

H13

FLOATING POINT TESTS

```
10573 051166  170402                          CLRD    AC2
10574 051170  170403                          CLRD    AC3
10575 051172  170404                          CLRD    AC4
10576 051174  170405                          CLRD    AC5
10577
10578 051176  010501                          MOV     R5,R1           ;GET FPP AC NUMBER INTO R1
10579 051200  070127  000014                  MUL     #14,R1          ;ALLOW 10 LOCATIONS FOR OPERATION
10580 051204  062701  051212                  ADD     #MAC0,R1        ;SETUP JMP LOCATION
10581 051210  000111                          JMP     (R1)
10582 051212  172467  131662          MAC0:   LDD     BTEXP,AC0       ;LOAD TEST DATA INTO TEST REGISTER
10583 051216  174067  131666          MAC0A:  STD     AC0,BTRES       ;SAVE TEST RESULT
10584 051222  000167  000074                  JMP     MACE            ;GET OUT
10585 051226  172567  131646          MAC1:   LDD     BTEXP,AC1       ;LOAD TEST DATA INTO TEST REGISTER
10586 051232  174167  131652                  STD     AC1,BTRES       ;SAVE TEST RESULT
10587 051236  000167  000060                  JMP     MACE            ;GET OUT
10588 051242  172667  131632          MAC2:   LDD     BTEXP,AC2       ;LOAD TEST DATA INTO TEST REGISTER
10589 051246  174267  131636                  STD     AC2,BTRES       ;SAVE TEST RESULT
10590 051252  000167  000044                  JMP     MACE            ;GET OUT
10591 051256  172767  131616          MAC3:   LDD     BTEXP,AC3       ;LOAD TEST DATA INTO TEST REGISTER
10592 051262  174367  131622                  STD     AC3,BTRES       ;SAVE TEST RESULT
10593 051266  000167  000030                  JMP     MACE            ;GET OUT
10594 051272  172467  131602          MAC4:   LDD     BTEXP,AC0
10595 051276  174004                          STD     AC0,AC4         ;LOAD TEST DATA INTO TEST REGISTER
10596 051300  172404                          LDD     AC4,AC0         ;SAVE TEST RESULT
10597 051302  000167  177710                  JMP     MAC0A           ;GET OUT
10598 051306  172467  131566          MAC5:   LDD     BTEXP,AC0       ;LOAD TEST DATA INTO TEST XFER REGISTER
10599 051312  174005                          STD     AC0,AC5         ;LOAD TEST REGISTER
10600 051314  172405                          LDD     AC5,AC0         ;STORE RESULT INTO XFER FPP REGISTER
10601 051316  000167  177674                  JMP     MAC0A           ;GET OUT
10602 051322  026767  131552  131560  MACE:   CMP     BTEXP,BTRES
10603 051330  001014                          BNE     BTER            ;BRANCH IF REGISTER ERROR
10604 051332  026767  131544  131552          CMP     BTEXP+2,BTRES+2
10605 051340  001010                          BNE     BTER
10606 051342  026767  131536  131544          CMP     BTEXP+4,BTRES+4
10607 051350  001004                          BNE     BTER
10608 051352  026767  131530  131536          CMP     BTEXP+6,BTRES+6
10609 051360  001403                          BEQ     MBT8                    ;GOOD RESULT
10610 051362  004737  140132          BTER:   CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
10611 051366  104003                          ERROR   +3              ;FPP ERROR
10612                                                                 ;FPP AC LOADED INCORRECTLY
10613                                 ;NOW VERIFY THE OTHER REGISTERS REMAINED ZERO
10614 051370                          MBT8:
10615 051370  005001                          CLR     R1                      ;CLEAR TEMPORARY COUNTER
10616 051372  005705                          TST     R5                      ;SEE IF R0 UNDER TEST
10617 051374  001413                          BEQ     MBT8A                   ;BRANCH IF TEST ING R0
10618 051376  020527  000004                  CMP     R5,#4                   ;SEE IF TESTING FPP REGISTER >R4
10619 051402  100010                          BPL     MBT8A                   ;SKIP R0 TESTING
10620 051404  174067  131500                  STD     AC0,BTRES               ;SAVE AC TEST RESULT
10621 051410  004767  000246                  JSR     R7,BTTST                ;VERIFY THAT CONTENTS REMAINED ZERO
10622 051414  001403                          BEQ     MBT8A                   ;BRANCH IF EXPECTED RESULT
10623 051416  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
10624 051422  104003                          ERROR   +3              ;FPP ERROR
10625                                                                 ;BAD AC0
10626 051424  020527  000001          MBT8A:  CMP     R5,#1                   ;SEE IF R1 UNDER TEST
10627 051430  001410                          BEQ     MBT8B                   ;BRANCH IF R1 UNDER TEST
10628 051432  174167  131452                  STD     AC1,BTRES               ;SAVE AC TEST RESULT
10629 051436  004767  000220                  JSR     R7,BTTST                ;VERIFY THAT CONTENTS REMAINED ZERO
```

FLOATING POINT TESTS

```
10630 051442  001403                    BEQ     MBT8B           ;BRANCH IF GOOD
10631 051444  004737  140132              CALL    @#DETFPA        ;DETERMINE FLOATING POINT FAULT. $$$
10632 051450  104003                      ERROR   .+3             ;FPP ERROR
10633                                                             ;BAD AC1
10634 051452  020527  000002  MBT8B:     CMP     R5,#2           ;SEE IF TESTING FPP REGISTER AC2
10635 051456  001410                      BEQ     MBT8C           ;BRANCH IF R2 UNDER TEST
10636 051460  174267  131424              STD     AC2,BTRES       ;SAVE AC TEST RESULT
10637 051464  004767  000172              JSR     R7,BTTST        ;VERIFY THAT CONTENTS REMAINED ZERO
10638 051470  001403                      BEQ     MBT8C           ;BRANCH IF GOOD
10639 051472  004737  140132              CALL    @#DETFPA        ;DETERMINE FLOATING POINT FAULT. $$$
10640 051476  104003                      ERROR   .+3             ;FPP ERROR
10641                                                             ;BAD AC2
10642 051500  020527  000003  MBT8C:     CMP     R5,#3           ;SEE IF R3 UNDER TEST
10643 051504  001410                      BEQ     MBT8D           ;BRANCH IF R3 UNDER TEST
10644 051506  174367  131376              STD     AC3,BTRES       ;SAVE AC TEST RESULT
10645 051512  004767  000144              JSR     R7,BTTST        ;VERIFY THAT CONTENTS REMAINED ZERO
10646 051516  001403                      BEQ     MBT8D           ;BRANCH IF GOOD
10647 051520  004737  140132              CALL    @#DETFPA        ;DETERMINE FLOATING POINT FAULT. $$$
10648 051524  104003                      ERROR   .+3             ;FPP ERROR
10649                                                             ;BAD AC3
10650 051526  020527  000004  MBT8D:     CMP     R5,#4           ;SEE IF R4 UNDER TEST
10651 051532  001411                      BEQ     MBT8E           ;BRANCH IF R4 UNDER TEST
10652 051534  172404                      LDD     AC4,AC0         ;MOVE REGISTER CONTENT
10653 051536  174067  131346              STD     AC0,BTRES       ;SAVE AC TEST RESULT
10654 051542  004767  000114              JSR     R7,BTTST        ;VERIFY THAT CONTENTS REMAINED ZERO
10655 051546  001403                      BEQ     MBT8E           ;BRANCH IF GOOD
10656 051550  004737  140132              CALL    @#DETFPA        ;DETERMINE FLOATING POINT FAULT. $$$
10657 051554  104003                      ERROR   .+3             ;FPP ERROR
10658                                                             ;BAD AC4
10659 051556  020527  000005  MBT8E:     CMP     R5,#5           ;SEE IF R0 UNDER TEST
10660 051562  001411                      BEQ     MBT8F           ;BRANCH IF R0 UNDER TEST
10661 051564  172405                      LDD     AC5,AC0         ;MOVE REGISTER CONTENTS
10662 051566  174067  131316              STD     AC0,BTRES       ;SAVE AC TEST RESULT
10663 051572  004767  000064              JSR     R7,BTTST        ;VERIFY THAT CONTENTS REMAINED ZERO
10664 051576  001403                      BEQ     MBT8F           ;BRANCH IF GOOD
10665 051600  004737  140132              CALL    @#DETFPA        ;DETERMINE FLOATING POINT FAULT. $$$
10666 051604  104003                      ERROR   .+3             ;FPP ERROR
10667                                                             ;BAD AC5
10668 051606  005204  MBT8F:     INC     R4              ;INCREMENT PATTERN COUNTER
10669 051610  000241                      CLC
10670 051612  042704  177776              BIC     #177776,R4      ; TEST FOR ODD /EVEN
10671 051616  001401                      BEQ     MBT8FG          ;BRANCH IF EVEN
10672 051620  000261                      SEC                     ;SET CARRY FOR TEST PATTERN SHIFT
10673 051622  006112  MBT8FG:    ROL     (R2)            ;ROTATE LSW OF TEST PATTERN
10674 051624  006162  000002              ROL     2(R2)           ;ROTATE 2 WORD OF TEST PATTERN
10675 051630  006162  000004              ROL     4(R2)           ;ROTATE 3 WORD OF TEST PATTERN
10676 051634  006162  000006              ROL     6(R2)           ;ROTATE 4 WORD OF TEST PATTERN
10677 051640  103402                      BCS     MBT8I           ;JUMP IF THROUGH WITH TEST PATTERN
10678 051642  000167  177314              JMP     BTGO            ;CONTINUE WITH NEW TEST PATTERN
10679
10680 051646  005205  MBT8I:     INC     R5              ;GO TO NEXT REGISTER TEST
10681 051650  020527  000006              CMP     R5,#6           ;SEE IF THROUGH TESTING
10682 051654  100016                      BPL     MBTE            ;JUMP IF THROUGH
10683 051656  000167  177272              JMP     MBT2A           ;CONTINUE TESTING WITH NEW PATTERN
10684                         ;
10685                         ;
10686 051662  005767  131222  BTTST:     TST     BTRES           ;VERIFY CONTENTS AS ZERO
```

FLOATING POINT TESTS

```
10687 051666  001010                          BNE     BTTSTE          ;EXIT IF NOT ZERO
10688 051670  005767  131216                  TST     BTRES+2         ;VERIFY CONTENTS AS ZERO
10689 051674  001005                          BNE     BTTSTE          ;EXIT IF NOT ZERO
10690 051676  005767  131212                  TST     BTRES+4         ;VERIFY CONTENTS AS ZERO
10691 051702  001002                          BNE     BTTSTE          ;EXIT IF NOT ZERO
10692 051704  005767  131206                  TST     BTRES+6         ;VERIFY CONTENTS AS ZERO
10693 051710  000207                  BTTSTE: RTS     R7              ;GO BACK TO CALLING ROUTIN
10694
10695
10696                                         ;
10697                                         ;
10698 051712                          MBTE:
10699
10701                                         ;
10712 051712                          MFACU:
10713
10714                                 ;       TEST UNIQUENESS OF FPP ACCUMULATORS
10715                                 ;;*****************************************************************
                                      ;THIS TEST LOADS UNIQUE PATTERNS INTO EACH ACCUMULATOR SIMULTANEOUSLY.
                                      ;R2=POINTER TO EXPECTED DATA
                                      ;R3=POINTER TO RECEIVED DATA
10716                                           ;
10717 051712  170011                          SETD
10718 051714  005000                  MFA:    CLR     R0                      ;SETUP FPP ACC POINTER
10719 051716  005004                          CLR     R4
10720 051720  012702  003100                  MOV     #BTEXP,R2               ;POINT TO TEST DATA
10721 051724  012703  003110                  MOV     #BTRES,R3               ;POINT TO RECEIVED DATA
10722 051730  012767  000051  131142          MOV     #51,BTEXP               ;SETUP EXPECTED DATA
10723 051736  012767  000052  131136          MOV     #52,BTEXP+2             ;
10724 051744  012767  000053  131132          MOV     #53,BTEXP+4             ;
10725 051752  012767  000054  131126          MOV     #54,BTEXP+6             ;
10726 051760  172467  131114                  LDD     BTEXP,AC0               ;MOVE DATA TEMPORARILY
10727 051764  174005                          STD     AC0,AC5                 ;PUT DATA INTO TEST REGISTER
10728 051766  004567  000240                  JSR     R5,SUBT                 ;SUBTRACT TEN FROM EACH EXPECTED DATA
10729 051772  172467  131102                  LDD     BTEXP,AC0               ;MOVE DATA TEMPORARILY
10730 051776  174004                          STD     AC0,AC4                 ;MOVE DATA INTO TEST REGISTER
10731 052000  004567  000226                  JSR     R5,SUBT                 ;SUBTRACT 10 FROM TEST DATA WORDS
10732 052004  172767  131070                  LDD     BTEXP,AC3               ;STORE INTO TEST REGISTER
10733 052010  004567  000216                  JSR     R5,SUBT                 ;GET NEXT SET OF UNIQUE DATA WORDS
10734 052014  172667  131060                  LDD     BTEXP,AC2               ;STORE INTO TEST REGISTER
10735 052020  004567  000206                  JSR     R5,SUBT                 ;GET NEXT SET OF TEST DATAS
10736 052024  172567  131050                  LDD     BTEXP,AC1               ;LOAD TEST REGISTER
10737 052030  004567  000176                  JSR     R5,SUBT                 ;GET NEXT SET OF TEST WORDS
10738 052034  172467  131040                  LDD     BTEXP,AC0               ;LOAD FINAL TEST REGISTER
10739                                                                         ;ALL REGISTER CONTAIN UNIQUE TEST WORDS
10740 052040  174067  131044                  STD     AC0,BTRES               ;STORE AC0,RESULT
10741 052044  004567  000246                  JSR     R5,BFA                  ;CHECK RESULT
10742 052050  001403                          BEQ     BFAC1                   ;BRANCH IF GOOD
10743 052052  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
10744 052056  104003                          ERROR   +3              ;FPP ERROR
10745                                                                 ;BAD AC0
10746 052060  004567  000200          BFAC1:  JSR     R5,ADDT                 ;UPDATE EXPECTED RESULT
10747 052064  174167  131020                  STD     AC1,BTRES               ;STORE AC1 RESULT
10748 052070  004567  000222                  JSR     R5,BFA                  ;CHECK RESULT
10749 052074  001403                          BEQ     BFAC2                   ;BRANCH IF GOOD
10750 052076  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
10751 052102  104003                          ERROR   +3              ;FPP ERROR
```

FLOATING POINT TESTS

```
10752
10753 052104 004567 000154          BFAC2:  JSR     R5,ADDT          ;BAD RESULT AC1
10754 052110 174267 130774                  STD     AC2,BTRES                ;UPDATE EXPECTED RESULT
10755 052114 004567 000176                  JSR     R5,BFA                   ;STORE AC2 RESULT
10756 052120 001403                         BEQ     BFAC3                    ;CHECK RESULT
10757 052122 004737 140132                  CALL    @#DETFPA                 ;BRANCH IF GOOD
10758 052126 104003                         ERROR   .+3                      ;DETERMINE FLOATING POINT FAULT. $$$
10759                                                                        ;FPP ERROR
10760 052130 004567 000130          BFAC3:  JSR     R5,ADDT          ;BAD AC2 RESLUT
10761 052134 174267 130750                  STD     AC3,BTRES                ;UPDATE EXPECTED RESULT
10762 052140 004567 000152                  JSR     R5,BFA                   ;SAVE TEST RESULT
10763 052144 001403                         BEQ     BFAC4                    ;CHECK RESULT
10764 052146 004737 140132                  CALL    @#DETFPA                 ;BRANCH IF GOOD
10765 052152 104003                         ERROR   .+3                      ;DETERMINE FLOATING POINT FAULT. $$$
10766                                                                        ;FPP ERROR
10767 052154 004567 000104          BFAC4:  JSR     R5,ADDT          ;BAD AC3 RESULT
10768 052160 172704                         LDD     AC4,AC3                  ;UPDATE EXPECTED RESULT
10769 052162 174367 130722                  STD     AC3,BTRES                ;SAVE TEMPORARY
10770 052166 004567 000124                  JSR     R5,BFA                   ;STORE AC4 RESULT
10771 052172 001403                         BEQ     BFAC5                    ;CHECK RESULT
10772 052174 004737 140132                  CALL    @#DETFPA                 ;BRANCH IF GOOD
10773 052200 104003                         ERROR   .+3                      ;DETERMINE FLOATING POINT FAULT. $$$
10774                                                                        ;FPP ERROR
10775 052202 004567 000056          BFAC5:  JSR     R5,ADDT          ;BAD AC4 RESULT
10776 052206 172605                         LDD     AC5,AC2                  ;UPDATE EXPECTED RESULT
10777 052210 174267 130674                  STD     AC2,BTRES                ;SAVE TEMPORARY COPY
10778 052214 004567 000076                  JSR     R5,BFA                   ;MOVE AC5 RESULT
10779 052220 001456                         BEQ     BFAE                     ;CHECK RESULT
10780 052222 004737 140132                  CALL    @#DETFPA                 ;BRANCH IF GOOD
10781 052226 104003                         ERROR   .+3                      ;DETERMINE FLOATING POINT FAULT. $$$
10782                                                                        ;FPP ERROR
10783 052230 000452                         BR      BFAE             ;BAD AC5 RESULT
10784                                                                        ;EXIT MODULE
10785 052232 162767 000010 130640   SUBT:   SUB     #10,BTEXP                ·UPDATE EXPECTED CONTENTS
10786 052240 162767 000010 130634           SUB     #10,BTEXP+2              ;UPDATE EXPECTED CONTENTS
10787 052246 162767 000010 130630           SUB     #10,BTEXP+4              ;UPDATE EXPECTED CONTENTS
10788 052254 162767 000010 130624           SUB     #10,BTEXP+6              ;UPDATE EXPECTED CONTENTS
10789 052262 000205                         RTS     R5               ;
10790 052264 062767 000010 130606   ADDT:   ADD     #10,BTEXP                ;UPDATE EXPECTED CONTENTS
10791 052272 062767 000010 130602           ADD     #10,BTEXP+2              ;UPDATE EXPECTED CONTENTS
10792 052300 062767 000010 130576           ADD     #10,BTEXP+4              ;UPDATE EXPECTED CONTENTS
10793 052306 062767 000010 130572           ADD     #10,BTEXP+6              ;UPDATE EXPECTED CONTENTS
10794 052314 000205                         RTS     R5               ;
10795
10796 052316 026767 130556 130564   BFA:    CMP     BTEXP,BTRES      ;VERIFY CONTENTS
10797 052324 001013                         BNE     BFB              ;EXIT IF NOT ZERO
10798 052326 026767 130550 130556           CMP     BTEXP+2,BTRES+2  ;VERIFY CONTENTS
10799 052334 001007                         BNE     BFB              ;EXIT IF NOT ZERO
10800 052336 026767 130542 130550           CMP     BTEXP+4,BTRES+4  ;VERIFY CONTENTS
10801 052344 001003                         BNE     BFB              ;EXIT IF NOT ZERO
10802 052346 026767 130534 130542           CMP     BTEXP+6,BTRES+6  ;VERIFY CONTENTS
10803 052354 000205                 BFB:    RTS     R5               ;GO BACK TO CALLING ROUTINE
10804
10805
10806 052356                        BFAE:
10807
10808
```

FLOATING POINT TESTS

```
10809
10812 052356                                TSFP1:
10813                                       ;       TEST LDFPS AND STFPS MODE 0
10814 052356  005037  140054                        CLR     @#TRPFLG                ;CLEAR TRAP FLAG
10815 052362  012704  147757                        MOV     #147757,R4              ;SETUP DATA TO BE LOADED
10816 052366  004767  000032                        JSR     PC,LOST                 ;LOAD AND STORE FPS WITH DATA
10817 052372  012704  105252                        MOV     #105252,R4              ;SETUP DATA TO BE LOADED
10818 052376  004767  000022                        JSR     PC,LOST                 ;LOAD AND STORE FPS WITH DATA
10819 052402  012704  042505                        MOV     #42505,R4               ;SETUP DATA TO BE LOADED
10820 052406  004767  000012                        JSR     PC,LOST                 ;LOAD AND STORE FPS WITH DATA
10821 052412  005004                                CLR     R4                      ;SETUP DATA TO BE LOADED
10822 052414  004767  000004                        JSR     PC,LOST                 ;LOAD AND STORE FPS WITH DATA
10823
10824                                       ;
10825 052420  000167  000020                        JMP     FIN1
10826                                       ;
10827 052424  170104                        LOST:   LDFPS   R4                      ;LOAD FPS WITH DATA
10828 052426  170201                                STFPS   R1                      ;LOAD R1 WITH (FPS)
10829 052430  020401                                CMP     R4,R1                   ;DID THE INSTRUCTIONS WORK
10830 052432  001403                                BEQ     1$                      ;YES GO ON
10831 052434  004737  140132                        CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT  $$$
10832 052440  104003                                ERROR   +3              ;FPP ERROR
10833                                                                       ;NO GO TO ERROR
10834 052442  000207                        1$:     RTS     PC                      ;RETURN
10835 052444  000240                        FIN1:   NOP
10836                                       ;
10839 052446                                TSFP2:
10840                                       ;       TEST CFCC
10841 052446  005037  140054                        CLR     @#TRPFLG                ;CLEAR TRAP FLAG
10842 052452  012704  000017                        MOV     #17,R4                  ;SETUP DATA TO BE LOADED
10843 052456  004767  000032                        JSR     PC,TSF2                 ;LOAD FPS AND COPY CONDITION CODES TO PS
10844 052462  012704  000012                        MOV     #12,R4                  ;SETUP DATA TO BE LOADED
10845 052466  004767  000022                        JSR     PC,TSF2                 ;LOAD FPS AND COPY CONDITION CODES TO PS
10846 052472  012704  000005                        MOV     #5,R4                   ;SETUP DATA TO BE LOADED
10847 052476  004767  000012                        JSR     PC,TSF2                 ;LOAD FPS AND COPY CONDITION CODES TO PS
10848 052502  005004                                CLR     R4                      ;SETUP DATA TO BE LOADED
10849 052504  004767  000004                        JSR     PC,TSF2                 ;LOAD FPS AND COPY CONDITION CODES TO PS
10850                                       ;
10851 052510  000167  000030                        JMP     FIN2
10852                                       ;
10853 052514  170104                        TSF2:   LDFPS   R4                      ;LOAD FPS
10854 052516  170000                                CFCC                            ;COPY CONDITION CODES TO PS
10855 052520  013701  177776                        MOV     @#177776,R1             ;SAVE PS TO R1
10856 052524  042701  177760                        BIC     #177760,R1              ;MASK OUT UNWANTED BITS
10857 052530  020401                                CMP     R4,R1                   ;WAS CONDITION CODE BITS TRANSFERRED
10858                                                                               ;CORRECTLY
10859 052532  001403                                BEQ     1$                      ;YES GO ON
10860 052534  004737  140132                        CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
10861 052540  104003                                ERROR   +3              ;FPP ERROR
10862                                                                       ;NO GO TO ERROR
10863 052542  000207                        1$:     RTS     PC                      ;RETURN
10864 052544                                FIN2:
10865                                       ;
10868 052544                                TSFP3:
10869                                       ;       TEST SETF, SETD, SETI, SETL
10870 052544  005037  140054                        CLR     @#TRPFLG                ;CLEAR TRAP FLAG
10871 052550  012704  000200                        MOV     #200,R4                 ;SETUP DATA TO BE LOADED
```

FLOATING POINT TESTS

```
10872 052554  170104                    LDFPS   R4                      ;LOAD FPS
10873 052556  170001                    SETF                            ;MAKE FD=0
10874 052560  170201                    STFPS   R1                      ;STORE FPS
10875 052562  020127  000000            CMP     R1,#0                   ;IS FD=0
10876 052566  001403                    BEQ     1$                      ;YES GO ON
10877 052570  004737  140132            CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
10878 052574  104003                    ERROR   .+3                   ;FPP ERROR
                                                                      ;NO GO TO ERROR
10879
10880 052576  170011            1$:     SETD                            ;MAKE FD=1
10881 052600  170201                    STFPS   R1                      ;STORE FPS
10882 052602  020104                    CMP     R1,R4                   ;IS FD=1
10883 052604  001403                    BEQ     2$                      ;YES GO ON
10884 052606  004737  140132            CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
10885 052612  104003                    ERROR   .+3                   ;FPP ERROR
                                                                      ;NO GO TO ERROR
10886
10887 052614  012704  000100      2$:   MOV     #100,R4                 ;SETUP DATA TO BE LOADED
10888 052620  170104                    LDFPS   R4                      ;LOAD FPS
10889 052622  170002                    SETI                            ;MAKE FL=0
10890 052624  170201                    STFPS   R1                      ;STORE FPS
10891 052626  020127  000000            CMP     R1,#0                   ;IS FL=0
10892 052632  001403                    BEQ     3$                      ;YES GO ON
10893 052634  004737  140132            CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
10894 052640  104003                    ERROR   .+3                   ;FPP ERROR
                                                                      ;NO GO TO ERROR
10895
10896 052642  170012            3$:     SETL                            ;MAKE FL=1
10897 052644  170201                    STFPS   R1                      ;STORE FPS
10898 052646  020104                    CMP     R1,R4                   ;IS FL=1
10899 052650  001403                    BEQ     4$                      ;YES GO ON
10900 052652  004737  140132            CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$;
10901 052656  104003                    ERROR   .+3                   ;FPP ERROR
                                                                      ;NO GO TO ERROR
10902
10903 052660                      4$:
10904                             ;
10907 052660                      TSFP4:
10908                             ;       TEST ILLEGAL OP CODES AND STST
10909 052660  005037  140054            CLR     @#TRPFLG                ;CLEAR TRAP FLAG
10910 052664  012705  170003            MOV     #170003,R5              ;INIT OP CODE
10911 052670  013746  000244            MOV     @#244,-(SP)             ;SAVE FP VECTOR
10912 052674  012737  053030  000244    MOV     #ILLOP1,@#244           ;SETUP NEW VECTOR
10913 052702  013746  000004            MOV     @#4,-(SP)               ;SAVE TIME OUT VECTOR
10914 052706  012737  053114  000004    MOV     #TIMEOU,@#4             ;SETUP NEW VECTOR
10915 052714  013746  000010            MOV     @#10,-(SP)              ;SAVE ILLEGAL VECTOR
10916 052720  012737  053124  000010    MOV     #ILLOP2,@#10            ;SETUP NEW VECTOR
10917 052726  005003            D1:     CLR     R3                      ;
10918 052730  170103                    LDFPS   R3                      ;CLEAR FPS
10919 052732  005002                    CLR     R2                      ;
10920 052734  010537  052740            MOV     R5,@#D2                 ;SETUP THE ILLEGAL INST
10921 052740  000000            D2:     .WORD   0                       ;
10922 052742  170000            D3:     CFCC                            ;MEMORY WORDS TO BE USED WITH
10923 052744  005202                    INC     R2                      ;EXECUTION OF ILLEGAL OP CODE
10924 052746  005202                    INC     R2                      ;
10925 052750  170201                    STFPS   R1                      ;SAVE FPS
10926 052752  004737  140132            CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
10927 052756  104003                    ERROR   .+3                   ;FPP ERROR
                                                                      ;GO TO ERROR
10928
10929 052760  022705  170010      D4:   CMP     #170010,R5              ;COMPUTE NEXT OP CODE
10930 052764  001003                    BNE     D5                      ;
```

FLOATING POINT TESTS

```
10931 052766  012705  170013            MOV     #170013,R5              ;
10932 052772  000755                    BR      D1                      ;
10933 052774  022705  170077    D5:     CMP     #170077,R5              ;
10934 053000  001001                    BNE     D6                      ;
10935 053002  000402                    BR      D7                      ;
10936 053004  005205            D6:     INC     R5                      ;
10937 053006  000747                    BR      D1                      ;
10938 053010  012637  000010    D7:     MOV     (SP)+,@#10              ;RESTORE VECTORS
10939 053014  012637  000004            MOV     (SP)+,@#4               ;
10940 053020  012637  000244            MOV     (SP)+,@#244             ;
10941
10942                           ;
10943 053024  000167  000104            JMP     FIN4
10944                           ;
10945 053030  022716  052742    ILLOP1: CMP     #D3,(SP)                ;DID TRAP OCCUR ON TEST INST
10946 053034  001403                    BEQ     1$                      ;YES GO ON
10947 053036  004737  140132            CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
10948 053042  104003                    ERROR   +3              ;FPP ERROR
                                                              ;NO GO TO ERROR
10949
10950 053044  022626            1$:     CMP     (SP)+,(SP)+             ;CLEAN UP STACK
10951 053046  170201                    STFPS   R1                      ;STORE FPS
10952 053050  022701  100000            CMP     #100000,R1              ;IS FPS CORRECT
10953 053054  001403                    BEQ     2$                      ;YES GO ON
10954 053056  004737  140132            CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
10955 053062  104003                    ERROR   +3              ;FPP ERROR
                                                              ;NO GO TO ERROR
10956
10957 053064  005004            2$:     CLR     R4                      ;INT R4 TO A KNOWN STATE
10958 053066  170304                    STST    R4                      ;STORE FEC AT R4
                                                                      ;IF THE DESTINATION MODE IS IMPROPERLY
                                                                      ;DECODED AN ODD ADDRESS TRAP TO 4
10959                                                                 ;SHOULD OCCUR
10960
10961
10962 053070  022704  000002            CMP     #2,R4                   ;IS FEC CORRECT
10963 053074  001002                    BNE     3$                      ;NO GO TO ERROR
10964 053076  000167  177656            JMP     D4                      ;YES GO ON
10965 053102  004737  140132    3$:     CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
10966 053106  104003                    ERROR   +3              ;FPP ERROR
                                                              ;GO TO ERROR
10967
10968 053110  000167  177644            JMP     D4                      ;THEN GO ON
10969                           ;
10970 053114  004737  140132    TIMEOU: CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
10971 053120  104003                    ERROR   +3              ;FPP ERROR
                                                              ;ERROR BECAUSE OF TRAP TO 4
10972
10973 053122  000006                    RTT                             ;RETURN
10974                           ;
10975 053124  004737  140132    ILLOP2: CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
10976 053130  104003                    ERROR   +3              ;FPP ERROR
                                                              ;ERROR BECAUSE OF TRAP TO 10
10977
10978 053132  000006                    RTT                             ;RETURN
10979 053134                    FIN4:
10980                           ;
10983 053134                    TSFP5:
10984                           ;       TEST FID (INTERRUPT DISABLE BIT)
10985 053134  005037  140054            CLR     @#TRPFLG                ;CLEAR TRAP FLAG
10986 053140  013746  000244            MOV     @#244,-(SP)             ;SAVE FP VECTOR
10987 053144  012737  053230  000244    MOV     #ILL,@#244              ;SETUP NEW VECTOR
10988 053152  012703  040000            MOV     #40000,R3               ;SETUP DATA TO BE LOADED
10989 053156  170103                    LDFPS   R3                      ;LOAD FPS, FID=1
```

FLOATING POINT TESTS

```
10990 053160  170020                    .WORD   170020                  ;ILLEGAL FP INSTRUCTION
10991 053162  170000                    CFCC
10992 053164  170201                    STFPS   R1                      ;SEE IF ERROR WAS RECORDED IN FPS
10993 053166  022701  140000            CMP     #140000,R1              ;
10994 053172  001403                    BEQ     1$                      ;YES GO ON
10995 053174  004737  140132            CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
10996 053200  104003                    ERROR   +3              ;FPP ERROR
10997                                                           ;NO GO TO ERROR
10998 053202  170304            1$:     STST    R4                      ;SEE IF FEC=2
10999 053204  022704  000002            CMP     #2,R4                   ;
11000 053210  001403                    BEQ     2$                      ;YES GO ON
11001 053212  004737  140132            CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11002 053216  104003                    ERROR   +3              ;FPP ERROR
11003                                                           ;NO GO TO ERROR
11004 053220  012637  000244    2$:     MOV     (SP)+,@#244             ;RESTORE VECTOR
11005                                   ;
11006 053224  000167  000010            JMP     FIN5
11007                                   ;
11008 053230  004737  140132    ILL:    CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11009 053234  104003                    ERROR   +3              ;FPP ERROR
11010                                                           ;FID ERROR
11011 053236  000006                    RTT                             ;RETURN
11012 053240                    FIN5:
11013                                   ;
11016 053240                    TSFP6:
11017                                   ;       TEST LDD, STD FSRC AND FDST MODE 1
11018 053240  005037  140054            CLR     @#TRPFLG                ;CLEAR TRAP FLAG
11019 053244  005004                    CLR     R4                      ;SETUP TO LOAD DATA
11020 053246  170104                    LDFPS   R4                      ;CLEAR FPS
11021 053250  170011                    SETD                            ;SET FD TO 1
11022 053252  013746  000004            MOV     @#4,-(SP)               ;SAVE TIMEOUT VECTOR
11023 053256  012737  053430  000004    MOV     #TSF6,@#4               ;SETUP NEW VECTOR
11024 053264  012704  053420            MOV     #TS6DAT,R4              ;SETUP POINTER TO DATA
11025 053270  172414                    LDD     (R4),AC0                ; TEST INSTRUCTION
11026 053272  020427  053420            CMP     R4,#TS6DAT              ;IS R4 CORRECT
11027 053276  001403                    BEQ     1$                      ;YES GO ON
11028 053300  004737  140132            CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11029 053304  104003                    ERROR   +3              ;FPP ERROR
11030                                                           ;NO GO TO ERROR
11031 053306  012701  053410    1$:     MOV     #TS6DA,R1               ;SETUP POINTER TO DATA
11032 053312  012703  000004            MOV     #4,R3                   ;INIT COUNTER
11033 053316  022421            2$:     CMP     (R4)+,(R1)+             ;WAS SOURCE DATA ALTERED
11034 053320  001403                    BEQ     3$                      ;NO GO ON
11035 053322  004737  140132            CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11036 053326  104003                    ERROR   +3              ;FPP ERROR
11037                                                           ;YES GO TO ERROR
11038 053330  077306            3$:     SOB     R3,2$                   ;ARE WE DONE
11039 053332  012704  003162            MOV     #TSTLOC,R4              ;SETUP POINTER FOR DATA
11040 053336  174014                    STD     AC0,(R4)                ; TEST INSTRUCTION
11041 053340  020427  003162            CMP     R4,#TSTLOC                      ;IS R4 CORRECT
11042 053344  001403                    BEQ     4$                      ;YES GO ON
11043 053346  004737  140132            CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11044 053352  104003                    ERROR   +3              ;FPP ERROR
11045                                                           ;NO GO TO ERROR
11046 053354  012701  053410    4$:     MOV     #TS6DA,R1               ;SETUP POINTER TO DATA
11047 053360  012703  000004            MOV     #4,R3                   ;INIT COUNTER
11048 053364  022421            5$:     CMP     (R4)+,(R1)+             ;IS DESTINATION DATA CORRECT
```

FLOATING POINT TESTS

```
11049 053366  001403                    BEQ     6$                      ;YES GO ON
11050 053370  004737  140132            CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11051 053374  104003                    ERROR   .+3                     ;FPP ERROR
11052                                                                   ;NO GO TO ERROR
11053 053376  077306            6$:     SOB     R3,5$                   ;ARE WE DONE
11054 053400  012637  000004            MOV     (SP)+,@#4               ;RESTORE VECTOR
11055
11056                           ;
11057 053404  000167  000030            JMP     FIN6
11058                           ;
11059 053410  177777            TS6DA:  .WORD   177777
11060 053412  000000                    .WORD   000000
11061 053414  052525                    .WORD   052525
11062 053416  125252                    .WORD   125252
11063 053420  177777            TS6DAT: .WORD   177777
11064 053422  000000                    .WORD   000000
11065 053424  052525                    .WORD   052525
11066 053426  125252                    .WORD   125252
11067                           ;
11068 053430  004737  140132    TSF6:   CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11069 053434  104003                    ERROR   .+3                     ;FPP ERROR
11070                                                                   ;ODD ADDRESS TRAP
11071 053436  000006                    RTT                             ;RETURN
11072 053440                    FIN6:
11073                           ;
11076 053440                    TSFP7:
11077                           ;       TEST LDD, LDF FSRC MODE 0
11078 053440  005037  140054            CLR     @#TRPFLG                ;CLEAR TRAP FLAG
11079 053444  012704  000200            MOV     #200,R4                 ;SETUP TO LOAD FPS
11080 053450  170104                    LDFPS   R4                      ;LOAD FPS, FD=1
11081 053452  013746  000004            MOV     @#4,-(SP)               ;SAVE TIMEOUT VECTOR
11082 053456  012737  053624  000004    MOV     #TSF7,@#4               ;SETUP NEW VECTOR
11083 053464  012704  053634            MOV     #TS7DA1,R4              ;SETUP POINTER TO DATA
11084 053470  172414                    LDD     (R4),AC0                ;CLEAR AC0
11085 053472  012701  053644            MOV     #TS7DA2,R1              ;SETUP POINTER TO DATA
11086 053476  172511                    LDD     (R1),AC1                ;LOAD AC1 WITH DATA
11087 053500  172401                    LDD     AC1,AC0                 ; TEST INSTRUCTION
11088 053502  012704  003162            MOV     #TSTLOC,R4              ;
11089 053506  174114                    STD     AC1,(R4)                ;CHECK IF AC1 HAS BEEN ALTERED
11090 053510  004767  000072            JSR     PC,CHECK7               ;
11091 053514  012704  003162            MOV     #TSTLOC,R4              ;SETUP POINTERS FOR DATA
11092 053520  012701  053644            MOV     #TS7DA2,R1              ;
11093 053524  174014                    STD     AC0,(R4)                ;CHECK IF AC0 RECEIVED CORRECT DATA
11094 053526  004767  000054            JSR     PC,CHECK7               ;
11095 053532  012701  053634            MOV     #TS7DA1,R1              ;SETUP POINTER TO DATA
11096 053536  172511                    LDD     (R1),AC1                ;CLEAR AC1
11097 053540  170001                    SETF                            ;SET FD=0
11098 053542  172401                    LDF     AC1,AC0                 ; TEST INSTRUCTION
11099 053544  170011                    SETD                            ;SET FD=1
11100 053546  012704  003162            MOV     #TSTLOC,R4              ;SETUP POINTER TO DATA
11101 053552  174114                    STD     AC1,(R4)                ;CHECK IF AC1 HAS BEEN ALTERED
11102 053554  004767  000026            JSR     PC,CHECK7               ;
11103 053560  012704  053654            MOV     #TS7DA4,R4              ;SETUP POINTERS FOR DATA
11104 053564  012701  003162            MOV     #TSTLOC,R1              ;
11105 053570  174011                    STD     AC0,(R1)                ;CHECK IF AC0 HAS CORRECT DATA
11106 053574  004767  000010            JSR     PC,CHECK7               ;
11107 053576  012637  000004            MOV     (SP)+,@#4               ;RESTORE VECTOR
```

FLOATING POINT TESTS

```
11108
11109                                   ;
11110 053602  000167  000056                      JMP     FIN7
11111                                   ;
11112 053606  012703  000004            CHECK7:   MOV     #4,R3                 ;INIT COUNTER
11113 053612  022421                    CHEK7:    CMP     (R4)+,(R1)+           ;IS DATA OK
11114 053614  001401                              BEQ     CHK7                  ;YES GO ON
11115 053616  104003                              ERROR   +3            ;FPP ERROR
11116                                                                   ;NO GO TO ERROR
11117 053620  077304                    CHK7:     SOB     R3,CHEK7              ;ARE WE DONE
11118 053622  000207                              RTS     PC                    ;YES WE RN
11119                                   ;
11120 053624  004737  140132            TSF7:     CALL    @#DETFPA              ;DETERM. IF FLOATING POINT FAULT. $$$
11121 053630  104003                              ERROR   +3            ;FPP ERROR
11122                                                                   ;ODD ADDRESS TRAP
11123 053632  000006                              RTT                   ;RETURN
11124                                   ;
11125 053634  000000                    TS7DA1:   .WORD   0
11126 053636  000000                              .WORD   0
11127 053640  000000                              .WORD   0
11128 053642  000000                              .WORD   0
11129 053644  037641                    TS7DA2:   .WORD   37641
11130 053646  065121                              .WORD   65121
11131 053650  037373                              .WORD   37373
11132 053652  022265                              .WORD   22265
11133 053654  000000                    TS7DA4:   .WORD   0
11134 053656  000000                              .WORD   0
11135 053660  037373                              .WORD   37373
11136 053662  022265                              .WORD   22265
11137 053664                            FIN7:
11138                                   ;
11141 053664                            TSFP10:
11142                                   ;         TEST STD, STF FDST MODE 0
11143 053664  005037  140054                      CLR     @#TRPFLG              ;CLEAR TRAP FLAG
11144 053670  012704  000200                      MOV     #200,R4               ;SETUP TO LOAD FPS
11145 053674  170104                              LDFPS   R4                    ;LOAD FPS, FD=1
11146 053676  013746  000004                      MOV     @#4,-(SP)             ;SAVE TIMEOUT VECTOR
11147 053702  012737  054054  000004              MOV     #TSF10,@#4            ;SETUP NEW VECTOR
11148 053710  012704  054064                      MOV     #TS10D1,R4            ;SETUP POINTER TO DATA
11149 053714  172414                              LDD     (R4),AC0              ;CLEAR AC0
11150 053716  012701  054074                      MOV     #TS10D2,R1            ;SETUP POINTER TO DATA
11151 053722  172511                              LDD     (R1),AC1              ;LOAD AC1 WITH DATA
11152 053724  174100                              STD     AC1,AC0               ; TEST INSTRUCTION
11153 053726  012704  003162                      MOV     #TSTLOC,R4            ;
11154 053732  174114                              STD     AC1,(R4)              ;CHECK IF AC1 HAS BEEN ALTERED
11155 053734  004767  000072                      JSR     PC,CHEC10             ;
11156 053740  012704  003162                      MOV     #TSTLOC,R4            ;SETUP POINTERS FOR DATA
11157 053744  012701  054074                      MOV     #TS10D2,R1            ;
11158 053750  174014                              STD     AC0,(R4)              ;CHECK IF AC0 RECEIVED CORRECT DATA
11159 053752  004767  000054                      JSR     PC,CHEC10             ;
11160 053756  012701  054064                      MOV     #TS10D1,R1            ;SETUP POINTER TO DATA
11161 053762  172511                              LDD     (R1),AC1              ;CLEAR AC1
11162 053764  170001                              SETF                          ;SET FD=0
11163 053766  174100                              STF     AC1,AC0               ; TEST INSTRUCTION
11164 053770  170011                              SETD                          ;SET FD=1
11165 053772  012704  003162                      MOV     #TSTLOC,R4            ;SETUP POINTER TO DATA
11166 053776  174114                              STD     AC1,(R4)              ;CHECK IF AC1 HAS BEEN ALTERED
```

E16

FLOATING POINT TESTS

```
11167 054000  004767  000026                JSR     PC,CHEC10               ;
11168 054004  012704  054104                MOV     #TS10D4,R4              ;SETUP POINTERS FOR DATA
11169 054010  012701  003162                MOV     #TSTLOC,R1              ;
11170 054014  174011                        STD     AC0,(R1)                ;CHECK IF AC0 HAS CORRECT DATA
11171 054016  004767  000010                JSR     PC,CHEC10               ;
11172 054022  012637  000004                MOV     (SP)+,@#4               ;RESTORE VECTOR
11173
11174                               ;
11175 054026  000167  000062                JMP     FIN10
11176                               ;
11177 054032  012703  000004        CHEC10: MOV     #4,R3                   ;INIT COUNTER
11178 054036  022421                CH10:   CMP     (R4)+,(R1)+             ;IS DATA OK
11179 054040  001403                        BEQ     CHK10                   ;YES GO ON
11180 054042  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11181 054046  104003                        ERROR   +3              ;FPP ERROR
11182                                                               ;NO GO TO ERROR
11183 054050  077306                CHK10:  SOB     R3,CH10                 ;ARE WE DONE
11184 054052  000207                        RTS     PC                      ;YES RETURN
11185                               ;
11186 054054  004737  140132        TSF10:  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11187 054060  104003                        ERROR   +3              ;FPP ERROR
11188                                                               ;ODD ADDRESS TRAP
11189 054062  000006                        RTT                             ;
11190                               ;
11191 054064  000000                TS10D1: .WORD   0
11192 054066  000000                        .WORD   0
11193 054070  000000                        .WORD   0
11194 054072  000000                        .WORD   0
11195 054074  177777                TS10D2: .WORD   177777
11196 054076  111236                        .WORD   111236
11197 054100  100045                        .WORD   100045
11198 054102  003651                        .WORD   3651
11199 054104  000000                TS10D4: .WORD   0
11200 054106  000000                        .WORD   0
11201 054110  100045                        .WORD   100045
11202 054112  003651                        .WORD   3651
11203 054114                        FIN10:
11204                               ;
11207 054114                        TSFP11:
11208                               ;       TEST FDST SINGLE OPERAND MODE 0
11209 054114  005037  140054                CLR     @#TRPFLG                ;CLEAR TRAP FLAG
11210 054120  012704  000200                MOV     #200,R4                 ;SETUP TO LOAD FPS
11211 054124  170104                        LDFPS   R4                      ;SET FD=1
11212 054126  012704  054210                MOV     #TS11D1,R4              ;SETUP POINTER TO DATA
11213 054132  172414                        LDD     (R4),AC0                ;LOAD ALL ONES TO AC0
11214 054134  170400                        CLRD    AC0                     ; TEST INSTRUCTION
11215 054136  170203                        STFPS   R3                      ;GET FPS
11216 054140  012704  003162                MOV     #TSTLOC,R4              ;
11217 054144  174014                        STD     AC0,(R4)                ;CHECK AC0 FOR ALL ZEROES
11218 054146  012701  000004                MOV     #4,R1                   ;INIT COUNTER
11219 054152  022427  000000        1$:     CMP     (R4)+,#0                ;
11220 054156  001403                        BEQ     2$                      ;OK GO ON
11221 054160  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11222 054164  104003                        ERROR   +3              ;FPP ERROR
11223                                                               ;NO GO TO ERROR
11224 054166  077107                2$:     SOB     R1,1$                   ;ARE WE DONE
11225 054170  020327  000204                CMP     R3,#204                 ;CHECK FPS
```

F16

FLOATING POINT TESTS

```
11226 054174  001403                        BEQ     3$                      ;OK GO ON
11227 054176  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11228 054202  104003                        ERROR   +3              ;FPP ERROR
11229                                                               ;NO GO TO ERROR
11230 054204                        3$:
11231                               ;
11232 054204  000167  000010                JMP     FIN11
11233                               ;
11234 054210  177777                TS11D1: .WORD   177777
11235 054212  177777                        .WORD   177777
11236 054214  177777                        .WORD   177777
11237 054216  177777                        .WORD   177777
11238 054220                        FIN11:
11239                               ;
11242 054220                        TSFP12:
11243                               ;       TEST FDST SOP MODE 0 WITH ILLEGAL AC7
11244 054220  005037  140054                CLR     @#TRPFLG                ;CLEAR TRAP FLAG
11245 054224  012703  040200                MOV     #40200,R3               ;SETUP TO LOAD FPS
11246 054230  170103                        LDFPS   R3                      ;SET FID=1, AND FD=1
11247 054232  170407                        CLRD    AC7                     ; TEST INSTRUCTION
11248 054234  170204                        STFPS   R4                      ;GET FPS
11249 054236  170305                        STST    R5                      ;GET FEC
11250 054240  022704  140200                CMP     #140200,R4              ;IS FPS CORRECT
11251 054244  001403                        BEQ     1$                      ;YES GO ON
11252 054246  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11253 054252  104003                        ERROR   +3              ;FPP ERROR
11254                                                               ;NO GO TO ERROR
11255 054254  022705  000002        1$:     CMP     #2,R5                   ;IS FEC CORRECT
11256 054260  001403                        BEQ     2$                      ;YES GO ON
11257 054262  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11258 054266  104003                        ERROR   +3              ;FPP ERROR
11259                                                               ;NO GO TO ERROR
11260 054270                        2$:
11261                               ;
11264 054270                        TSFP13:
11265                               ;       TEST FDST SOP MODE 1
11266 054270  013746  000004                MOV     @#4,-(SP)               ;SAVE TIMEOUT VECTOR
11267 054274  012737  054424  000004        MOV     #TSF13,@#4              ;SETUP NEW VECTOR
11268 054302  005037  140054                CLR     @#TRPFLG                ;CLEAR TRAP FLAG
11269 054306  012702  000200                MOV     #200,R2                 ;SETUP TO LOAD FPS
11270 054312  170102                        LDFPS   R2                      ;SET FD=1
11271 054314  012705  000004                MOV     #4,R5                   ;INIT COUNTER
11272 054320  012704  003162                MOV     #TSTLOC,R4              ;SETUP POINTER TO TEST LOCATION
11273 054324  012724  177777        100$:   MOV     #177777,(R4)+           ;MOVE ALL ONES TO TEST LOCATION
11274 054330  077503                        SOB     R5,100$                 ;ARE WE DONE
11275 054332  012702  003162                MOV     #TSTLOC,R2                      ;SETUP POINTER TO DATA
11276 054336  170412                        CLRD    (R2)                    ; TEST INSTRUCTION
11277 054340  170203                        STFPS   R3                      ;GET FPS
11278 054342  020227  003162                CMP     R2,#TSTLOC                      ;WAS R2 ALTERED
11279 054346  001403                        BEQ     1$                      ;NO GO ON
11280 054350  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11281 054354  104003                        ERROR   +3              ;FPP ERROR
11282                                                               ;YES GO TO ERROR
11283 054356  012701  000004        1$:     MOV     #4,R1                   ;INIT COUNTER
11284 054362  022227  000000        2$:     CMP     (R2)+,#0                ;CHECK LOCATION FOR 0
11285 054366  001403                        BEQ     3$                      ;OK GO ON
11286 054370  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
```

G16

FLOATING POINT TESTS

```
11287 054374  104003                    ERROR   +3              ;FPP ERROR
11288                                                           ;NO GO TO ERROR
11289 054376  077107            3$:     SOB     R1,2$                   ;ARE WE DONE
11290 054400  020327  000204              CMP     R3,#204                 ;CHECK FPS
11291 054404  001403                      BEQ     4$                      ;OK GO ON
11292 054406  004737  140132              CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11293 054412  104003                      ERROR   +3              ;FPP ERROR
11294                                                           ;NO GO TO ERROR
11295 054414  012637  000004      4$:     MOV     (SP)+,@#4               ;RESTORE VECTOR
11296
11297                             ;
11298 054420  000167  000010              JMP     FIN13
11299                             ;
11300 054424  004737  140132      TSF13:  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11301 054430  104003                      ERROR   +3              ;FPP ERROR
11302                                                           ;ODD ADDRESS TRAP
11303 054432  000006                      RTT                     ;RETURN
11304                             ;
11305 054434                      FIN13:
11306                             ;
11309 054434                      TSFP14:
11310                             ;       TEST FDST SOP MODE 2
11311 054434  013746  000004              MOV     @#4,-(SP)               ;SAVE TIMEOUT VECTOR
11312 054440  012737  054574  000004      MOV     #TSF14,@#4              ;SETUP NEW VECTOR
11313 054446  005037  140054              CLR     @#TRPFLG                ;CLEAR TRAP FLAG
11314 054452  012702  000200              MOV     #200,R2                 ;SETUP TO LOAD FPS
11315 054456  170102                      LDFPS   R2                      ;SET FD=1
11316 054460  012705  000004              MOV     #4,R5                   ;INIT COUNTER
11317 054464  012704  003162              MOV     #TSTLOC,R4              ;SETUP POINTER TO TEST LOCATION
11318 054470  012724  177777      100$:   MOV     #177777,(R4)+           ;MOVE ALL ONES TO TEST LOCATION
11319 054474  077503                      SOB     R5,100$                 ;ARE WE DONE
11320 054476  012702  003162              MOV     #TSTLOC,R2                      ;SETUP POINTER TO DATA
11321 054502  170422                      CLRD    (R2)+                   ; TEST INSTRUCTION
11322 054504  170203                      STFPS   R3                      ;GET FPS
11323 054506  020227  003172              CMP     R2,#TSTLOC+10                   ;IS R2 CORRECT
11324 054512  001403                      BEQ     1$                      ;YES GO ON
11325 054514  004737  140132              CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11326 054520  104003                      ERROR   +3              ;FPP ERROR
11327                                                           ;NO GO TO ERROR
11328 054522  012702  003162      1$:     MOV     #TSTLOC,R2                      ;SETUP POINTER TO DATA
11329 054526  012701  000004              MOV     #4,R1                   ;INIT COUNTER
11330 054532  022227  000000      2$:     CMP     (R2)+,#0                ;CHECK LOCATION FOR 0
11331 054536  001403                      BEQ     3$                      ;YES GO ON
11332 054540  004737  140132              CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11333 054544  104003                      ERROR   +3              ;FPP ERROR
11334                                                           ;NO GO TO ERROR
11335 054546  077107            3$:     SOB     R1,2$                   ;ARE WE DONE
11336 054550  020327  000204              CMP     R3,#204                 ;CHECK FPS
11337 054554  001403                      BEQ     4$                      ;OK GO ON
11338 054556  004737  140132              CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11339 054562  104003                      ERROR   +3              ;FPP ERROR
11340                                                           ;NO GO TO ERROR
11341 054564  012637  000004      4$:     MOV     (SP)+,@#4               ;RESTORE VECTOR
11342
11343                             ;
11344 054570  000167  000010              JMP     FIN14
11345                             ;
```

FLOATING POINT TESTS

```
11346 054574  004737  140132          TSF14:  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11347 054600  104003                          ERROR   +3              ;FPP ERROR
11348                                                                 ;ODD ADDRESS TRAP
11349 054602  000006                          RTT                             ;RETURN
11350                                         ;
11351 054604  000240                  FIN14:  NOP
11352                                         ;
11355                                         ;
11356                                         ;----   ---- -- -- - ----- ------
11357                                         ;TEST  FDST SOP MODE 3
11358                                         ;
11359                                         ;
11360 054606                          TSFP15:
11361                                         ;
11362 054606  013746  000004                  MOV     @#4,-(SP)               ;SAVE TIMEOUT VECTOR
11363 054612  012737  055012  000004          MOV     #TSF15,@#4              ;SETUP NEW VECTOR
11364 054620  005037  140054                  CLR     @#TRPFLG                ;CLEAR TRAP FLAG
11365 054624  012702  000200                  MOV     #200,R2                 ;SETUP TO LOAD FPS
11366 054630  170102                          LDFPS   R2                      ;SET FD=1
11367 054632  012705  000011                  MOV     #9.,R5                  ;INIT COUNTER
11368 054636  012704  003162                  MOV     #TSTLOC,R4              ;SETUP POINTER TO TEST LOCATION
11369 054642  012724  177777          100$:   MOV     #177777,(R4)+           ;INIT TEST LOCATION
11370 054646  077503                          SOB     R5,100$                 ;ARE WE DONE
11371 054650  012737  003174  003162          MOV     #TSTLOC+12,@#TSTLOC             ;INIT TEST LOCATION
11372 054656  012702  003162                  MOV     #TSTLOC,R2              ;SETUP POINTER TO DATA
11373 054662  170432                          CLRD    @(R2)+          ; TEST INSTRUCTION
11374 054664  170203                          STFPS   R3                      ;GET FPS
11375 054666  020227  003164                  CMP     R2,#TSTLOC+2                    ;IS R2 CORRECT
11376 054672  001403                          BEQ     1$                      ;YES GO ON
11377 054674  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11378 054700  104003                          ERROR   +3              ;FPP ERROR
11379                                                                 ;NO GO TO ERROR
11380 054702  012702  003162          1$:     MOV     #TSTLOC,R2                      ;SETUP POINTER TO DATA
11381 054706  022227  003174                  CMP     (R2)+,#TSTLOC+12                ;IS DATA CORRECT
11382 054712  001403                          BEQ     2$              ;YES GO ON
11383 054714  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11384 054720  104003                          ERROR   +3              ;FPP ERROR
11385                                                                 ;NO GO TO ERROR
11386 054722  012701  000004          2$:     MOV     #4,R1                   ;INIT COUNTER
11387 054726  022227  177777          3$:     CMP     (R2)+,#177777           ;IS LOCATION ALL ONES
11388 054732  001403                          BEQ     4$              ;YES GO ON
11389 054734  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11390 054740  104003                          ERROR   +3              ;FPP ERROR
11391                                                                 ;NO GO TO ERROR
11392
11393 054742  077107                  4$:     SOB     R1,3$                   ;ARE WE DONE
11394 054744  012701  000004                  MOV     #4,R1                   ;INIT COUNTER
11395 054750  022227  000000          5$:     CMP     (R2)+,#0                ;IS LOCATION 0
11396 054754  001403                          BEQ     6$              ;YES GO ON
11397 054756  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11398 054762  104003                          ERROR   +3              ;FPP ERROR
11399                                                                 ;NO GO TO ERROR
11400 054764  077107                  6$:     SOB     R1,5$                   ;ARE WE DONE
11401 054766  020327  000204                  CMP     R3,#204                 ;CHECK FPS
11402 054772  001403                          BEQ     7$              ;OK GO ON
11403 054774  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11404 055000  104003                          ERROR   +3              ;FPP ERROR
```

FLOATING POINT TESTS

```
11405                                                          ;NO GO TO ERROR
11406 055002  012637  000004          7$:     MOV     (SP)+,@#4        ;RESTORE VECTOR
11407 055006  000167  000010                  JMP     FIN15
11408                                 ;
11409 055012  004737  140132          TSF15:  CALL    @#DETFPA         ;DETERMINE FLOATING POINT FAULT. $$$
11410 055016  104003                          ERROR   +3               ;FPP ERROR
11411                                                                  ;ODD ADDRESS TRAP
11412 055020  000006                          RTT                      ;RETURN
11413                                 ;
11414 055022  000240                  FIN15:  NOP
11415                                 ;
11418                                 ;
11419                                 ;-------------- -- -  -
11420                                 ;TEST  FDST SOP MODE 4
11421                                 ;
11422                                 ;
11423 055024                          TSFP16:
11424                                 ;
11425 055024  013746  000004                  MOV     @#4,-(SP)        ;SAVE TIMEOUT VECTOR
11426 055030  012737  055202  000004          MOV     #TSF16,@#4       ;SETUP NEW VECTOR
11427 055036  005037  140054                  CLR     @#TRPFLG         ;CLEAR TRAP FLAG
11428 055042  012702  000200                  MOV     #200,R2          ;SETUP TO LOAD FPS
11429 055046  170102                          LDFPS   R2               ;SET FD=1
11430 055050  012705  000010                  MOV     #8.,R5           ;INIT COUNTER
11431 055054  012704  003162                  MOV     #TSTLOC,R4       ;SETUP POINTER TO TEST LOCATION
11432 055060  012724  177777          100$:   MOV     #177777,(R4)+    ;INIT TEST LOCATION
11433 055064  077503                          SOB     R5,100$          ;ARE WE DONE
11434 055066  012702  003172                  MOV     #TSTLOC+10,R2            ;SETUP POINTER TO DATA
11435 055072  170442                          CLRD    -(R2)            ; TEST INSTRUCTION
11436 055074  170203                          STFPS   R3               ;GET FPS
11437 055076  020227  003162                  CMP     R2,#TSTLOC               ;IS R2 CORRECT
11438 055102  001403                          BEQ     1$               ;YES GO ON
11439 055104  004737  140132                  CALL    @#DETFPA         ;DETERMINE FLOATING POINT FAULT. $$$
11440 055110  104003                          ERROR   +3               ;FPP ERROR
11441                                                                  ;NO GO TO ERROR
11442 055112  012701  000004          1$:     MOV     #4,R1            ;INIT COUNTER
11443 055116  022227  000000          2$:     CMP     (R2)+,#0         ;IS LOCATION 0
11444 055122  001403                          BEQ     3$               ;YES GO ON
11445 055124  004737  140132                  CALL    @#DETFPA         ;DETERMINE FLOATING POINT FAULT. $$$
11446 055130  104003                          ERROR   +3               ;FPP ERROR
11447                                                                  ;NO GO TO ERROR
11448 055132  077107                  3$:     SOB     R1,2$            ;ARE WE DONE
11449 055134  012701  000004                  MOV     #4,R1            ;INIT COUNTER
11450 055140  022227  177777          4$:     CMP     (R2)+,#177777    ;IS LOCATION UNCHANGED
11451 055144  001403                          BEQ     5$               ;YES GO ON
11452 055146  004737  140132                  CALL    @#DETFPA         ;DETERMINE FLOATING POINT FAULT. $$$
11453 055152  104003                          ERROR   +3               ;FPP ERROR
11454                                                                  ;NO GO TO ERROR
11455 055154  077107                  5$:     SOB     R1,4$            ;ARE WE DONE
11456 055156  020327  000204                  CMP     R3,#204          ;CHECK FPS
11457 055162  001403                          BEQ     6$               ;OK GO ON
11458 055164  004737  140132                  CALL    @#DETFPA         ;DETERMINE FLOATING POINT FAULT. $$$
11459 055170  104003                          ERROR   +3               ;FPP ERROR
11460                                                                  ;NO GO TO ERROR
11461 055172  012637  000004          6$:     MOV     (SP)+,@#4        ;RESTORE VECTOR
11462 055176  000167  000010                  JMP     FIN16
11463                                 ;
```

FLOATING POINT TESTS

```
11464 055202  004737  140132          TSF16:  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11465 055206  104003                          ERROR   +3              ;FPP ERROR
11466                                                                 ;ODD ADDRESS TRAP
11467 055210  000006                          RTT                             ;RETURN
11468                                 ;
11469 055212  000240                  FIN16:  NOP
11470                                 ;
11473                                 ;
11474                                 ;---    ----- ---     ---  -----
11475                                 ;TEST   FDST SOP MODE 5
11476                                 ;
11477                                 ;
11478 055214                          TSFP17:
11479                                 ;
11480 055214  013746  000004                  MOV     @#4,-(SP)               ;SAVE TIMEOUT VECTOR
11481 055220  012737  055414  000004          MOV     #TSF17,@#4              ;SETUP NEW VECTOR
11482 055226  005037  140054                  CLR     @#TRPFLG                ;CLEAR TRAP FLAG
11483 055232  012702  000200                  MOV     #200,R2                 ;SETUP TO LOAD FPS
11484 055236  170102                          LDFPS   R2                      ;SET FD=1
11485 055240  012705  000011                  MOV     #9.,R5                  ;INIT COUNTER
11486 055244  012704  003162                  MOV     #TSTLOC,R4              ;SETUP POINTER TO TEST LOCATION
11487 055250  012724  177777          100$:   MOV     #177777,(R4)+           ;INIT TEST LOCATION
11488 055254  077503                          SOB     R5,100$                 ;ARE WE DONE
11489 055256  012737  003174  003162          MOV     #TSTLOC+12,@#TSTLOC             ;INIT TEST LOCATION
11490 055264  012702  003164                  MOV     #TSTLOC+2,R2                    ;SETUP POINTER TO DATA
11491 055270  170452                          CLRD    @-(R2)                  ; TEST INSTRUCTION
11492 055272  170203                          STFPS   R3                      ;GET FPS
11493 055274  020227  003162                  CMP     R2,#TSTLOC                      ;IS R2 CORRECT
11494 055300  001403                          BEQ     1$                      ;YES GO ON
11495 055302  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11496 055306  104003                          ERROR   +3              ;FPP ERROR
11497                                                                 ;NO GO TO ERROR
11498 055310  022227  003174          1$:     CMP     (R2)+,#TSTLOC+12                ;IS DATA CORRECT
11499 055314  001403                          BEQ     2$                      ;YES GO ON
11500 055316  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11501 055322  104003                          ERROR   +3              ;FPP ERROR
11502                                                                 ;NO GO TO ERROR
11503 055324  012701  000004          2$:     MOV     #4,R1                   ;INIT COUNTER
11504 055330  022227  177777          3$:     CMP     (R2)+,#177777           ;IS LOCATION ALL ONES
11505 055334  001403                          BEQ     4$                      ;YES GO ON
11506 055336  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11507 055342  104003                          ERROR   +3              ;FPP ERROR
11508                                                                 ;NO GO TO ERROR
11509 055344  077107                  4$:     SOB     R1,3$                   ;ARE WE DONE
11510 055346  012701  000004                  MOV     #4,R1                   ;INIT COUNTER
11511 055352  022227  000000          5$:     CMP     (R2)+,#0                ;IS LOCATION 0
11512 055356  001403                          BEQ     6$                      ;YES GO ON
11513 055360  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11514 055364  104003                          ERROR   +3              ;FPP ERROR
11515                                                                 ;NO GO TO ERROR
11516 055366  077107                  6$:     SOB     R1,5$                   ;ARE WE DONE
11517 055370  020327  000204                  CMP     R3,#204                 ;CHECK FPS
11518 055374  001403                          BEQ     7$                      ;OK GO ON
11519 055376  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11520 055402  104003                          ERROR   +3              ;FPP ERROR
11521                                                                 ;NO GO TO ERROR
11522 055404  012637  000004          7$:     MOV     (SP)+,@#4               ;RESTORE VECTOR
```

K16

FLOATING POINT TESTS

```
11523 055410  000167  000010                    JMP     FIN17
11524
11525 055414  004737  140132          TSF17:    CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11526 055420  104003                            ERROR   +3                      ;FPP ERROR
11527                                                                           ;ODD ADDRESS TRAP
11528 055422  000006                            RTT                             ;RETURN
11529
11530 055424  000240          FIN17:    NOP
11531                           ;
11534                           ;
11535                           ;----- ----    --- -----    -----
11536                           ;TEST  FDST SOP MODE 6
11537                           ;
11538                           ;
11539 055426                  TSFP20:
11540                           ;
11541 055426  005037  140054            CLR     @#TRPFLG                ;CLEAR TRAP FLAG
11542 055432  013746  000004            MOV     @#4,-(SP)               ;SAVE TIMEOUT VECTOR
11543 055436  012737  055612  000004    MOV     #TSF20,@#4              ;SETUP NEW VECTOR
11544 055444  012702  000200            MOV     #200,R2                 ;SETUP TO LOAD FPS
11545 055450  170102                    LDFPS   R2                      ;SET FD=1
11546 055452  012705  000010            MOV     #8.,R5                  ;INIT COUNTER
11547 055456  012704  003162            MOV     #TSTLOC,R4              ;SETUP POINTER TO TEST LOCATION
11548 055462  012724  177777    100$:   MOV     #177777,(R4)+           ;INIT TEST LOCATION
11549 055466  077503                    SOB     R5,100$                 ;ARE WE DONE
11550 055470  012702  003163            MOV     #TSTLOC+1,R2                    ;SETUP POINTER TO DATA
11551 055474  170462  000007            CLRD    7(R2)                   ; TEST INSTRUCTION
11552 055500  170203                    STFPS   R3                      ;GET FPS
11553 055502  020227  003163            CMP     R2,#TSTLOC+1                    ;IS R2 CORRECT
11554 055506  001403                    BEQ     1$                      ;YES GO ON
11555 055510  004737  140132            CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11556 055514  104003                    ERROR   +3                      ;FPP ERROR
11557                                                                   ;NO GO TO ERROR
11558 055516  012702  003162    1$:     MOV     #TSTLOC,R2                      ;SETUP POINTER TO DATA
11559 055522  012701  000004            MOV     #4,R1                   ;INIT COUNTER
11560 055526  022227  177777    2$:     CMP     (R2)+,#177777           ;IS DATA CORRECT
11561 055532  001403                    BEQ     3$                      ;YES GO ON
11562 055534  004737  140132            CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11563 055540  104003                    ERROR   +3                      ;FPP ERROR
11564                                                                   ;NO GO TO ERROR
11565 055542  077107    3$:     SOB     R1,2$                   ;ARE WE DONE
11566 055544  012701  000004            MOV     #4,R1                   ;INIT COUNTER
11567 055550  022227  000000    4$:     CMP     (R2)+,#0                ;IS DATA CORRECT
11568 055554  001403                    BEQ     5$                      ;YES GO ON
11569 055556  004737  140132            CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11570 055562  104003                    ERROR   +3                      ;FPP ERROR
11571                                                                   ;NO GO TO ERROR
11572 055564  077107    5$:     SOB     R1,4$                   ;ARE WE DONE
11573 055566  020327  000204            CMP     R3,#204                 ;IS FPS CORRECT
11574 055572  001403                    BEQ     6$                      ;YES GO ON
11575 055574  004737  140132            CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11576 055600  104003                    ERROR   +3                      ;FPP ERROR
11577                                                                   ;NO GO TO ERROR
11578 055602  012637  000004    6$:     MOV     (SP)+,@#4               ;RESTORE VECTOR
11579 055606  000167  000010            JMP     FIN20
11580
11581 055612  004737  140132    TSF20:  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
```

L16

FLOATING POINT TESTS

```
11582 055616  104003                          ERROR   +3              ;FPP ERROR
11583                                                                 ;ODD ADDRESS TRAP
11584 055620  000006                          RTT                             ;RETURN
11585                                 ;
11586 055622  000240                  FIN20:  NOP
11587                                 ;
11590                                 ;
11591                                 ;-----------  -----------  -----
11592                                 ;TEST  FDST SOP MODE 7
11593                                 ;
11594                                 ;
11595 055624                          TSFP21:
11596                                 ;
11597 055624  005037  140054                  CLR     @#TRPFLG                ;CLEAR TRAP FLAG
11598 055630  013746  000004                  MOV     @#4,-(SP)               ;SAVE TIMEOUT VECTOR
11599 055634  012737  056032  000004          MOV     #TSF21,@#4              ;SETUP NEW VECTOR
11600 055642  012702  000200                  MOV     #200,R2                 ;SETUP TO LOAD FPS
11601 055646  170102                          LDFPS   R2                      ;SET FD=1
11602 055650  012705  000010                  MOV     #8.,R5                  ;INIT COUNTER
11603 055654  012704  003162                  MOV     #TSTLOC,R4                      ;SETUP POINTER TO TEST LOCATION
11604 055660  012724  177777          100$:   MOV     #177777,(R4)+   ;INIT TEST LOCATION
11605 055664  077503                          SOB     R5,100$                 ;ARE WE DONE
11606 055666  012737  003162  003172          MOV     #TSTLOC,@#TSTLOC+10             ;INIT TEST LOCATION
11607 055674  012702  003165                  MOV     #TSTLOC+3,R2            ;SETUP POINTER TO DATA
11608 055700  170472  000005                  CLRD    @5(R2)                  ; TEST INSTRUCTION
11609 055704  170203                          STFPS   R3                      ;GET FPS
11610 055706  020227  003165                  CMP     R2,#TSTLOC+3                    ;IS R2 CORRECT
11611 055712  001403                          BEQ     1$                      ;YES GO ON
11612 055714  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11613 055720  104003                          ERROR   +3              ;FPP ERROR
11614                                                                 ;NO GO TO ERROR
11615 055722  012702  003162          1$:     MOV     #TSTLOC,R2                      ;SETUP POINTER TO DATA
11616 055726  012701  000004                  MOV     #4,R1                   ;INIT COUNTER
11617 055732  022227  000000          2$:     CMP     (R2)+,#0                ;IS DATA CORRECT
11618 055736  001403                          BEQ     3$                      ;YES GO ON
11619 055740  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11620 055744  104003                          ERROR   +3              ;FPP ERROR
11621                                                                 ;NO GO TO ERROR
11622 055746  077107                  3$:     SOB     R1,2$                   ;ARE WE DONE
11623 055750  022227  003162                  CMP     (R2)+,#TSTLOC           ;IS DATA CORRECT
11624 055754  001403                          BEQ     4$                      ;YES GO ON
11625 055756  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11626 055762  104003                          ERROR   +3              ;FPP ERROR
11627                                                                 ;NO GO TO ERROR
11628 055764  012701  000003          4$:     MOV     #3,R1                   ;INIT COUNTER
11629 055770  022227  177777          5$:     CMP     (R2)+,#177777           ;IS DATA CORRECT
11630 055774  001403                          BEQ     6$                      ;YES GO ON
11631 055776  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11632 056002  104003                          ERROR   +3              ;FPP ERROR
11633                                                                 ;NO GO TO ERROR
11634 056004  077107                  6$:     SOB     R1,5$                   ;ARE WE DONE
11635 056006  020327  000204                  CMP     R3,#204                 ;CHECK FPS
11636 056012  001403                          BEQ     7$                      ;OK GO ON
11637 056014  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11638 056020  104003                          ERROR   +3              ;FPP ERROR
11639                                                                 ;NO GO TO ERROR
11640 056022  012637  000004          7$:     MOV     (SP)+,@#4               ;RESTORE VECTOR
```

M16

FLOATING POINT TESTS

```
11641 056026  000167  000010                  JMP     FIN21
11642
11643 056032  004737  140132          TSF21:  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11644 056036  104003                          ERROR   +3                      ;FPP ERROR
11645                                                                         ;ODD ADDRESS TRAP
11646 056040  000006                          RTT                             ;RETURN
11647
11648 056042  000240          FIN21:  NOP
11649
11652                                 ;
11653                                 ;-- ------------------- --------
11654                                 ;TEST  FDST SOP MODE 3 GR7
11655                                 ;
11656                                 ;
11657 056044                   TSFP22:
11658                                 ;
11659 056044  005037  140054                  CLR     @#TRPFLG                ;CLEAR TRAP FLAG
11660 056050  013746  000004                  MOV     @#4,-(SP)               ;SAVE TIME OUT VECTOR
11661 056054  012737  056216  000004          MOV     #TSF22,@#4              ;SETUP NEW VECTOR
11662 056062  012702  000200                  MOV     #200,R2                 ;SETUP TO LOAD FPS
11663 056066  170102                          LDFPS   R2                      ;SET FD=1
11664 056070  012705  000010                  MOV     #8.,R5                  ;INIT COUNTER
11665 056074  012704  003162                  MOV     #TSTLOC,R4              ;SETUP POINTER TO TEST LOCATION
11666 056100  012724  177777          100$:   MOV     #177777,(R4)+           ;INIT TEST LOCATION
11667 056104  077503                          SOB     R5,100$                 ;ARE WE DONE
11668 056106  012737  003172  003162          MOV     #TSTLOC+10,@#TSTLOC             ;INIT TEST LOCATION
11669 056114  170437  003162                  CLRD    @#TSTLOC                        ; TEST INSTRUCTION
11670 056120  170203                          STFPS   R3                      ;GET FPS
11671 056122  012702  003162                  MOV     #TSTLOC,R2                      ;SETUP POINTER TO DATA
11672 056126  012701  000004                  MOV     #4,R1                   ;INIT COUNTER
11673 056132  022227  000000          1$:     CMP     (R2)+,#0                ;IS DATA CORRECT
11674 056136  001403                          BEQ     2$                      ;YES GO ON
11675 056140  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11676 056144  104003                          ERROR   +3                      ;FPP ERROR
11677                                                                         ;NO GO TO ERROR
11678 056146  077107          2$:     SOB     R1,1$                           ;ARE WE DONE
11679 056150  012701  000004                  MOV     #4,R1                   ;INIT COUNTER
11680 056154  022227  177777          3$:     CMP     (R2)+,#177777           ;IS DATA CORRECT
11681 056160  001403                          BEQ     4$                      ;YES GO ON
11682 056162  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11683 056166  104003                          ERROR   +3                      ;FPP ERROR
11684                                                                         ;NO GO TO ERROR
11685 056170  077107          4$:     SOB     R1,3$                           ;ARE WE DONE
11686 056172  020327  000204                  CMP     R3,#204                 ;CHECK FPS
11687 056176  001403                          BEQ     5$                      ;OK GO ON
11688 056200  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11689 056204  104003                          ERROR   +3                      ;FPP ERROR
11690                                                                         ;NO GO TO ERROR
11691 056206  012637  000004          5$:     MOV     (SP)+,@#4               ;RESTORE VECTOR
11692 056212  000167  000010                  JMP     FIN22
11693
11694 056216  004737  140132          TSF22:  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11695 056222  104003                          ERROR   +3                      ;FPP ERROR
11696                                                                         ;ODD ADDRESS TRAP
11697 056224  000006                          RTT                             ;RETURN
11698
11699 056226  000240          FIN22:  NOP
```

FLOATING POINT TESTS

```
11700                                   ;
11703                                   ;
11704                                   ;-------- - ---- - ------ ------ -
11705                                   ;TEST  FDST SOP MODE 6 GR7
11706                                   ;
11707                                   ;
11708 056230                            TSFP23:
11709                                   ;
11710 056230  005037  140054                    CLR     @#TRPFLG                ;CLEAR TRAP FLAG
11711 056234  013746  000004                    MOV     @#4,-(SP)               ;SAVE TIMEOUT VECTOR
11712 056240  012737  056352  000004          MOV     #TSF23,@#4              ;SETUP NEW VECTOR
11713 056246  012702  000200                    MOV     #200,R2                 ;SETUP TO LOAD FPS
11714 056252  170102                            LDFPS   R2                      ;SET FD=1
11715 056254  012705  000004                    MOV     #4,R5                   ;INIT COUNTER
11716 056260  012704  003162                    MOV     #TSTLOC,R4              ;SETUP POINTER TO TEST LOCATION
11717 056264  012724  177777            100$:   MOV     #177777,(R4)+           ;INIT TEST LOCATION
11718 056270  077503                            SOB     R5,100$                 ;ARE WE DONE
11719 056272  170467  124664                    CLRD    TSTLOC                  ; TEST INSTRUCTION
11720 056276  170203                            STFPS   R3                      ;GET FPS
11721 056300  012701  000004                    MOV     #4,R1                   ;INIT COUNTER
11722 056304  012702  003162                    MOV     #TSTLOC,R2                      ;SETUP POINTER TO DATA
11723 056310  022227  000000            1$:     CMP     (R2)+,#0                ;IS DATA CORRECT
11724 056314  001403                            BEQ     2$                      ;YES GO ON
11725 056316  004737  140132                    CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11726 056322  104003                            ERROR   +3              ;FPP ERROR
11727                                                                   ;NO GO TO ERROR
11728 056324  077107            2$:     SOB     R1,1$                           ;ARE WE DONE
11729 056326  020327  000204                    CMP     R3,#204                 ;CHECK FPS
11730 056332  001403                            BEQ     3$                      ;OK GO ON
11731 056334  004737  140132                    CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11732 056340  104003                            ERROR   +3              ;FPP ERROR
11733                                                                   ;NO GO TO ERROR
11734 056342  012637  000004            3$:     MOV     (SP)+,@#4               ;RESTORE VECTOR
11735 056346  000167  000010                    JMP     FIN23
11736                                   ;
11737 056352  004737  140132            TSF23:  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11738 056356  104003                            ERROR   +3              ;FPP ERROR
11739                                                                   ;ODD ADDRESS TRAP
11740 056360  000006                            RTT                             ;
11741 056362  000240            FIN23:  NOP
11742                                   ;
11745                                   ;
11746                                   ;---------------------------------
11747                                   ;TEST  FDST SOP MODE 7 GR7
11748                                   ;
11749                                   ;
11750 056364                            TSFP24:
11751                                   ;
11752 056364  005037  140054                    CLR     @#TRPFLG                ;CLEAR TRAP FLAG
11753 056370  013746  000004                    MOV     @#4,-(SP)               ;SAVE TIMEOUT VECTOR
11754 056374  012737  056552  000004          MOV     #TSF24,@#4              ;SETUP NEW VECTOR
11755 056402  012702  000200                    MOV     #200,R2                 ;SETUP TO LOAD FPS
11756 056406  170102                            LDFPS   R2                      ;SET FD=1
11757 056410  012705  000010                    MOV     #8.,R5          ;INIT COUNTER
11758 056414  012704  003162                    MOV     #TSTLOC,R4              ;SETUP TEST LOCATION POINTER
11759 056420  012724  177777            100$:   MOV     #177777,(R4)+           ;INIT TEST LOCATION
11760 056424  077503                            SOB     R5,100$                 ;ARE WE DONE
```

FLOATING POINT TESTS

```
11761 056426  012737  003162  003172          MOV     #TSTLOC,@#TSTLOC+10             ;INIT TEST LOCATION
11762 056434  170477  124532                   CLRD    @TSTLOC+10                      ; TEST INSTRUCTION
11763 056440  170203                           STFPS   R3                      ;GET FPS
11764 056442  012702  003162                   MOV     #TSTLOC,R2                      ;SETUP POINTER TO DATA
11765 056446  012701  000004                   MOV     #4,R1                   ;INIT COUNTER
11766 056452  022227  000000           1$:     CMP     (R2)+,#0                ;IS DATA CORRECT
11767 056456  001403                           BEQ     2$                      ;YES GO ON
11768 056460  004737  140132                   CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11769 056464  104003                           ERROR   +3              ;FPP ERROR
11770                                                                  ;NO GO TO ERROR
11771 056466  077107           2$:     SOB     R1,1$                   ;ARE WE DONE
11772 056470  022227  003162                   CMP     (R2)+,#TSTLOC           ;IS DATA CORRECT
11773 056474  001403                           BEQ     3$                      ;YES GO ON
11774 056476  004737  140132                   CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11775 056502  104003                           ERROR   +3              ;FPP ERROR
11776                                                                  ;NO GO TO ERROR
11777 056504  012701  000003           3$:     MOV     #3,R1                   ;INIT COUNTER
11778 056510  022227  177777           4$:     CMP     (R2)+,#177777           ;IS DATA CORRECT
11779 056514  001403                           BEQ     5$                      ;YES GO ON
11780 056516  004737  140132                   CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11781 056522  104003                           ERROR   +3              ;FPP ERROR
11782                                                                  ;NO GO TO ERROR
11783 056524  077107           5$:     SOB     R1,4$                   ;ARE WE DONE
11784 056526  020327  000204                   CMP     R3,#204                 ;CHECK FPS
11785 056532  001403                           BEQ     6$                      ;OK GO ON
11786 056534  004737  140132                   CALL    @#DFTFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11787 056540  104003                           ERROR   +3              ;FPP ERROR
11788                                                                  ;NO GO TO ERROR
11789 056542  012637  000004           6$:     MOV     (SP)+,@#4               ;RESTORE VECTOR
11790 056546  000167  000010                   JMP     FIN24
11791
11792 056552  004737  140132           TSF24:  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11793 056556  104003                           ERROR   +3              ;FPP ERROR
11794                                                                  ;ODD ADDRESS TRAP
11795 056560  000006                           RTT                             ;
11796
11797 056562  000240           FIN24:  NOP
11798                          ;
11801                          ;
11802                          ;--- ----------------------------
11803                          ;TEST  CLRF
11804                          ;
11805                          ;
11806 056564                   TSFP25:
11807                          ;
11808 056564  005037  140054                   CLR     @#TRPFLG                ;CLEAR TRAP FLAG
11809 056570  005002                           CLR     R2                      ;SETUP TO LOAD FPS
11810 056572  170102                           LDFPS   R2                      ;SET FD=0
11811 056574  012705  000004                   MOV     #4,R5                   ;INIT COUNTER
11812 056600  012704  003162                   MOV     #TSTLOC,R4              ;SETUP POINTER TO TEST LOCATION
11813 056604  012724  177777           100$:   MOV     #177777,(R4)+           ;INIT TEST LOCATION
11814 056610  077503                           SOB     R5,100$                 ;ARE WE DONE
11815 056612  012702  003162                   MOV     #TSTLOC,R2                      ;SETUP POINTER TO DATA
11816 056616  170422                           CLRF    (R2)+                   ; TEST INSTRUCTION
11817 056620  170203                           STFPS   R3                      ;GET FPS
11818 056622  020227  003166                   CMP     R2,#TSTLOC+4                    ;IS R2 CORRECT
11819 056626  001403                           BEQ     1$                      ;YES GO ON
```

FLOATING POINT TESTS

```
11820 056630  004737  140132                CALL    @#DETFPA            ;DETERMINE FLOATING POINT FAULT. $$$
11821 056634  104003                        ERROR   +3                  ;FPP ERROR
11822                                                                     ;NO GO TO ERROR
11823 056636  012702  003162          1$:   MOV     #TSTLOC,R2                  ;SETUP POINTER TO DATA
11824 056642  012701  000002                MOV     #2,R1               ;INIT COUNTER
11825 056646  022227  000000          2$:   CMP     (R2)+,#0            ;IS DATA CORRECT
11826 056652  001403                        BEQ     3$                  ;YES GO ON
11827 056654  004737  140132                CALL    @#DETFPA            ;DETERMINE FLOATING POINT FAULT. $$$
11828 056660  104003                        ERROR   +3                  ;FPP ERROR
11829                                                                     ;NO GO TO ERROR
11830 056662  077107          3$:   SOB     R1,2$               ;ARE WE DONE
11831 056664  012701  000002                MOV     #2,R1               ;INIT COUNTER
11832 056670  022227  177777          4$:   CMP     (R2)+,#177777       ;IS DATA CORRECT
11833 056674  001403                        BEQ     5$                  ;YES GO ON
11834 056676  004737  140132                CALL    @#DETFPA            ;DETERMINE FLOATING POINT FAULT. $$$
11835 056702  104003                        ERROR   +3                  ;FPP ERROR
11836                                                                     ;NO GO TO ERROR
11837 056704  077107          5$:   SOB     R1,4$               ;ARE WE DONE
11838 056706  020327  000004                CMP     R3,#4               ;CHECK FPS
11839 056712  001403                        BEQ     6$                  ;OK GO ON
11840 056714  004737  140132                CALL    @#DETFPA            ;DETERMINE FLOATING POINT FAULT. $$$
11841 056720  104003                        ERROR   +3                  ;FPP ERROR
11842                                                                     ;NO GO TO ERROR
11843 056722                          6$:
11844                                 ;
11847                                 ;
11848                                 ;-- - ------ --------------------
11849                                 ;TEST  TSTF AND TSTD
11850                                 ;
11851                                 ;
11852 056722                          TSFP26:
11853                                 ;
11854 056722  005037  140054                CLR     @#TRPFLG                    ;CLEAR TRAP FLAG
11855 056726  005004                        CLR     R4                          ;SETUP TO LOAD FPS
11856 056730  170104                        LDFPS   R4                          ;SET FD=0
11857 056732  170567  000300                TSTF    TS26D0                      ; TEST INSTRUCTION
11858 056736  170203                        STFPS   R3                          ;GET FPS
11859 056740  020327  000004                CMP     R3,#4                       ;CHECK FPS
11860 056744  001403                        BEQ     1$                          ;OK GO ON
11861 056746  004737  140132                CALL    @#DETFPA                    ;DETERMINE FLOATING POINT FAULT. $$$
11862 056752  104003                        ERROR   +3                  ;FPP ERROR
11863                                                                     ;NO GO TO ERROR
11864 056754  012704  057236          1$:   MOV     #TS26D0,R4                  ;SETUP POINTERS TO DATA
11865 056760  012702  057266                MOV     #TS26D3,R2                  ;
11866 056764  004767  000224                JSR     PC,CHEC26                   ;CHECK IF DATA IS CORRECT
11867 056770  170537  057246                TSTF    @#TS26D1                    ; TEST INSTRUCTION
11868 056774  170203                        STFPS   R3                          ;GET FPS
11869 056776  020327  000010                CMP     R3,#10                      ;CHECK FPS
11870 057002  001403                        BEQ     2$                          ;OK GO ON
11871 057004  004737  140132                CALL    @#DETFPA                    ;DETERMINE FLOATING POINT FAULT. $$$
11872 057010  104003                        ERROR   +3                  ;FPP ERROR
11873                                                                     ;NO GO TO ERROR
11874 057012  012704  057246          2$:   MOV     #TS26D1,R4                  ;SETUP POINTERS TO DATA
11875 057016  012702  057276                MOV     #TS26D4,R2                  ;
11876 057022  004767  000166                JSR     PC,CHEC26                   ;CHECK IF DATA IS CORRECT
11877 057026  170567  000224                TSTF    TS26D2                      ; TEST INSTRUCTION
11878 057032  170203                        STFPS   R3                          ;GET FPS
```

FLOATING POINT TESTS

```
11879 057034  020327  000000                CMP     R3,#0                   ;CHECK FPS
11880 057040  001403                        BEQ     3$                      ;OK GO ON
11881 057042  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11882 057046  104003                        ERROR   +3                    ;FPP ERROR
11883                                                                       ;NO GO TO ERROR
11884 057050  012704  057256          3$:   MOV     #TS26D2,R4              ;SETUP POINTERS TO DATA
11885 057054  012702  057306                MOV     #TS26D5,R2              ;
11886 057060  004767  000130                JSR     PC,CHEC26               ;CHECK IF DATA IS CORRECT
11887 057064  012704  000200                MOV     #200,R4                 ;SETUP TO LOAD FPS
11888 057070  170104                        LDFPS   R4                      ;SET FD=1
11889 057072  170537  057236                TSTD    @#TS26D0                ; TEST INSTRUCTION
11890 057076  170203                        STFPS   R3                      ;GET FPS
11891 057100  020327  000204                CMP     R3,#204                 ;CHECK FPS
11892 057104  001403                        BEQ     4$                      ;OK GO ON
11893 057106  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11894 057112  104003                        ERROR   +3                    ;FPP ERROR
11895                                                                       ;NO GO TO ERROR
11896 057114  012704  057236          4$:   MOV     #TS26D0,R4              ;SETUP POINTERS TO DATA
11897 057120  012702  057266                MOV     #TS26D3,R2              ;
11898 057124  004767  000064                JSR     PC,CHEC26               ;CHECK IF DATA IS CORRECT
11899 057130  170567  000112                TSTD    TS26D1                  ; TEST INSTRUCTION
11900 057134  170203                        STFPS   R3                      ;GET FPS
11901 057136  020327  000210                CMP     R3,#210                 ;CHECK FPS
11902 057142  001403                        BEQ     5$                      ;OK GO ON
11903 057144  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11904 057150  104003                        ERROR   +3                    ;FPP ERROR
11905                                                                       ;NO GO TO ERROR
11906 057152  012704  057246          5$:   MOV     #TS26D1,R4              ;SETUP POINTERS TO DATA
11907 057156  012702  057276                MOV     #TS26D4,R2              ;
11908 057162  004767  000026                JSR     PC,CHEC26               ;CHECK IF DATA IS CORRECT
11909 057166  170567  000064                TSTD    TS26D2                  ; TEST INSTRUCTION
11910 057172  170203                        STFPS   R3                      ;GET FPS
11911 057174  020327  000200                CMP     R3,#200                 ;CHECK FPS
11912 057200  001403                        BEQ     6$                      ;OK GO ON
11913 057202  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11914 057206  104003                        ERROR   +3                    ;FPP ERROR
11915                                                                       ;NO GO TO ERROR
11916 057210                          6$:
11917 057210  000167  000102                JMP     FIN26
11918                                       ;
11919 057214  012701  000004          CHEC26: MOV     #4,R1                   ;INIT COUNTER
11920 057220  022422                1$:   CMP     (R4)+,(R2)+             ;IS DATA CORRECT
11921 057222  001403                        BEQ     2$                      ;YES GO ON
11922 057224  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11923 057230  104003                        ERROR   +3                    ;FPP ERROR
11924                                                                       ;NO GO TO ERROR
11925 057232  077106                2$:   SOB     R1,1$                   ;ARE WE DONE
11926 057234  000207                        RTS     PC                      ;RETURN
11927                                       ;
11928 057236  000177                TS26D0: .WORD   177
11929 057240  177777                        .WORD   177777
11930 057242  177777                        .WORD   177777
11931 057244  177777                        .WORD   177777
11932 057246  177777                TS26D1: .WORD   177777
11933 057250  000000                        .WORD   0
11934 057252  000000                        .WORD   0
11935 057254  000000                        .WORD   0
```

```
FLOATING POINT TESTS

11936 057256  077777                  TS26D2: .WORD   77777
11937 057260  000000                          .WORD   0
11938 057262  000000                          .WORD   0
11939 057264  000000                          .WORD   0
11940 057266  000177                  TS26D3: .WORD   177
11941 057270  177777                          .WORD   177777
11942 057272  177777                          .WORD   177777
11943 057274  177777                          .WORD   177777
11944 057276  177777                  TS26D4: .WORD   177777
11945 057300  000000                          .WORD   0
11946 057302  000000                          .WORD   0
11947 057304  000000                          .WORD   0
11948 057306  077777                  TS26D5: .WORD   77777
11949 057310  000000                          .WORD   0
11950 057312  000000                          .WORD   0
11951 057314  000000                          .WORD   0
11952 057316  000240                  FIN26:  NOP
11953                                 ;
11956                                 ;
11957                                 ; ---------------------- ------  --
11958                                 ;TEST  ABSF
11959                                 ;
11960                                 ;
11961 057320                          TSFP27:
11962                                 ;
11963 057320  005037  140054                  CLR     @#TRPFLG                ;CLEAR TRAP FLAG
11964 057324  005005                          CLR     R5                      ;SETUP TO LOAD FPS
11965 057326  170105                          LDFPS   R5                      ;SET FD=0
11966 057330  012701  000014                  MOV     #12.,R1                 ;INIT COUNTER
11967 057334  012704  003162                  MOV     #TSTLOC,R4                      ;SETUP POINTER TO TEST LOCATION
11968 057340  012703  057600                  MOV     #TS27D0,R3              ;SETUP POINTER TO TEST VALUE
11969 057344  012324                  100$:   MOV     (R3)+,(R4)+             ;INIT TEST LOCATION
11970 057346  077102                          SOB     R1,100$                 ;ARE WE DONE
11971 057350  012705  003162                  MOV     #TSTLOC,R5                      ;SETUP POINTER TO DATA
11972 057354  170615                          ABSF    (R5)                    ; TEST INSTRUCTION
11973 057356  170203                          STFPS   R3                      ;GET FPS
11974 057360  020527  003162                  CMP     R5,#TSTLOC                      ;IS R5 CORRECT
11975 057364  001403                          BEQ     1$                      ;YES GO ON
11976 057366  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11977 057372  104003                          ERROR   +3                      ;FPP ERROR
11978                                                                         ;NO GO TO ERROR
11979 057374  012702  057630          1$:     MOV     #TS27D3,R2                      ;SETUP POINTER TO DATA
11980 057400  004767  000152                  JSR     PC,CHEC27               ;CHECK IF DATA IS CORRECT
11981 057404  020327  000000                  CMP     R3,#0                   ;CHECK FPS
11982 057410  001403                          BEQ     2$                      ;OK GO ON
11983 057412  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11984 057416  104003                          ERROR   +3                      ;FPP ERROR
11985                                                                         ;NO GO TO ERROR
11986 057420  012705  003172          2$:     MOV     #TSTLOC+10,R5                   ;SETUP POINTER TO DATA
11987 057424  170625                          ABSF    (R5)+                   ; TEST INSTRUCTION
11988 057426  170203                          STFPS   R3                      ;GET FPS
11989 057430  020527  003176                  CMP     R5,#TSTLOC+14                   ;IS R5 CORRECT
11990 057434  001403                          BEQ     3$                      ;YES GO ON
11991 057436  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
11992 057442  104003                          ERROR   +3                      ;FPP ERROR
11993                                                                         ;NO GO TO ERROR
11994 057444  012705  003172          3$:     MOV     #TSTLOC+10,R5                   ;SETUP POINTER TO DATA
```

FLOATING POINT TESTS

```
11995 057450  012702  057640                MOV     #TS27D4,R2              ;
11996 057454  004767  000076                JSR     PC,CHEC27               ;CHECK IF DATA IS CORRECT
11997 057460  020327  000000                CMP     R3,#0                   ;CHECK FPS
11998 057464  001403                        BEQ     4$                      ;OK GO ON
11999 057466  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12000 057472  104003                        ERROR   +3                  ;FPP ERROR
12001                                                                   ;NO GO TO ERROR
12002 057474  012705  003162        4$:     MOV     #TSTLOC,R5                      ;SETUP POINTER TO DATA
12003 057500  170665  000020                ABSF    20(R5)                  ; TEST INSTRUCTION
12004 057504  170203                        STFPS   R3                      ;GET FPS
12005 057506  020527  003162                CMP     R5,#TSTLOC                      ;IS R5 CORRECT
12006 057512  001403                        BEQ     5$                      ;YES GO ON
12007 057514  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12008 057520  104003                        ERROR   +3                  ;FPP ERROR
12009                                                                   ;NO GO TO ERROR
12010 057522  012705  003202        5$:     MOV     #TSTLOC+20,R5                   ;SETUP POINTERS TO DATA
12011 057526  012702  057650                MOV     #TS27D5,R2              ;
12012 057532  004767  000020                JSR     PC,CHEC27               ;CHECK IF DATA IS CORRECT
12013 057536  020327  000004                CMP     R3,#4                   ;CHECK FPS
12014 057542  001403                        BEQ     6$                      ;OK GO ON
12015 057544  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12016 057550  104003                        ERROR   +3                  ;FPP ERROR
12017                                                                   ;NO GO TO ERROR
12018 057552                        6$:
12019 057552  000167  000102                JMP     FIN27
12020                                       ;
12021 057556  012701  000004        CHEC27: MOV     #4,R1                   ;INIT COUNTER
12022 057562  022522                1$:     CMP     (R5)+,(R2)+             ;IS DATA CORRECT
12023 057564  001403                        BEQ     2$                      ;YES GO ON
12024 057566  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12025 057572  104003                        ERROR   +3                  ;FPP ERROR
12026                                                                   ;NO GO TO ERROR
12027 057574  077106                2$:     SOB     R1,1$                   ;ARE WE DONE
12028 057576  000207                        RTS     PC                      ;RETURN
12029                                       ;
12030 057600  177777        TS27D0: .WORD   177777
12031 057602  177777                .WORD   177777
12032 057604  177777                .WORD   177777
12033 057606  177777                .WORD   177777
12034 057610  000377        TS27D1: .WORD   377
12035 057612  175436                .WORD   175436
12036 057614  136477                .WORD   136477
12037 057616  000001                .WORD   1
12038 057620  000177        TS27D2: .WORD   177
12039 057622  175436                .WORD   175436
12040 057624  136477                .WORD   136477
12041 057626  000001                .WORD   1
12042 057630  077777        TS27D3: .WORD   77777
12043 057632  177777                .WORD   177777
12044 057634  177777                .WORD   177777
12045 057636  177777                .WORD   177777
12046 057640  000377        TS27D4: .WORD   377
12047 057642  175436                .WORD   175436
12048 057644  136477                .WORD   136477
12049 057646  000001                .WORD   1
12050 057650  000000        TS27D5: .WORD   0
12051 057652  000000                .WORD   0
```

F_OATING POINT TESTS

```
12052 057654  136477                        .WORD   136477
12053 057656  000001                        .WORD   1
12054 057660  000240                FIN27:  NOP
12055                               ;
12058                               ;
12059                               ;---------- -----    -- --- ----
12060                               ;TEST  ABSD
12061                               ;
12062                               ;
12063 057662                        TSFP30:
12064                               ;
12065 057662  005037  140054                CLR     @#TRPFLG                ;CLEAR TRAP FLAG
12066 057666  012705  000200                MOV     #200,R5                 ;SETUP TO LOAD FPS
12067 057672  170105                        LDFPS   R5                      ;SET FD=1
12068 057674  012701  000014                MOV     #12.,R1                 ;INIT COUNTER
12069 057700  012704  003162                MOV     #TSTLOC,R4                      ;SETUP POINTER TO TEST LOCATION
12070 057704  012703  060144                MOV     #TS30D0,R3              ;SETUP POINTER TO TEST VALUE
12071 057710  012324                100$:   MOV     (R3)+,(R4)+             ;INIT TEST LOCATION
12072 057712  077102                        SOB     R1,100$                 ;ARE WE DONE
12073 057714  012705  003162                MOV     #TSTLOC,R5                      ;SETUP POINTER TO DATA
12074 057720  170615                        ABSD    (R5)                    ; TEST INSTRUCTION
12075 057722  170203                        STFPS   R3                      ;GET FPS
12076 057724  020527  003162                CMP     R5,#TSTLOC                      ;IS R5 CORRECT
12077 057730  001403                        BEQ     1$                      ;YES GO ON
12078 057732  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12079 057736  104003                        ERROR   +3              ;FPP ERROR
12080                                                               ;NO GO TO ERROR
12081 057740  012702  060174          1$:   MOV     #TS30D3,R2                      ;SETUP POINTER TO DATA
12082 057744  004767  000152                JSR     PC,CHEC30               ;CHECK IF DATA IS CORRECT
12083 057750  020327  000200                CMP     R3,#200                 ;CHECK FPS
12084 057754  001403                        BEQ     2$                      ;OK GO ON
12085 057756  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12086 057762  104003                        ERROR   +3              ;FPP ERROR
12087                                                               ;NO GO TO ERROR
12088 057764  012705  003172          2$:   MOV     #TSTLOC+10,R5                   ;SETUP POINTER TO DATA
12089 057770  170625                        ABSD    (R5)+                   ; TEST INSTRUCTION
12090 057772  170203                        STFPS   R3                      ;GET FPS
12091 057774  020527  003202                CMP     R5,#TSTLOC+20                   ;IS R5 CORRECT
12092 060000  001403                        BEQ     3$                      ;YES GO ON
12093 060002  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12094 060006  104003                        ERROR   +3              ;FPP ERROR
12095                                                               ;NO GO TO ERROR
12096 060010  012705  003172          3$:   MOV     #TSTLOC+10,R5                   ;SETUP POINTERS TO DATA
12097 060014  012702  060204                MOV     #TS30D4,R2              ;
12098 060020  004767  000076                JSR     PC,CHEC30               ;CHECK IF DATA IS CORRECT
12099 060024  020327  000200                CMP     R3,#200                 ;CHECK FPS
12100 060030  001403                        BEQ     4$                      ;OK GO ON
12101 060032  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12102 060036  104003                        ERROR   +3              ;FPP ERROR
12103                                                               ;NO GO TO ERROR
12104 060040  012705  003162          4$:   MOV     #TSTLOC,R5                      ;SETUP POINTER TO DATA
12105 060044  170665  000020                ABSD    20(R5)                  ; TEST INSTRUCTION
12106 060050  170203                        STFPS   R3                      ;GET FPS
12107 060052  020527  003162                CMP     R5,#TSTLOC                      ;IS R5 CORRECT
12108 060056  001403                        BEQ     5$                      ;YES GO ON
12109 060060  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12110 060064  104003                        ERROR   +3              ;FPP ERROR
```

FLOATING POINT TESTS

```
12111
12112 060066  012705  003202          5$:     MOV     #TSTLOC+20,R5           ;NO GO TO ERROR
12113 060072  012702  060214                  MOV     #TS30D5,R2                      ;SETUP POINTERS TO DATA
12114 060076  004767  000020                  JSR     PC,CHEC30               ;
12115 060102  020327  000204                  CMP     R3,#204                 ;CHECK IF DATA IS CORRECT
12116 060106  001403                          BEQ     6$                      ;CHECK FPS
12117 060110  004737  140132                  CALL    @#DETFPA                ;OK GO ON
12118 060114  104003                          ERROR   +3                      ;DETERMINE FLOATING POINT FAULT. $$$
12119                                                                         ;FPP ERROR
12120 060116                                  6$:                             ;NO GO TO ERROR
12121 060116  000167  000102                  JMP     FIN30
12122                                         ;
12123 060122  012701  000004          CHEC30: MOV     #4,R1                   ;INIT COUNTER
12124 060126  022522                  1$:     CMP     (R5)+,(R2)+             ;IS DATA CORRECT
12125 060130  001403                          BEQ     2$                      ;YES GO ON
12126 060132  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12127 060136  104003                          ERROR   +3                      ;FPP ERROR
12128                                                                         ;NO GO TO ERROR
12129 060140  077106                  2$:     SOB     R1,1$                   ;ARE WE DONE
12130 060142  000207                          RTS     PC                      ;RETURN
12131                                         ;
12132 060144  177777                  TS30D0: .WORD   177777
12133 060146  177777                          .WORD   177777
12134 060150  177777                          .WORD   177777
12135 060152  177777                          .WORD   177777
12136 060154  000377                  TS30D1: .WORD   377
12137 060156  175436                          .WORD   175436
12138 060160  136477                          .WORD   136477
12139 060162  000001                          .WORD   1
12140 060164  000177                  TS30D2: .WORD   177
12141 060166  175436                          .WORD   175436
12142 060170  136477                          .WORD   136477
12143 060172  000001                          .WORD   1
12144 060174  077777                  TS30D3: .WORD   77777
12145 060176  177777                          .WORD   177777
12146 060200  177777                          .WORD   177777
12147 060202  177777                          .WORD   177777
12148 060204  000377                  TS30D4: .WORD   377
12149 060206  175436                          .WORD   175436
12150 060210  136477                          .WORD   136477
12151 060212  000001                          .WORD   1
12152 060214  000000                  TS30D5: .WORD   0
12153 060216  000000                          .WORD   0
12154 060220  000000                          .WORD   0
12155 060222  000000                          .WORD   0
12156 060224  000240                  FIN30:  NOP
12157                                         ;
12160                                         ;
12161                                         ;---------------------------------
12162                                         ;TEST  FDST SOP MODE 2 GR7
12163                                         ;
12164 06022u                          TSFP31:
12165                                         ;
12166 060226  005037  140054                  CLR     @#TRPFLG                ;CLEAR TRAP FLAG
12167 060232  013746  000004                  MOV     @#4,-(SP)               ;SAVE TIMEOUT VECTOR
12168 060236  012737  060354  000004          MOV     #TSF31,@#4              ;SETUP NEW VECTOR
12169 060244  012702  000200                  MOV     #200,R2                 ;SETUP TO LOAD FPS
```

FLOATING POINT TESTS

```
12170 060250  170102                    LDFPS   R2                      ;SET FD=1
12171 060252  170527  000005    TSD31:  TSTD    #5                      ; TEST INSTRUCTION
12172 060256  000240                    NOP
12173 060260  000240                    NOP
12174 060262  000240                    NOP
12175 060264  170203                    STFPS   R3                      ;GET FPS
12176 060266  020327  000204            CMP     R3,#204                 ;CHECK FPS
12177 060272  001403                    BEQ     1$                      ;OK GO ON
12178 060274  004737  140132            CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12179 060300  104003                    ERROR   +3              ;FPP ERROR
12180                                                           ;NO GO TO ERROR
12181 060302  012702  060254    1$:     MOV     #TSD31+2,R2             ;SETUP POINTER TO DATA
12182 060306  022227  000005            CMP     (R2)+,#5                ;IS DATA CORRECT
12183 060312  001403                    BEQ     2$                      ;YES GO ON
12184 060314  004737  140132            CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12185 060320  104003                    ERROR   +3              ;FPP ERROR
12186                                                           ;NO GO TO ERROR
12187 060322  012701  000003    2$:     MOV     #3,R1                   ;INIT COUNTER
12188 060326  022227  000240    3$:     CMP     (R2)+,#240              ;IS DATA CORRECT
12189 060332  001403                    BEQ     4$                      ;YES GO ON
12190 060334  004737  140132            CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12191 060340  104003                    ERROR   +3              ;FPP ERROR
12192                                                           ;NO GO TO ERROR
12193 060342  077107    4$:     SOB     R1,3$                   ;ARE WE DONE
12194 060344  012637  000004            MOV     (SP)+,@#4               ;RESTORE VECTOR
12195 060350  000167  000010            JMP     FIN31
12196                                   ;
12197 060354  004737  140132    TSF31:  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12198 060360  104003                    ERROR   +3              ;FPP ERROR
12199                                                           ;ODD ADDRESS TRAP
12200 060362  000006                    RTT                             ;RETURN
12201                                   ;
12202 060364  000240    FIN31:  NOP
12203                                   ;
12206                                   ;
12207                                   ; ---------------------  - ---
12208                                   ;TEST  NEGF
12209                                   ;
12210                                   ;
12211 060366            TSFP32:
12212                                   ;
12213 060366  005037  140054            CLR     @#TRPFLG                ;CLEAR TRAP FLAG
12214 060372  005005                    CLR     R5                      ;SETUP TO LOAD FPS
12215 060374  170105                    LDFPS   R5                      ;SET FD=0
12216 060376  012701  000014            MOV     #12.,R1                 ;INIT COUNTER
12217 060402  012704  003162            MOV     #TSTLOC,R4                      ;SETUP POINTER TO TEST LOCATION
12218 060406  012703  060576            MOV     #TS32D0,R3              ;SETUP POINTER TO TEST VALUE
12219 060412  012324    100$:   MOV     (R3)+,(R4)+             ;INIT TEST LOCATION
12220 060414  077102                    SOB     R1,100$                 ;ARE WE DONE
12221 060416  170767  122540            NEGF    TSTLOC                  ; TEST INSTRUCTION
12222 060422  170203                    STFPS   R3                      ;GET FPS
12223 060424  012705  003162            MOV     #TSTLOC,R5                      ;SETUP POINTERS TO DATA
12224 060430  012702  060626            MOV     #TS32D3,R2              ;
12225 060434  004767  000114            JSR     PC,CHEC32               ;CHECK IF DATA IS CORRECT
12226 060440  020327  000000            CMP     R3,#0                   ;CHECK FPS
12227 060444  001403                    BEQ     1$                      ;YES GO ON
12228 060446  004737  140132            CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
```

-LOATING POINT TESTS

```
12229 060452  104003                        ERROR   +3              ;FPP ERROR
12230                                                               ;NO GO TO ERROR
12231 060454  170767  122512          1$:   NEGF    TSTLOC+10                 ; TEST INSTRUCTION
12232 060460  170203                        STFPS   R3                      ;GET FPS
12233 060462  012705  003172                MOV     #TSTLOC+10,R5                   ;SETUP POINTERS TO DATA
12234 060466  012702  060636                MOV     #TS32D4,R2              ;
12235 060472  004767  000056                JSR     PC,CHEC32               ;CHECK IF DATA IS CORRECT
12236 060476  020327  000010                CMP     R3,#10                  ;CHECK FPS
12237 060502  001403                        BEQ     2$                      ;OK GO ON
12238 060504  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12239 060510  104003                        ERROR   +3              ;FPP ERROR
12240                                                               ;NO GO TO ERROR
12241 060512  170767  122464          2$:   NEGF    TSTLOC+20                 ; TEST INSTRUCTION
12242 060516  170203                        STFPS   R3                      ;GET FPS
12243 060520  012705  003202                MOV     #TSTLOC+20,R5                   ;SETUP POINTERS TO DATA
12244 060524  012702  060646                MOV     #TS32D5,R2              ;
12245 060530  004767  000020                JSR     PC,CHEC32               ;CHECK IF DATA IS CORRECT
12246 060534  020327  000004                CMP     R3,#4                   ;CHECK FPS
12247 060540  001403                        BEQ     3$                      ;OK GO ON
12248 060542  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12249 060546  104003                        ERROR   +3              ;FPP ERROR
12250                                                               ;NO GO TO ERROR
12251 060550                          3$:
12252 060550  000167  000102                JMP     FIN32
12253                                       ;
12254 060554  012701  000004        CHEC32: MOV     #4,R1                   ;INIT COUNTER
12255 060560  022522                  1$:   CMP     (R5)+,(R2)+             ;IS DATA CORRECT
12256 060562  001403                        BEQ     2$                      ;YES GO ON
12257 060564  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12258 060570  104003                        ERROR   +3              ;FPP ERROR
12259                                                               ;NO GO TO ERROR
12260 060572  077106                  2$:   SOB     R1,1$                   ;ARE WE DONE
12261 060574  000207                        RTS     PC                      ;RETURN
12262                                       ;
12263 060576  170000        TS32D0: .WORD   170000
12264 060600  003541                .WORD   3541
12265 060602  177777                .WORD   177777
12266 060604  172710                .WORD   172710
12267 060606  070000        TS32D1: .WORD   70000
12268 060610  003541                .WORD   3541
12269 060612  177777                .WORD   177777
12270 060614  172710                .WORD   172710
12271 060616  000177                .WORD   177
12272 060620  100000                .WORD   100000
12273 060622  177777                .WORD   177777
12274 060624  177007                .WORD   177007
12275 060626  070000        TS32D3: .WORD   70000
12276 060630  003541                .WORD   3541
12277 060632  177777                .WORD   177777
12278 060634  172710                .WORD   172710
12279 060636  170000        TS32D4: .WORD   170000
12280 060640  003541                .WORD   3541
12281 060642  177777                .WORD   177777
12282 060644  172710                .WORD   172710
12283 060646  000000        TS32D5: .WORD   0
12284 060650  000000                .WORD   0
12285 060652  177777                .WORD   177777
```

FLOATING POINT TESTS

```
12286 060654  177007                        .WORD   177007
12287 060656  000240                FIN32:  NOP
12288                               ;
12291                               ;
12292                               ;---- ------------ ----- - -----
12293                               ;TEST  NEGD
12294                               ;
12295 060660                        TSFP33:
12296                               ;
12297 060660  005037  140054                CLR     @#TRPFLG                ;CLEAR TRAP FLAG
12298 060664  012705  000200                MOV     #200,R5                 ;SETUP TO LOAD FPS
12299 060670  170105                        LDFPS   R5                      ;SET FD=1
12300 060672  012701  000014                MOV     #12.,R1                 ;INIT COUNTER
12301 060676  012704  003162                MOV     #TSTLOC,R4                      ;SETUP POINTER TO TEST LOCATION
12302 060702  012703  061072                MOV     #TS33D0,R3              ;SETUP POINTER TO TEST VALUE
12303 060706  012324                100$:   MOV     (R3)+,(R4)+             ;INIT TEST LOCATION
12304 060710  077102                        SOB     R1,100$                 ;ARE WE DONE
12305 060712  170767  122244                NEGD    TSTLOC                  ; TEST INSTRUCTION
12306 060716  170203                        STFPS   R3                      ;GET FPS
12307 060720  012705  003162                MOV     #TSTLOC,R5                      ;SETUP POINTERS TO DATA
12308 060724  012702  061122                MOV     #TS33D3,R2              ;
12309 060730  004767  000114                JSR     PC,CHEC33               ;CHECK IF DATA IS CORRECT
12310 060734  020327  000200                CMP     R3,#200                 ;CHECK FPS
12311 060740  001403                        BEQ     1$                      ;OK GO ON
12312 060742  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12313 060746  104003                        ERROR   +3              ;FPP ERROR
12314                                                               ;NO GO TO ERROR
12315 060750  170767  122216          1$:   NEGD    TSTLOC+10                       ; TEST INSTRUCTION
12316 060754  170203                        STFPS   R3                      ;GET FPS
12317 060756  012705  003172                MOV     #TSTLOC+10,R5                   ;SETUP POINTERS TO DATA
12318 060762  012702  061132                MOV     #TS33D4,R2              ;
12319 060766  004767  000056                JSR     PC,CHEC33               ;CHECK IF DATA IS CORRECT
12320 060772  020327  000210                CMP     R3,#210                 ;CHECK FPS
12321 060776  001403                        BEQ     2$                      ;OK GO ON
12322 061000  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12323 061004  104003                        ERROR   +3              ;FPP ERROR
12324                                                               ;NO GO TO ERROR
12325 061006  170767  122170          2$:   NEGD    TSTLOC+20                       ; TEST INSTRUCTION
12326 061012  170203                        STFPS   R3                      ;GET FPS
12327 061014  012705  003202                MOV     #TSTLOC+20,R5                   ;SETUP POINTERS TO DATA
12328 061020  012702  061142                MOV     #TS33D5,R2              ;
12329 061024  004767  000020                JSR     PC,CHEC33               ;CHECK IF DATA IS CORRECT
12330 061030  020327  000204                CMP     R3,#204                 ;CHECK FPS
12331 061034  001403                        BEQ     3$                      ;OK GO ON
12332 061036  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12333 061042  104003                        ERROR   +3              ;FPP ERROR
12334                                                               ;NO GO TO ERROR
12335 061044                        3$:
12336 061044  000167  000102                JMP     FIN33
12337                               ;
12338 061050  012701  000004        CHEC33: MOV     #4,R1                   ,INIT COUNTER
12339 061054  022522                1$:     CMP     (R5)+,(R2)+             ;IS DATA CORRECT
12340 061056  001403                        BEQ     2$                      ;YES GO ON
12341 061060  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12342 061064  104003                        ERROR   +3              ;FPP ERROR
12343                                                               ;NO GO TO ERROR
12344 061066  077106                2$:     SOB     R1,1$                   ;ARE WE DONE
```

FLOATING POINT TESTS

```
12345 061070  000207                    RTS     PC                      ;RETURN
12346                                   ;
12347 061072  170000            TS33D0: .WORD   170000
12348 061074  003541                    .WORD   3541
12349 061076  177777                    .WORD   177777
12350 061100  172710                    .WORD   172710
12351 061102  070000            TS33D1: .WORD   70000
12352 061104  003541                    .WORD   3541
12353 061106  177777                    .WORD   177777
12354 061110  172710                    .WORD   172710
12355 061112  000177            TS33D2: .WORD   177
12356 061114  100000                    .WORD   100000
12357 061116  177777                    .WORD   177777
12358 061120  177007                    .WORD   177007
12359 061122  070000            TS33D3: .WORD   70000
12360 061124  003541                    .WORD   3541
12361 061126  177777                    .WORD   177777
12362 061130  172710                    .WORD   172710
12363 061132  170000            TS33D4: .WORD   170000
12364 061134  003541                    .WORD   3541
12365 061136  177777                    .WORD   177777
12366 061140  172710                    .WORD   172710
12367 061142  000000            TS33D5: .WORD   0
12368 061144  000000                    .WORD   0
12369 061146  000000                    .WORD   0
12370 061150  000000                    .WORD   0
12371 061152  000240            FIN33:  NOP
12372                                   ;
12375                                   ;
12376                                   ;--------------------------------
12377                                   ;TEST LDD MODE 0, ILLEGAL AC7
12378                                   ;
12379                                       ;
12380 061154                    MFSRCM0:
12381
12382                                   ;
12383 061154  012704  047600                    MOV     #47600,R4               ;SETUP FPP STATUS
12384 061160  170104                            LDFPS   R4                      ;LOAD FP STATUS
12385 061162  012702  003122                    MOV     #RECFEC,R2              ;POINT TO RECEIVED FEC MEMORY
12386 061166  172407            1$:     LDD     R7,AC0                  ;*TEST INSTRUCTION
12387                                                                   ;LOAD AC0 FROM ILLEGAL AC7
12388 061170  170201                            STFPS   R1                      ;SAVE FPP STATUS
12389 061172  022701  147600                    CMP     #147600,R1              ;VERIFY FER BIT SET
12390 061176  001403                            BEQ     2$                      ;BRANCH IF GOOD ERROR CONDITION
12391 061200  004737  140132                    CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12392 061204  104003                            ERROR   +3              ;FPP ERROR
12393                                                                   ;THE FER BIT DIDNT SET
12394 061206  170312            2$:     STST    (R2)                    ;SAVE FEC AND FEA
12395 061210  022722  000002                    CMP     #2,(R2)+                ;VERIFY FEC CONTENTS
12396 061214  001403                            BEQ     3$                      ;BRANCH IF GOOD
12397 061216  004737  140132                    CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12398 061222  104003                            ERROR   +3              ;FPP ERROR
12399                                                                   ;FEC NE 2 (OPCODE ERROR)
12400 061224  022722  061166            3$:     CMP     #1$,(R2)+               ;VERFIY FEC CONTENTS
12401 061230  001403                            BEQ     4$                      ;BRANCH FI GOOD
12402 061232  004737  140132                    CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12403 061236  104003                            ERROR   +3              ;FPP ERROR
```

FLOATING POINT TESTS

```
12404                                                          ;FEA NOT CORRECT ERROR ADDRESS
12405 061240                                4$:
12408                                       ;
12409                                       ;--- - -----     ---- - - -- -- --
12410                                       ;TEST LDD MODE2
12411                                       ;
12412 061240                                MLDDM2:
12413                                       ;
12414 061240  012701  003142                        MOV     #RECDST,R1              ;POINT TO RECEIVED DATA LOCATION
12415 061244  012704  003234                        MOV     #TAB1,R4                ;POINT TO GOOD DATA
12416 061250  012702  047750                        MOV     #47750,R2               ;LOAD GOOD STATUS
12417 061254  170102                                LDFPS   R2                      ;LOAD FPP STATUS - DOUBLE,ID
12418 061256  172424                                LDD     (R4)+,AC0               ;*TEST INSTRUCTION - MODE 2
12419 061260  170203                                STFPS   R3                      ;SAVE TEST FPP STATUS
12420 061262  174021                                STD     AC0,(R1)+               ;SAVE TEST RESULT MODE 2
12421 061264  020203                                CMP     R2,R3                   ;VERIFY FPP STATUS
12422 061266  001403                                BEQ     1$                      ;BRANCH IF GOOD
12423 061270  004737  140132                        CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12424 061274  104003                                ERROR   +3                      ;FPP ERROR
12425                                                                               ;BAD FPP STSTUS
12426 061276  022704  003244                1$:     CMP     #TAB1+10,R4                     ;VERFIY AUTO-INCR
12427 061302  001403                                BEQ     2$                      ;BRANCH IF GOOD
12428 061304  004737  140132                        CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12429 061310  104003                                ERROR   +3                      ;FPP ERROR
12430                                                                               ;BAD AUTO-INCR
12431 061312  012704  003234                2$:     MOV     #TAB1,R4                        ;POINT TO RECEIVED DATA
12432 061316  162701  000010                        SUB     #10,R1                  ;RETURN R1 TO PROPER VALUE
12433 061322  004767  056552                        JSR     R7,DATVER               ;VERFIY DATA FROM FPP
12434 061326  005767  121566                        TST     COUNT                   ;SEE IF COUNTER=0
12435 061332  001403                                BEQ     3$                      ;BRANCH IF GOOD COMPARE
12436 061334  004737  140132                        CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12437 061340  104003                                ERROR   +3                      ;FPP ERROR                      ;BAD DATA FROM FPP
12438 061342                                3$:
12439                                       ;
12442                                       ;
12443                                       ;------------------------- ------
12444                                       ;TEST LDD MODE 3
12445                                       ;
12446 061342                                MLDDM3:
12447                                       ;
12448 061342  012737  003142  003164                MOV     #RECDST,@#TSTLOC+2              ;POINT TO RECEIVED DATA LOCATION
12449 061350  012701  003164                        MOV     #TSTLOC+2,R1            ;SETUP STD IN MODE 3
12450 061354  012737  003244  003162                MOV     #TAB2,@#TSTLOC          ;POINT TO DATA TABLE
12451 061362  012704  003162                        MOV     #TSTLOC,R4                      ;POINT TO GOOD DATA
12452 061366  012702  047750                        MOV     #47750,R2               ;LOAD GOOD STATUS
12453 061372  170102                                LDFPS   R2                      ;LOAD FPP STATUS - DOUBLE,ID
12454 061374  172434                                LDD     @(R4)+,AC0                      ;*TEST INSTRUCTION - MODE 2
12455 061376  170203                                STFPS   R3                      ;SAVE TEST FPP STATUS
12456 061400  174031                                STD     AC0,@(R1)+                      ;SAVE TEST RESULT IN MODE 3
12457 061402  022703  047740                        CMP     #47740,R3                       ;VERIFY FPP STATUS
12458 061406  001403                                BEQ     1$                      ;BRANCH IF GOOD
12459 061410  004737  140132                        CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12460 061414  104003                                ERROR   +3                      ;FPP ERROR
12461                                                                               ;BAD FPP STSTUS
12462 061416  022704  003164                1$:     CMP     #TSTLOC+2,R4                    ;VERFIY AUTO INCR
12463 061422  001403                                BEQ     2$                      ;BAD AUTO-DEC ON LDD
12464 061424  004737  140132                        CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
```

FLOATING POINT TESTS

```
12465 061430  104003                      ERROR   +3              ;FPP ERROR
12466                                                             ;BAD AUTO-INC
12467 061432  022701  003166        2$:   CMP     #TSTLOC+4,R1                ; TEST STD AUTO-INC
12468 061436  001403                      BEQ     3$                  ;BRANCH IF GOOD
12469 061440  004737  140132              CALL    @#DETFPA            ;DETERMINE FLOATING POINT FAULT. $$$
12470 061444  104003                      ERROR   +3              ;FPP ERROR
12471                                                             ;BAD AUTO-INCR
12472 061446  012704  003244        3$:   MOV     #TAB2,R4            ;POINT TO RECEIVED DATA
12473 061452  012701  003142              MOV     #RECDST,R1          ;POINT TO RECEIVED DATA
12474 061456  004767  056416              JSR     R7,DATVER           ;VERFIY DATA FROM FPP
12475 061462  005767  121432              TST     COUNT               ;SEE IF COUNTER=0
12476 061466  001403                      BEQ     4$                  ;BRANCH IF GOOD COMPARE
12477 061470  004737  140132              CALL    @#DETFPA            ;DETERMINE FLOATING POINT FAULT. $$$
12478 061474  104003                      ERROR   +3              ;FPP ERROR
12479                                                             ;BAD DATA FROM FPP
12480 061476                        4$:
12481                                     ;
12484                                     ;
12485                                     ;--------------- --  ----   -----
12486                                     ;TEST LDF, STD MODE 4
12487                                     ;
12488 061476                        MLDDM4:
12489                                     ;
12490 061476  012701  003146              MOV     #RECDST+4,R1        ;POINT TO RECEIVED DATA LOCATION
12491 061502  012704  003260              MOV     #TAB3+4,R4          ;POINT TO GOOD DATA
12492 061506  012705  003314              MOV     #TAB6,R5            ;CLEAR OUT AC0
12493 061512  170127  000200              LDFPS   #200                ;SET TO DOUBLE
12494 061516  172415                      LDD     (R5),AC0                    ;AC0=0
12495 061520  012702  047550              MOV     #47550,R2           ;LOAD GOOD STATUS FLOATING
12496 061524  170102                      LDFPS   R2                  ;LOAD FPP STATUS - DOUBLE,ID
12497 061526  172444                      LDF     -(R4),AC0                   ;*TEST INSTRUCTION   MODE 4
12498 061530  170203                      STFPS   R3                  ;SAVE TEST FPP STATUS
12499 061532  012702  047750              MOV     #47750,R2           ;SET TO DOUBLE MODE
12500 061536  170102                      LDFPS   R2                  ;SET FPP TO DOUBLE
12501 061540  174041                      STD     AC0,-(R1)                   ;SAVE TEST RESULT
12502 061542  022703  047540              CMP     #47540,R3                   ;VERIFY FPP STATUS
12503 061546  001403                      BEQ     1$                  ;BRANCH IF GOOD
12504 061550  004737  140132              CALL    @#DETFPA            ;DETERMINE FLOATING POINT FAULT. $$$
12505 061554  104003                      ERROR   +3              ;FPP ERROR
12506                                                             ;BAD FPP STSTUS
12507 061556  022703  003254        1$:   CMP     #TAB3,R4                    ;VERFIY AUTO-DEC
12508 061562  001403                      BEQ     2$                  ;BRANCH IF GOOD
12509 061564  004737  140132              CALL    @#DETFPA            ;DETERMINE FLOATING POINT FAULT. $$$
12510 061570  104003                      ERROR   +3              ;FPP ERROR
12511                                                             ;BAD AUTO-INCR
12512 061572  012704  003254        2$:   MOV     #TAB3,R4                    ;POINT TO RECEIVED DATA
12513 061576  004767  056276              JSR     R7,DATVER           ;VERFIY DATA FROM FPP
12514 061602  005767  121312              TST     COUNT               ;SEE IF COUNTER=0
12515 061606  001403                      BEQ     3$                  ;BRANCH IF GOOD COMPARE
12516 061610  004737  140132              CALL    @#DETFPA            ;DETERMINE FLOATING POINT FAULT. $$$
12517 061614  104003                      ERROR   +3              ;FPP ERROR
12518                                                             ;BAD DATA FROM FPP
12519 061616                        3$:
12520                                     ;
12523                                     ;
12524                                     ;----------------------   --- ---
12525                                     ;TEST LDD MODE 5
```

FLOATING POINT TESTS

```
12526                                   ;
12527 061616                            MLDDM5:
12528                                   ;
12529 061616  012701  003142                    MOV     #RECDST,R1              ;POINT TO RECEIVED DATA LOCATION
12530 061622  012704  003164                    MOV     #TSTLOC+2,R4                    ;POINT TO GOOD DATA
12531 061626  012737  003234  003162            MOV     #TAB1,@#TSTLOC                  ;SET UP MODE 5 POINTER TO DATA
12532 061634  012702  047750                    MOV     #47750,R2               ;LOAD GOOD STATUS
12533 061640  170102                            LDFPS   R2                      ;LOAD FPP STATUS   DOUBLE,ID
12534 061642  172454                            LDD     @-(R4),AC0              ;*TEST INSTRUCTION - MODE 5
12535 061644  170203                            STFPS   R3                      ;SAVE TEST FPP STATUS
12536 061646  174011                            STD     AC0,(R1)                ;SAVE TEST RESULT
12537 061650  020203                            CMP     R2,R3                   ;VERIFY FPP STATUS
12538 061652  001403                            BEQ     1$                      ;BRANCH IF GOOD
12539 061654  004737  140132                    CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12540 061660  104003                            ERROR   +3              ;FPP ERROR
12541                                           ;BAD FPP STATUS
12542 061662  022704  003162            1$:     CMP     #TSTLOC,R4                      ;VERFIY AUTO-DEC
12543 061666  001403                            BEQ     2$                      ;BRANCH IF GOOD
12544 061670  004737  140132                    CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12545 061674  104003                            ERROR   +3              ;FPP ERROR
12546                                           ;BAD AUTO-DEC
12547 061676  012704  003234            2$:     MOV     #TAB1,R4                        ;POINT TO EXPECTED DATA
12548 061702  004767  056172                    JSR     R7,DATVER               ;VERFIY DATA FROM FPP
12549 061706  005767  121206                    TST     COUNT                   ;SEE IF COUNTER=0
12550 061712  001403                            BEQ     3$                      ;BRANCH IF GOOD COMPARE
12551 061714  004737  140132                    CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12552 061720  104003                            ERROR   +3              ;FPP ERROR
12553                                           ;BAD DATA FROM FPP
12554 061722                            3$:
12557                                   ;
12558                                   ;---------------------------------
12559                                   ;TEST LDD MODE 6
12560                                   ;
12561 061722                            MLDDM6:
12562                                   ;
12563 061722  012701  003342                    MOV     #RECDST+200,R1          ;POINT TO RECEIVED DATA LOCATION
12564 061726  012704  003044                    MOV     #TAB2-200,R4                    ;SETUP R4 FOR MODE 6
12565 061732  012702  047750                    MOV     #47750,R2               ;LOAD GOOD STATUS
12566 061736  170102                            LDFPS   R2                      ;LOAD FPP STATUS - DOUBLE,ID
12567 061740  172464  000200                    LDD     200(R4),AC0             ;LDD MODE 6
12568 061744  170203                            STFPS   R3                      ;SAVE TEST FPP STATUS
12569 061746  174061  177600                    STD     AC0,-200(R1)                    ;SAVE TEST RESULT
12570 061752  022703  047740                    CMP     #47740,R3                       ;VERIFY FPP STATUS
12571 061756  001403                            BEQ     1$                      ;BRANCH IF GOOD
12572 061760  004737  140132                    CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12573 061764  104003                            ERROR   +3              ;FPP ERROR
12574                                           ;BAD FPP STATUS
12575 061766  162701  000200            1$:     SUB     #200,R1                 ;R1=RECDST
12576 061772  062704  000200            2$:     ADD     #200,R4                 ;POINT TO EXPECTED DATA
12577 061776  004767  056076                    JSR     R7,DATVER               ;VERFIY DATA FROM FPP
12578 062002  005767  121112                    TST     COUNT                   ;SEE IF COUNTER=0
12579 062006  001403                            BEQ     3$                      ;BRANCH IF GOOD COMPARE
12580 062010  004737  140132                    CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12581 062014  104003                            ERROR   +3              ;FPP ERROR
12582                                           ;BAD DATA FROM FPP
12583 062016                            3$:
12586                                   ;
```

FLOATING POINT TESTS

```
12587                                   ;---- ------------ ---- - ---- -
12588                                   ;TEST LDD MODE 7
12589                                   ;
12590 062016                            MLDDM7:
12591                                   ;
12592 062016  012701  003142                    MOV     #RECDST,R1              ;POINT TO RECEIVED DATA LOCATION
12593 062022  005004                            CLR     R4                      ;R4=0
12594 062024  012727  003234  003162            MOV     #TAB1,#TSTLOC           ;POINTER FOR MODE 7 GOOD DATA
12595 062032  012702  047750                    MOV     #47750,R2               ;LOAD GOOD STATUS
12596 062036  170102                            LDFPS   R2                      ;LOAD FPP STATUS   DOUBLE.ID
12597 062040  172474  003162                    LDD     @TSTLOC(R4),AC0         ;*TEST INSTRUCTION   MODE 7
12598 062044  170203                            STFPS   R3                      ;SAVE TEST FPP STATUS
12599 062046  174011                            STD     AC0,(R1)                        ;SAVE TEST RESULT
12600 062050  020203                            CMP     R2,R3                   ;VERIFY FPP STATUS
12601 062052  001403                            BEQ     1$                      ;BRANCH IF GOOD
12602 062054  004737  140132                    CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAU T. $$$
12603 062060  104003                            ERROR   +3              ;FPP ERROR
12604                                                                   ;BAD FPP STATUS
12605 062062  005704                    1$:     TST     R4                      ;VERFIY CONTENTS OF R4
12606 062064  001403                            BEQ     2$                      ;BRANCH IF GOOD
12607 062066  004737  140132                    CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12608 062072  104003                            ERROR   +3              ;FPP ERROR
12609                                                                   ;BAD R4
12610 062074  012704  003234            2$:     MOV     #TAB1,R4                        ;POINT TO RECEIVED DATA
12611 062100  004767  055774                    JSR     R7,DATVER               ;VERFIY DATA FROM FPP
12612 062104  005767  121010                    TST     COUNT                   ;SEE IF COUNTER=0
12613 062110  001403                            BEQ     3$                      ;BRANCH IF GOOD COMPARE
12614 062112  004737  140132                    CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12615 062116  104003                            ERROR   +3              ;FPP ERROR
12616                                                                   ;BAD DATA FROM FPP
12617 062120                            3$:
12620                                   ;
12621                                   ;------ - -------------------- ---
12622                                   ;TEST LDD MODE 27 - ONLY 16 BITS ARE LOADED OR STORED
12623                                   ;
12624 062120                            MLDM27:
12625                                   ;
12626 062120  012701  003142                    MOV     #RECDST,R1              ;POINT TO RECEIVED DATA LOCATION
12627 062124  012704  003274                    MOV     #TAB5,R4                        ;POINT TO GOOD DATA
12628 062130  012702  047750                    MOV     #47750,R2               ;LOAD GOOD STATUS
12629 062134  005005                            CLR     R5                      ;R5=0
12630 062136  170102                            LDFPS   R2                      ;LOAD FPP STATUS - DOUBLE.ID
12631 062140  172427  043243                    LDD     #5205,AC0               ;*TEST INSTRUCTION - MODE 27
12632 062144  005205                            INC     R5
12633 062146  005205                            INC     R5
12634 062150  005205                            INC     R5                      ; TEST PROPER PC PATH
12635 062152  022705  000003                    CMP     #3,R5                   ;VERIFY ONLY 3 PC INCREMENT
12636 062156  001403                            BEQ     1$                      ;BRANCH IF PROPER PC ACTION
12637 062160  004737  140132                    CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12638 062164  104003                            ERROR   +3              ;FPP ERROR
12639                                                                   ;BAD MODE 27 LOAD
12640 062166  170203                    1$:     STFPS   R3                      ;SAVE TEST FPP STATUS
12641 062170  174011                            STD     AC0,(R1)                        ;SAVE TEST RESULT
12642 062172  022703  047740                    CMP     #47740,R3                       ;VERIFY FPP STATUS
12643 062176  001403                            BEQ     2$                      ;BRANCH IF GOOD
12644 062200  004737  140132                    CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12645 062204  104003                            ERROR   +3              ;FPP ERROR
```

FLOATING POINT TESTS

```
12646                                                   ;BAD FPP STATUS
12647 062206  004767  055666          2$:     JSR     R7,DATVER               ;VERFIY DATA FROM FPP
12648 062212  005767  120702                  TST     COUNT                   ;SEE IF COUNTER=0
12649 062216  001403                          BEQ     3$                      ;BRANCH IF GOOD COMPARE
12650 062220  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12651 062224  104003                          ERROR   +3                      ;FPP ERROR
12652                                                                         ;BAD DATA FROM FPP
12653 062226                          3$:
12656                                         ;
12657                                         ;------- ----- - -- ------ -----
12658                                         ;TEST ADDF, ADDD, SUBF, SUBD   AC0=0 FSRC=0;
12659                                         ;
12660 062226                          MNNRM1:
12661                                         ;
12662 062226  012704  003314                  MOV     #TAB6,R4                ;POINT TO FSRC TEST DATA
12663 062232  005067  120710                  CLR     RECDST+4                ;CLEAR OUT RECEIVED DATA TABLE
12664 062236  005067  120706                  CLR     RECDST+6                ;
12665 062242  012702  040000                  MOV     #40000,R2               ;SET UP GOOD STATUS
12666 062246  170102                          LDFPS   R2                      ;LOAD FPP STATUS, FLOATING
12667 062250  172414                          LDF     (R4),AC0                ;LOAD AC0 WITH 0
12668 062252  172014                          ADDF    (R4),AC0                ;0+0
12669 062254  170203                          STFPS   R3                      ;SAVE STATUS
12670 062256  022703  040004                  CMP     #40004,R3               ;VERIFY STATUS
12671 062262  001403                          BEQ     1$                      ;BRANCH IF GOOD
12672 062264  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12673 062270  104003                          ERROR   +3                      ;FPP ERROR
12674                                                                         ;BAD FPP STATUS
12675 062272  012701  003142          1$:     MOV     #RECDST,R1              ;POINT TO RECEIVERD DATA
12676 062276  174011                          STF     AC0,(R1)                ;SAVE DATA
12677 062300  004767  055574                  JSR     R7,DATVER               ;VERIFY DATA
12678 062304  005767  120610                  TST     COUNT                   ;
12679 062310  001403                          BEQ     2$                      ;BRANCH IF GOOD
12680 062312  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12681 062316  104003                          ERROR   +3                      ;FPP ERROR
12682                                                                         ;BAD DATA IN AC0
12683 062320  012702  040200          2$:     MOV     #40200,R2               ;LOAD FLOATING STATUS
12684 062324  170102                          LDFPS   R2                      ;
12685 062326  172414                          LDD     (R4),AC0                ;LOAD AC0 WITH 0
12686 062330  172014                          ADDD    (R4),AC0                ;*TEST INSTRUCTION
12687 062332  174011                          STD     AC0,(R1)                ;SAVE DATA
12688 062334  170203                          STFPS   R3                      ;SAVE FPS
12689 062336  022703  040204                  CMP     #40204,R3               ;VERFIY STATUS
12690 062342  001403                          BEQ     3$                      ;BRANCH IF GOOD
12691 062344  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12692 062350  104003                          ERROR   +3                      ;FPP ERROR
12693                                                                         ;BAD FPS
12694 062352  004767  055522          3$:     JSR     R7,DATVER                       ;VERFIY DATA
12695 062356  005737  003120                  TST     @#COUNT                 ;VERIFY RESULT
12696 062362  001403                          BEQ     44$                     ;BRANCH IF GOOD
12697 062364  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12698 062370  104003                          ERROR   +3                      ;FPP ERROR
12699                                                                         ;BAD AC0
12700 062372  172414                  44$:    LDD     (R4),AC0                ;SETUP DATA
12701 062374  173014                          SUBD    (R4),AC0                ;*TEST INSTRUCTION
12702 062376  170203                          STFPS   R3                      ;SAVE STATUS
12703 062400  022703  040204                  CMP     #40204,R3                       ;VERFIY STATUS
12704 062404  001403                          BEQ     4$                      ;BRANCH IF GOOD
```

F2

FLOATING POINT TESTS

```
12705 062406  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12706 062412  104003                          ERROR   +3                      ;FPP ERROR
12707                                                                         ;BAD FPS
12708 062414  174011                  4$:     STD     AC0,(R1)                ;SAVE AC0 DATA
12709 062416  004767  055456                  JSR     R7,DATVER                       ;VERFIY DATA
12710 062422  005767  120472                  TST     COUNT                   ;
12711 062426  001403                          BEQ     5$                      ;BRANCH IF GOOD
12712 062430  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12713 062434  104003                          ERROR   +3                      ;FPP ERROR
12714                                                                         ;BAD AC0
12715 062436  170127  000000          5$:     LDFPS   #0                      ;STORE FPP STATUS
12716 062442  172414                          LDD     (R4),AC0                ;LOAD AC0
12717 062444  173014                          SUBF    (R4),AC0                        ;0-0
12718 062446  170203                          STFPS   R3                      ;SAVE STATUS
12719 062450  174011                          STD     AC0,(R1)                ;SAVE AC0
12720 062452  022703  000004                  CMP     #4,R3                   ;VERFIY STATUS
12721 062456  001403                          BEQ     6$                      ;BRANCH IF GOOD
12722 062460  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12723 062464  104003                          ERROR   +3                      ;FPP ERROR
12724                                                                         ;BAD FPS
12725 062466  004767  055406          6$:     JSR     R7,DATVER                       ;VERIFY DATAT
12726 062472  005767  120422                  TST COUNT                       ;
12727 062476  001403                          BEQ     7$                      ;BRANC IF GOOD
12728 062500  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12729 062504  104003                          ERROR   +3                      ;FPP ERROR
12730                                                                         ;BAD AC0
12731 062506                          7$:
12732                                 ;
12735                                 ;
12736                                 ;---------- ---------- -
12737                                 ;TEST ADDF,SUBD - FSRC=0, AC0 NE 0
12738                                 ;
12739 062506                          MNNRM2:
12740                                 ;
12741 062506  012701  003142                  MOV     #RECDST,R1              ;POINT TO RECEIVED DATA TABLE
12742 062512  012705  003314                  MOV     #TAB6,R5                ;POINT TO SOURCE DATA TABLE
12743 062516  012704  003244                  MOV     #TAB2,R4                ;POINT TO AC0 DATA
12744 062522  170127  000200                  LDFPS   #200                    ;SET TO DOUBLE FOR CLEAR
12745 062526  172415                          LDD     (R5),AC0                ;
12746 062530  005002                          CLR     R2                      ;SETUP FPP STATUS
12747 062532  170102                          LDFPS   R2                      ;LOAD FPS
12748 062534  172414                          LDF     (R4),AC0                ;LOAD AC0
12749 062536  172015                          ADDF    (R5),AC0                        ;*TEST INSTRUCTION
12750 062540  170203                          STFPS   R3                      ;SAVE STATUS
12751 062542  174011                          STF     AC0,(R1)                ;SAVE AC0
12752 062544  022703  000000                  CMP     #0,R3                   ;VERIFY NEGATIVE RESULT
12753 062550  001403                          BEQ     1$                      ;BRANCH IF GOOD
12754 062552  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12755 062556  104003                          ERROR   +3                      ;FPP ERROR
12756                                                                         ;BAD FPS
12757 062560  012704  003264          1$:     MOV     #TAB4,R4                ;POINT TO EXPECTED DATA
12758 062564  004767  055310                  JSR     R7,DATVER                       ;VERFIY AC0
12759 062570  005767  120324                  TST     COUNT                   ;CHECK RESULT
12760 062574  001403                          BEQ     2$                      ;BRANCH IF GOOD
12761 062576  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $.$
12762 062602  104003                          ERROR   +3                      ;FPP ERROR
12763                                                                         ;BAD AC0
```

FLOATING POINT TESTS

```
12764 062604  170127  000200          2$:     LDFPS   #200             ;SET STATUS TO DUOBLE NODE
12765 062610  172414                          LDD     (R4),AC0                ;LOAD AC0 WITH A VALUE
12766 062612  173015                          SUBD    (R5),AC0                ;*TEST INSTRUCTION
12767 062614  170203                          STFPS   R3                      ;SAVE FPP STATUS
12768 062616  174011                          STD     AC0,(R1)                ;SAVE AC0
12769 062620  022703  000200                  CMP     #200,R3         ;VERIFY RESULT
12770 062624  001403                          BEQ     3$                      ;BRANCH IF GOOD
12771 062626  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12772 062632  104003                          ERROR   +3              ;FPP ERROR
12773                                                                 ;BAD SUBD
12774 062634  012704  003264          3$:     MOV     #TAB4,R4                ;POINT TO EXPECTED
12775 062640  004767  055234                  JSR     R7,DATVER               ;VERIFY AC0
12776 062644  005767  120250                  TST     COUNT                   ;
12777 062650  001403                          BEQ     4$                      ;BRANCH IF GOOD AC0
12778 062652  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12779 062656  104003                          ERROR   +3              ;FPP ERROR
12780                                                                 ;BAD AC0
12781 062660                          4$:
12784                                 ;
12785                                 ;-------------------- ------------
12786                                 ;TEST ADDD, SUBF - FSRC NE 0, AC0=0
12787                                 ;
12788 062660                          MNNRM3:
12789                                 ;
12790 062660  012701  003142                  MOV     #RECDST,R1              ;POINT TO RECEIVED DATA TABLE
12791 062664  012705  003314                  MOV     #TAB6,R5                ;POINT TO AC0 DATA TABLE
12792 062670  012704  003234                  MOV     #TAB1,R4                ;POINT TO FSRC DATA
12793 062674  012702  000200                  MOV     #200,R2                 ;SETUP FPP STATUS
12794 062700  170102                          LDFPS   R2                      ;LOAD FPS
12795 062702  172415                          LDD     (R5),AC0                ;LOAD AC0
12796 062704  172014                          ADDD    (R4),AC0                        ;*TEST INSTRUCTION
12797 062706  170203                          STFPS   R3                      ;SAVE STATUS
12798 062710  174011                          STD     AC0,(R1)                ;SAVE AC0
12799 062712  022703  000210                  CMP     #210,R3                 ;VERFIY NEGATIVE RESULT
12800 062716  001403                          BEQ     1$                      ;BRANCH IF GOOD
12801 062720  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12802 062724  104003                          ERROR   +3              ;FPP ERROR
12803                                                                 ;BAD FPS
12804 062726  004767  055146          1$:     JSR     R7,DATVER                       ;VERFIY AC0
12805 062732  005767  120162                  TST     COUNT                   ;CHECK RESULT
12806 062736  001403                          BEQ     2$                      ;BRANCH IF GOOD
12807 062740  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12808 062744  104003                          ERROR   +3              ;FPP ERROR
12809                                                                 ;BAD AC0
12810 062746  170127  000200          2$:     LDFPS   #200            ;SET STATUS TO DUOBLE NODE
12811 062752  172415                          LDF     (R5),AC0                ;LOAD AC0 WITH A VALUE
12812 062754  173014                          SUBD    (R4),AC0                ;*TEST INSTRUCTION
12813 062756  170203                          STFPS   R3                      ;SAVE FPP STATUS
12814 062760  174011                          STF     AC0,(R1)                        ;SAVE AC0
12815 062762  022703  000200                  CMP     #200,R3         ;VERIFY RESULT
12816 062766  001403                          BEQ     3$                      ;BRANCH IF GOOD
12817 062770  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12818 062774  104003                          ERROR   +3              ;FPP ERROR
12819                                                                 ;BAD SUBD
12820 062776  012704  003504          3$:     MOV     #TAB18,R4               ;POINT TO EXPECTED DATA
12821 063002  004767  055072                  JSR     R7,DATVER               ;VERIFY AC0
12822 063006  005767  120106                  TST     COUNT                   ;
```

H2

FLOATING POINT TESTS

```
12823 063012  001403                        BEQ     4$                      ;BRANCH IF GOOD ACO
12824 063014  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12825 063020  104003                        ERROR   +3              ;FPP ERROR
12826                                                               ;BAD ACO
12827 063022                        4$:
12828                               ;
12831                               ;
12832                               ;--------------------------------
12833                               ;TEST  ADDF, SUBD - EXP(ACO) = EXP(FSRC)
12834                               ;
12835 063022                        MNNRM4:
12836                               ;
12837 063022  012702  003240                MOV     #3240,R2                ;SET FIU,FD,FT
12838 063026  170102                        LDFPS   R2
12839 063030  012704  003334                MOV     #TAB7,R4                        ;SET FSRC
12840 063034  012705  003344                MOV     #TAB8,R5                ;SETUP ^CO
12841 063040  012701  003142                MOV     #RECDST,R1              ;POINT  J RECEIVED DATA
12842 063044  172415                        LDD     (R5),AC0                        ;LOAD AC0
12843 063046  172014                        ADDD    (R4),AC0                ;*TEST INSTRUCTION
12844 063050  174011                        STD     AC0,(R1)                ;SAVE TEST RESULT
12845 063052  012704  003354                MOV     #TAB9,R4                ;POINT TO EXPECTED DATA
12846 063056  004767  055016                JSR     R7,DATVER                       ;VERIFY AC0 DATA
12847 063062  005767  120032                TST     COUNT                   ;
12848 063066  001403                        BEQ     1$                      ;BRANCH IF GOOD
12849 063070  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12850 063074  104003                        ERROR   +3              ;FPP ERROR
12851                                                               ;BAD ADDD
12852 063076  012704  003344        1$:     MOV     #TAB8,R4                ;
12853 063102  012703  003344                MOV     #TAB8,R3                ;SETUP SAME AC0
12854 063106  012702  003200                MOV     #3200,R2                ;ROUND MODE
12855 063112  170102                        LDFPS   R2
12856 063114  172413                        LDD     (R3),AC0                ;LOAD AC0
12857 063116  061400                        ADD     (R4),AC0                ;*TEST INSTRUCTION
12858 063120  174011                        STD     AC0,(R1)                ;SAVE DATA
12859 063122  004767  054752                JSR     R7,DATVER               ;VERIFY AC0
12860 063126  005767  117766                TST     COUNT
12861 063132  001403                        BEQ     2$                      ;BRANCH IF GOOD
12862 063134  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12863 063140  104003                        ERROR   +3              ;FPP ERROR
12864                                                               ;BAD ROUND RESULT
12865 063142                        2$:
12866                               ;
12869                               ;
12870                               ;--------------------------------
12871                               ;TEST ADD - EXP(FSRC) > EXP(ACO)
12872                               ;
12873 063142                        MXDF1:
12874                               ;
12875 063142  012702  003200                MOV     #3200,R2                ;R2=FPP STATUS
12876 063146  170102                        LDFPS   R2                      ;LOAD FPS STATUS
12877 063150  012704  003374                MOV     #TAB11,R4               ;POINT TO FSRC DATA
12878 063154  012701  003142                MOV     #RECDST,R1              ;POINT TO AC0 RESULT
12879 063160  012705  003364                MOV     #TAB10,R5               ;POINT TO AC0 DATA
12880 063164  172415                        LDD     (R5),AC0                ;LOAD AC0 DATA
12881 063166  172014                        ADDD    (R4),AC0                ;*TEST INSTRUCTIONS
12882 063170  170203                        STFPS   R3                      ;SAVE FPP STATUS
12883 063172  174011                        STD     AC0,(R1)                ;SAVE AC0 DATA
```

FLOATING POINT TESTS

```
12884 063174  022703  003200            CMP     #3200,R3            ;VERIFY FPP STATUS
12885 063200  001403                    BEQ     1$                  ;BRANCH IF GOOD
12886 063202  004737  140132            CALL    @#DETFPA            ;DETERMINE FLOATING POINT FAULT. $$$
12887 063206  104003                    ERROR   +3                  ;FPP ERROR
12888                                                               ;BAD FPP STATUS
12889 063210  012704  003404      1$:   MOV     #TAB11A,R4          ;POINT TO EXPECTED DATA
12890 063214  004767  054660            JSR     R7,DATVER           ;VERIFY CONTENTS OF ACO
12891 063220  005767  117674            TST     COUNT               ;
12892 063224  001403                    BEQ     2$                  ;BRANCH IF GOOD ACO
12893 063226  004737  140132            CALL    @#DETFPA            ;DETERMINE FLOATING POINT FAULT. $$$
12894 063232  104003                    ERROR   +3                  ;FPP ERROR
12895                                                               ;BAD ACO, SHOULD = FSRC
12896 063234  012704  003414      2$:   MOV     #TAB12,R4           ;POINT TO FSRC DATA
12897 063240  172415                    LDD     (R5),AC0            ;AC0
12898 063242  172014                    ADDD    (R4),AC0            ;*TEST INSTRUCTION
12899 063244  012704  003434            MOV     #TAB13B,R4          ;POINT TO EXPECTED RESULT
12900 063250  174011                    STD     AC0,(R1)            ;SAVE AC0 DATA INTO RECDAT
12901 063252  004767  054622            JSR     R7,DATVER           ;VERIFY DATA
12902 063256  005767  117636            TST     COUNT
12903 063262  001403                    BEQ     3$                  ;BRANCH IF GOOD DATA
12904 063264  004737  140132            CALL    @#DETFPA            ;DETERMINE FLOATING POINT FAULT. $$$
12905 063270  104003                    ERROR   +3                  ;FPP ERROR
12906                                                               ;BAD AC0 DATA
12907 063272  012702  003000      3$:   MOV     #3000,R2            ;GET FPP STATUS DATA
12908 063276  012704  003444            MOV     #TAB14,R4           ;POINT TO FSRC DATA
12909 063302  012705  003454            MOV     #TAB15,R5           ;POINT TO AC0 DATA
12910 063306  172415                    LDD     (R5),AC0            ;LOAD AC0
12911 063310  170102                    LDFPS   R2                  ;FPP STATUS = FLOAT, INTERRUPTS ENABLE
12912 063312  172014                    ADDF    (R4),AC0            ;*TEST INSTRUCTION
12913 063314  170127  000200            LDFPS   #200                ;RESET TO DOUBLE
12914 063320  174011                    STD     AC0, (R1)           ;RECDST=AC0
12915 063322  012704  003364            MOV     #TAB10,R4           ;POINT TO GOOD DATA
12916 063326  004767  054546            JSR     R7,DATVER                   ;VERFIY CONTENTS OF AC0
12917 063332  005767  117562            TST     COUNT               ;
12918 063336  001403                    BEQ     4$                  ;BRANCH IF GOOD
12919 063340  004737  140132            CALL    @#DETFPA            ;DETERMINE FLOATING POINT FAULT. $$$
12920 063344  104003                    ERROR   +3                  ;FPP ERROR
12921                                                               ;BAD FLOATING ADD
12922 063346  012705  003464      4$:   MOV     #TAB16,R5           ;POINT TO AC0 DATA
12923 063352  170102                    LDFPS   R2                  ;FPP STATUS = FLOAT
12924 063354  172415                    LDF     (R5),AC0            ;LOAD AC0
12925 063356  172014                    ADDF    (R4),AC0            ;*TEST INSTRUCTION
12926 063360  174011                    STD     AC0,(R1)                    ;SAVE AC0  DATA
12927 063362  012704  003474            MOV     #TAB17,R4           ;POINT TO GOOD DATA
12928 063366  004767  054506            JSR     R7,DATVER
12929 063372  005767  117522            TST     COUNT
12930 063376  001403                    BEQ     5$                  ;BRANCH IF GOOD
12931 063400  004737  140132            CALL    @#DETFPA            ;DETERMINE FLOATING POINT FAULT. $$$
12932 063404  104003                    ERROR   +3                  ;FPP ERROR
12933                                                               ;BAD FLOATING ADD
12934 063406                      5$:
12935                             ;
12938                             ;
12939                             ;-------------- ----- -- ----
12940                             ;TEST ADDD WITH NEGATIVE OPERANDS
12941                             ;
12942 063406                      MNGOP:
```

FLOATING POINT TESTS

```
12943                                           ;
12944 063406  012702  003200                      MOV     #3200,R2                ;LOAD FPS VALUE
12945 063412  170102                              LDFPS   R2                      ;
12946 063414  012704  003514                      MOV     #TAB21,R4               ;DATA ADDRESS FOR AC0 AND FSR
12947 063420  172414                              LDD     (R4),AC0                ;AC0=100200 0 0 0
12948 063422  172014                              ADDD    (R4),AC0                ;*TEST INSTRUCTION
12949 063424  170203                              STFPS   R3                      ;SAVE STATUS
12950 063426  012701  003142                      MOV     #RECDST,R1              ;POINT TO RECEIVED DATA TABLE
12951 063432  174011                              STD     AC0,(R1)                ;SAVE AC0 DATA
12952 063434  022703  003210                      CMP     #3210,R3                ;VERIFY STATUS
12953 063440  001403                              BEQ     1$                      ;BRANCH IF GOOD
12954 063442  004737  140132                      CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12955 063446  104003                              ERROR   +3              ;FPP ERROR
12956                                             ;
12957 063450  012704  003524            1$:       MOV     #TAB22,R4               ;POINT TO EXPECTED DATA
12958 063454  004767  054420                      JSR     R7,DATVER               ;
12959 063460  005767  117434                      TST     COUNT                   ;VERIFY DATA
12960 063464  001403                              BEQ     2$                      ;BRANCH IF GOOD
12961 063466  004737  140132                      CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12962 063472  104003                              ERROR   +3              ;FPP ERROR
12963                                             ;
12964                                           ;       !-FSRC! = !AC0!
12965 063474  012704  003514            2$:       MOV     #TAB21,R4               ;POINT TO FSRC DATA
12966 063500  012701  003534                      MOV     #TAB23,R1               ;POINT TO AC0 DATA
12967 063504  012737  063526  000244              MOV     #101$,@#FPVEC           ;SETUP FP VECTOR
12968 063512  172411                              LDD     (R1),AC0                ;LOAD AC0
12969 063514  172014                              ADDD    (R4),AC0                ;*TEST INSTRUCTION
12970 063516  170000                    100$:     CFCC                            ;COPY FPP CC
12971 063520  004737  140132                      CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12972 063524  104003                              ERROR   +3              ;FPP ERROR
12973                                             ;GO TO ERROR
12974 063526  170203                    101$:     STFPS   R3                      ;SAVE FPP STATUS
12975 063530  012701  003142                      MOV     #RECDST,R1                      ;POINT TO RECEIVED DATA TABLE
12976 063534  174011                              STD     AC0,(R1)                ;SAVE AC0 DATA
12977 063536  022703  103200                      CMP     #103200,R3              ;VERIFY STATUS
12978 063542  001403                              BEQ     3$                      ;BRANCH IF GOOD
12979 063544  004737  140132                      CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12980 063550  104003                              ERROR   +3              ;FPP ERROR
12981                                             ;BAD STATUS
12982 063552  012605                    3$:       MOV     (SP)+,R5                ;GET ERROR PC
12983 063554  020527  063516                      CMP     R5,#100$                ;VERIFY ERROR ADDRESS ON STACK
12984 063560  001403                              BEQ     102$                    ;BRANCH IF GOOD
12985 063562  004737  140132                      CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12986 063566  104003                              ERROR   +3              ;FPP ERROR
12987                                             ;BAD ERROR RETURN ON STACK
12988 063570  005726                    102$:     TST     (SP)+                   ;RESTORE STACK
12989 063572  012704  003544                      MOV     #TAB24,R4               ;POINT TO EXPECTED DATA TABLE
12990 063576  004767  054276                      JSR     R7,DATVER               ;VERIFY DATA
12991 063602  005767  117312                      TST     COUNT
12992 063606  001403                              BEQ     4$                      ;BRANC IF GOOD
12993 063610  004737  140132                      CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
12994 063614  104003                              ERROR   +3              ;FPP ERROR
12995                                             ;BAD AC0 DATA
12996                                           ;       ! AC0! = !FSRC!
12997 063616  012704  003534            4$:       MOV     #TAB23,R4               ;POINT TO FSRC DATA
12998 063622  012701  003514                      MOV     #TAB21,R1               ;POINT TO AC0 DATA
12999 063626  012737  063656  000244              MOV     #104$,@#FPVEC           ;SETUP FP VECTOR
```

K2

FLOATING POINT TESTS

```
13000 063634  012702  003200                MOV     #3200,R2                ;LOAD FPS VALUE
13001 063640  170102                        LDFPS   R2                      ;
13002 063642  172411                        LDD     (R1),AC0                ;LOAD AC0 DATA
13003 063644  172014                        ADDD    (R4),AC0                ;*TEST INSTRUCTION
13004 063646  170000              103$:     CFCC                            ;COPY FPP CC
13005 063650  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
13006 063654  104003                        ERROR   +3                      ;FPP ERROR
13007                                                                       ;GO TO ERROR
13008 063656  170203              104$:     STFPS   R3                      ;SAVE FPS
13009 063660  012701  003142                MOV     #RECDST,R1              ;SAVE AC0
13010 063664  174011                        STD     AC0,(R1)                ;
13011 063666  022703  103200                CMP     #103200,R3              ;VERFIY STATUS
13012 063672  001403                        BEQ     5$                      ;BRANCH IF GOOD
13013 063674  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
13014 063700  104003                        ERROR   +3                      ;FPP ERROR
13015                                                                       ;BAD FPS STATUS
13016 063702  012605              5$:       MOV     (SP)+,R5                ;GET ERROR PC
13017 063704  020527  063646                CMP     R5,#103$                ;VERIFY ERROR ADDRESS ON STACK
13018 063710  001403                        BEQ     105$                    ;BRANCH IF GOOD
13019 063712  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
13020 063716  104003                        ERROR   +3                      ;FPP ERROR
13021                                                                       ;BAD ERROR RETURN ON STACK
13022 063720  005726              105$:     TST     (SP)+                   ;RESTORE STACK
13023 063722  012704  003544                MOV     #TAB24,R4               ;POINT TO EXPECTED DATA
13024 063726  004767  054146                JSR     R7,DATVER
13025 063732  005767  117162                TST     COUNT                   ;
13026 063736  001403                        BEQ     6$                      ;BRANCH IF  GOOD
13027 063740  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
13028 063744  104003                        ERROR   +3                      ;FPP ERROR
13029                                                                       ;BAD AC0
13030                                       ;       !-FSRC! <               !AC0.
13031 063746  012704  003564        6$:     MOV     #TAB26,R4               ;POINT TO FSRC DATA
13032 063752  012701  003554                MOV     #TAB25,R1               ;POINT TO AC0 DATA
13033 063756  012702  003200                MOV     #3200,R2                ;LOAD FPS VALUE
13034 063762  170102                        LDFPS   R2                      ;
13035 063764  012737  000246  000244        MOV     #246,@#FPVEC            ;SETUP FP VECTOR
13036 063772  172411                        LDD     (R1),AC0                ;LOAD AC0 DATA
13037 063774  172014                        ADDD    (R4),AC0                ;*TEST INSTRUCTION
13038 063776  170203                        STFPS   R3                      ;SAVE STATUS
13039 064000  012701  003142                MOV     #RECDST,R1              ;POINT TO RECEIVED DATA TABLE
13040 064004  174011                        STD     AC0,(R1)                        ;SAVE AC0
13041 064006  020327  005200                CMP     R3,#3200                        ;VERIFY STATUS
13042 064012  001403                        BEQ     7$                      ;BRANCH IF GOOD
13043 064014  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
13044 064020  104003                        ERROR   +3                      ;FPP ERROR
13045                                                                       ;BAD FPS
13046 064022  012704  003574        7$:     MOV     #TAB27,R4                       ;POINT TO EXPECTED DSATA
13047 064026  004767  054046                JSR     R7,DATVER                       ;VERIFY DATA
13048 064032  005767  117062                TST     COUNT                   ;
13049 064036  001403                        BEQ     8$                      ;BRANCH IF GOOD
13050 064040  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
13051 064044  104003                        ERROR   +3                      ;FPP ERROR
13052                                                                       ;
13053                                       ;       !FSRC!  >               !-AC!
13054 064046  012704  003554        8$:     MOV     #TAB25,R4               ;POINT TO FSRC DATA
13055 064052  012701  003564                MOV     #TAB26,R1               ;POINT TO AC0 DATA
13056 064056  172411                        LDD     (R1),AC0                ;LOAD AC0 DATA
```

FLOATING POINT TESTS

```
13057 064060  172014                          ADDD    (R4),AC0                ;*TEST INSTRUCTION
13058 064062  170203                          STFPS   R3                      ;SAVE STATUS
13059 064064  012701  003142                  MOV     #RECDST,R1              ;POINT TO RECEIVED DATA TABLE
13060 064070  174011                          STD     AC0,(R1)                        ;SAVE AC0
13061 064072  020327  003200                  CMP     R3,#3200                        ;VERIFY STATUS
13062 064076  001403                          BEQ     9$                      ;BRANCH IF GOOD
13063 064100  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
13064 064104  104003                          ERROR   +3                      ;FPP ERROR
13065                                                                         ;BAD FPS
13066 064106  012704  003574          9$:     MOV     #TAB27,R4                       ;POINT TO EXPECTED DSATA
13067 064112  004767  053762                  JSR     R7,DATVER                       ;VERIFY DATA
13068 064116  005767  116776                  TST     COUNT                   ;
13069 064122  001403                          BEQ     10$                     ;BRANCH IF GOOD
13070 064124  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
13071 064130  104003                          ERROR   +3                      ;FPP ERROR
13072                                                                         ;BAD AC0
13073                                         ;       !-FSRC! <               !AC0!
13074 064132  012704  003614          10$:    MOV     #TAB29,R4               ;POINT TO FSRC DATA
13075 064136  012701  003604                  MOV     #TAB28,R1               ;POINT TO AC0 DATA
13076 064142  172411                          LDD     (R1),AC0                ;LOAD AC0 DATA
13077 064144  172014                          ADDD    (R4),AC0                ;*TEST INSTRUCTION
13078 064146  170203                          STFPS   R3                      ;SAVE STATUS
13079 064150  012701  003142                  MOV     #RECDST,R1              ;POINT TO RECEIVED DATA TABLE
13080 064154  174011                          STD     AC0,(R1)                        ;SAVE AC0
13081 064156  020327  003200                  CMP     R3,#3200                        ;VERIFY STATUS
13082 064162  001403                          BEQ     11$                     ;BRANCH IF GOOD
13083 064164  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
13084 064170  104003                          ERROR   +3                      ;FPP ERROR
13085                                                                         ;BAD FPS
13086 064172  012704  003624          11$:    MOV     #TAB29A,R4                      ;POINT TO EXPECTED DATA
13087 064176  004767  053676                  JSR     R7,DATVER                       ;VERIFY DATA
13088 064202  005767  116712                  TST     COUNT                   ;
13089 064206  001403                          BEQ     12$                     ;BRANCH IF GOOD
13090 064210  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
13091 064214  104003                          ERROR   +3                      ;FPP ERROR
13092                                                                         ;
13093 064216                          12$:
13094                                         ;
13097                                         ;
13098                                         ;------- --- -------------- - ----
13099                                         ;TEST SUB WITH EXP[AC0]=EXP[FSRC]
13100                                         ;
13101 064216                          MSB:
13102                                         ;
13103 064216  012702  003200                  MOV     #3200,R2                ;LOAD FPS DATA
13104 064222  170102                          LDFPS   R2                      ;LOAD FPS
13105 064224  012704  003514                  MOV     #TAB21,R4               ;POINT TO FSRC DATA
13106 064230  012701  003142                  MOV     #RECDST,R1              ;POINT TO AC0 RECEIVED DATA TABLE
13107 064234  172414                          LDD     (R4),AC0                ;LOAD AC0
13108 064236  173014                          SUBD    (R4),AC0                        ;*TEST INSTRUCTION
13109 064240  170203                          STFPS   R3                      ;SAVE STATUS
13110 064242  174011                          STD     AC0,(R1)                ;SAVE AC0 INTO RECDST
13111 064244  022703  003204                  CMP     #3204,R3                ;VERIFY STATUS
13112 064250  001403                          BEQ     1$                      ;BRANCH IF GOOD
13113 064252  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
13114 064256  104003                          ERROR   +3                      ;FPP ERROR
13115                                                                         ;BAD FPS STATUS
```

M2

FLOATING POINT TESTS

```
13116 064260  012704  003314          1$:     MOV     #TAB6,R4                ;POINT TO EXPECTED DATA
13117 064264  004767  053610                  JSR     R7,DATVER               ;VERIFY ACO
13118 064270  005767  116624                  TST     COUNT                   ;
13119 064274  001403                          BEQ     2$                      ;BRANCH IF GOOD
13120 064276  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
13121 064302  104003                          ERROR   +3              ;FPP ERROR
13122                                                                 ;BAD ACO
13123 064304  012704  003444          2$:     MOV     #TAB14,R4               ;POINT TO FSRC AND ACO DATA
13124 064310  172414                          LDD     (R4),AC0                ;LOAD ACO DATA
13125 064312  173014                          SUBD    (R4),AC0                ;*TEST  INSTRUCTION
13126 064314  170203                          STFPS   R3                      ;SAVE FPS
13127 064316  174011                          STD     AC0,(R1)                ;SAVE ACO INTO RECDST
13128 064320  022703  003204                  CMP     #3204,R3                ;VERIFY FPS
13129 064324  001403                          BEQ     3$                      ;BRANCH IF GOOD
13130 064326  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
13131 064332  104003                          ERROR   +3              ;FPP ERROR
13132                                                                 ;BAD ACO
13133 064334  012704  003314          3$:     MOV     #TAB6,R4                ;POINT TO EXPECTED DATA
13134 064340  004767  053534                  JSR     R7,DATVER               ;VERIFY ACO
13135 064344  005767  116550                  TST     COUNT
13136 064350  001403                          BEQ     4$                      ;BRANCH IF GOOD
13137 064352  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
13138 064356  104003                          ERROR   +3              ;FPP ERROR
13139                                                                 ;BAD ACO
13140 064360                          4$:
13141                                 ;
13144                                 ;
13145                                 ;---------------------  -------  -
13146                                 ;TEST NORMALIZE
13147                                 ;
13148 064360                          MNRM:
13149                                 ;
13150 064360  012702  003200                  MOV     #3200,R2                ;LOAD FPS
13151 064364  170102                          LDFPS   R2                      ;
13152 064366  012705  003644                  MOV     #TAB31,R5               ;POINT TO FSRC DATA
13153 064372  012701  003634                  MOV     #TAB30,R1               ;POINT TO ACO DATA
13154 064376  172411                          LDD     (R1),AC0                ;LOAD ACO
13155 064400  173015                          SUBD    (R5),AC0                ;*TEST INSTRUCTION
13156                                                                         ;1 LEFT SHIFT
13157 064402  170203                          STFPS   R3                      ;SAVE STATUS
13158 064404  012704  003142                  MOV     #RECDST,R4              ;POINT TO RECDATA
13159 064410  174014                          STD     AC0,(R4)                ;SAVE ACO
13160 064412  012701  003674                  MOV     #TAB34,R1               ;POINT TO EXPECTED DATA
13161 064416  004767  053456                  JSR     R7,DATVER               ;VERIFY DATA
13162 064422  005767  116472                  TST     COUNT
13163 064426  001403                          BEQ     1$                      ;BRANCH IF GOOD
13164 064430  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
13165 064434  104003                          ERROR   +3              ;FPP ERROR
13166                                                                 ;
13167 064436  012701  003654          1$:     MOV     #TAB32,R1               ;ACO DATA
13168 064442  012705  003664                  MOV     #TAB33,R5               ;FSRC DATA
13169 064446  172411                          LDD     (R1),AC0                ;LOAD ACO
13170 064450  173015                          SUBD    (R5),AC0                ;*TST INSTRUCTION
13171                                                                         ;56 LEFT SHIFTS
13172 064452  012701  003142                  MOV     #RECDST,R1              ;SAVE DATA
13173 064456  174011                          STD     AC0,(R1)                ;
13174 064460  004767  053414                  JSR     R7,DATVER
```

N2

FLOATING POINT TESTS

```
13175 064464  005767  116430                  TST     COUNT
13176 064470  001403                          BEQ     2$                      ;
13177 064472  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
13178 064476  104003                          ERROR   +3              ;FPP ERROR
13179                                                                 ;
13180 064500                          2$:
13181                                 ;
13184                                 ;
13185                                 ;-------------------------------
13186                                 ;TEST ADDD WITH OVERFLOW AND UNDERFLOW
13187                                 ;
13188 064500                          MUVAD:
13189                                 ;
13190 064500  012702  000200                  MOV     #200,R2                 ;SETUP FLOATING POINT STATUS
13191 064504  170102                          LDFPS   R2                      ;LOAD FPS
13192 064506  012704  003704                  MOV     #TAB40,R4                       ;POINT TO FSRC DATA
13193 064512  012701  003704                  MOV     #TAB40,R1                       ;POINT TO ACO DATA
13194 064516  172411                          LDD     (R1),AC0                        ;LOAD AC0 WITH TEST DATA
13195 064520  172014                          ADDD    (R4),AC0                ;*TEST INSTRUCTION
13196 064522  170203                          STFPS   R3                      ;SAVE FPS
13197 064524  012701  003142                  MOV     #RECDST,R1              ;POINT TO RECEIVED DATA TABLE
13198 064530  174011                          STD     AC0,(R1)                ;SAVE AC0  RESULT
13199 064532  022703  000206                  CMP     #206,R3                 ;VERIFY STATUS
13200 064536  001403                          BEQ     1$                      ;BRANCH IF GOOD
13201 064540  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
13202 064544  104003                          ERROR   +3              ;FPP ERROR
13203                                                                 ;BAD FPS
13204 064546  012704  003314          1$:     MOV     #TAB6,R4                        ;POINT TO EXPECTED DATA
13205 064552  004767  053322                  JSR     R7,DATVER                       ;VERIFY DATA
13206 064556  005767  116336                  TST     COUNT
13207 064562  001403                          BEQ     2$                      ;BRANCH IF GOOD
13208 064564  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
13209 064570  104003                          ERROR   +3              ;FPP ERROR
13210                                                                 ;BAD AC0
13211                                 ;OVERFLOW TRAPS ENABLED
13212                                 ;
13213 064572  012702  001200          2$:     MOV     #1200,R2                        ;SETUP FLOATING POINT STATUS
13214 064576  170102                          LDFPS   R2                      ;LOAD FPS
13215 064600  012704  003704                  MOV     #TAB40,R4                       ;POINT TO FSRC DATA
13216 064604  012701  003704                  MOV     #TAB40,R1                       ;POINT TO AC0 DATA
13217 064610  172411                          LDD     (R1),AC0                        ;LOAD AC0 WITH TEST DATA
13218 064612  012737  064632  000244          MOV     #3$,@#FPVEC             ;CHANGE TRAP VECTOR
13219 064620  172014                          ADDD    (R4),AC0                ;*TEST INSTRUCTION
13220 064622  170000                  23$:    CFCC
13221 064624  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
13222 064630  104003                          ERROR   +3              ;FPP ERROR
13223                                                                 ;FAILED TO TRAP ON OVERFLOW
13224 064632  170203                  3$:     STFPS   R3                      ;SAVE FPS
13225 064634  012701  003142                  MOV     #RECDST,R1              ;POINT TO RECEIVED DATA TABLE
13226 064640  174011                          STD     AC0,(R1)                ;SAVE AC0  RESULT
13227 064642  022703  101206                  CMP     #101206,R3                      ;VERIFY STATUS
13228 064646  001403                          BEQ     4$                      ;BRANCH IF GOOD
13229 064650  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
13230 064654  104003                          ERROR   +3              ;FPP ERROR
13231                                                                 ;BAD FPS
13232 064656  012600                  4$:     MOV     (SP)+,R0                ;CHECK STORED PC
13233 064660  022700  064622                  CMP     #23$,R0                 ;
```

FLOATING POINT TESTS

```
13234 064664  001403                        BEQ     5$                      ;BRANCH IF RETURN ADDRESS IS GOOD
13235 064666  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
13236 064672  104003                        ERROR   +3                      ;FPP ERROR
13237                                                                       ;BAD RETURN ADDRESS
13238 064674  012600                  5$:   MOV     (SP)+,R0                ;CLEAN UP STACK
13239 064676  012704  003314                MOV     #TAB6,R4                        ;POINT TO EXPECTED DATA
13240 064702  004767  053172                JSR     R7,DATVER                       ;VERIFY DATA
13241 064706  005767  116206                TST     COUNT
13242 064712  001403                        BEQ     7$                      ;BRANCH IF GOOD
13243 064714  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
13244 064720  104003                        ERROR   +3                      ;FPP ERROR
13245                                                                       ;BAD ACO
13246                                 ;UNDERFLOW TRAPS DISABLED
13247                                 ;
13248 064722  012702  000200          7$:   MOV     #200,R2                 ;SETUP FLOATING POINT STATUS
13249 064726  170102                        LDFPS   R2                      ;LOAD FPS
13250 064730  012737  140044  000244        MOV     #WLDTRP,@#FPVEC                 ;REPLACE WILD TRAP VECTOR
13251 064736  012704  003334                MOV     #TAB7,R4                        ;POINT TO FSRC DATA
13252 064742  012701  003714                MOV     #TAB41,R1                       ;POINT TO ACO DATA
13253 064746  172411                        LDD     (R1),AC0                        ;LOAD AC0 WITH TEST DATA
13254 064750  172014                        ADDD    (R4),AC0                ;*TEST INSTRUCTION
13255 064752  170203                        STFPS   R3                      ;SAVE FPS
13256 064754  012701  003142                MOV     #RECDST,R1              ;POINT TO RECEIVED DATA TABLE
13257 064760  174011                        STD     AC0,(R1)                ;SAVE AC0  RESULT
13258 064762  022703  000204                CMP     #204,R3                 ;VERIFY STATUS
13259 064766  001403                        BEQ     8$                      ;BRANCH IF GOOD
13260 064770  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
13261 064774  104003                        ERROR   +3                      ;FPP ERROR
13262                                                                       ;BAD FPS
13263 064776  012704  003314          8$:   MOV     #TAB6,R4                        ;POINT TO EXPECTED DATA
13264 065002  004767  053072                JSR     R7,DATVER                       ;VERIFY DATA
13265 065006  005767  116106                TST     COUNT
13266 065012  001401                        BEQ     9$                      ;BRANCH IF GOOD
13267 065014  104003                        ERROR   +3                      ;FPP ERROR
13268                                                                       ;BAD ACO
13269                                 ;UNDERFLOW TRAPS ENABLED
13270                                 ;
13271 065016  012702  002200          9$:   MOV     #2200,R2                ;SETUP FLOATING POINT STATUS
13272 065022  170102                        LDFPS   R2                      ;LOAD FPS
13273 065024  012737  065056  000244        MOV     #11$,@#FPVEC            ;REPOSITION TRAP VECTOR
13274 065032  012704  003334                MOV     #TAB7,R4                        ;POINT TO FSRC DATA
13275 065036  012701  003714                MOV     #TAB41,R1                       ;POINT TO ACO DATA
13276 065042  172411                        LDD     (R1),AC0                        ;LOAD AC0 WITH TEST DATA
13277 065044  172014                        ADDD    (R4),AC0                ;*TEST INSTRUCTION
13278 065046  170000                  10$:  CFCC                            ;COPY FPP CC
13279 065050  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
13280 065054  104003                        ERROR   +3                      ;FPP ERROR
13281                                                                       ;FAILED TO TRAP ON UNDERFLOW
13282 065056  170203                  11$:  STFPS   R3                      ;SAVE FPS
13283 065060  012701  003142                MOV     #RECDST,R1              ;POINT TO RECEIVED DATA TABLE
13284 065064  174011                        STD     AC0,(R1)                ;SAVE AC0  RESULT
13285 065066  022703  102210                CMP     #102210,R3                      ;VERIFY STATUS
13286 065072  001403                        BEQ     12$                     ;BRANCH IF GOOD
13287 065074  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
13288 065100  104003                        ERROR   +3                      ;FPP ERROR
13289                                                                       ;BAD FPS
13290 065102  012605                  12$:  MOV     (SP)+,R5                ;GET ERROR PC
```

FLOATING POINT TESTS

```
13291 065104  020527  065046                  CMP     R5,#10$                 ;VERIFY ERROR ADDRESS ON STACK
13292 065110  001403                          BEQ     13$                     ;BRANCH IF GOOD
13293 065112  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
13294 065116  104003                          ERROR   .3                      ;FPP ERROR
13295                                                                         ;BAD ERROR RETURN ON STACK
13296 065120  005726                  13$:    TST     (SP)+                   ;RESTORE STACK
13297 065122  012704  003304                  MOV     #TAB5A,R4                       ;POINT TO EXPECTED DATA
13298 065126  004767  052746                  JSR     R7,DATVER                       ;VERIFY DATA
13299 065132  005767  115762                  TST     COUNT
13300 065136  001403                          BEQ     14$                     ;BRANCH IF GOOD
13301 065140  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
13302 065144  104003                          ERROR   .3                      ;FPP ERROR
13303                                                                         ;BAD ACO
13304                                 ;UNDERFLOW WITH TRAPS DISBLED           NON ZERO RESULT
13305                                 ;
13306 065146  012702  000200          14$:    MOV     #200,R2                 ;SETUP FLOATING POINT STATUS
13307 065152  170102                          LDFPS   R2                      ;LOAD FPS
13308 065154  012737  140044  000244          MOV     #WLDTRP,@#FPVEC                 ;RESTORE TRAP VECTOR
13309 065162  012704  003714                  MOV     #TAB41,R4                       ;POINT TO FSRC DATA
13310 065166  012701  003724                  MOV     #TAB42,R1                       ;POINT TO ACO DATA
13311 065172  172411                          LDD     (R1),AC0                        ;LOAD AC0 WITH TEST DATA
13312 065174  172014                          ADDD    (R4),AC0                ;*TEST INSTRUCTION
13313 065176  170203                          STFPS   R3                      ;SAVE FPS
13314 065200  012701  003142                  MOV     #RECDST,R1              ;POINT TO RECEIVED DATA TABLE
13315 065204  174011                          STD     AC0,(R1)                ;SAVE AC0  RESULT
13316 065206  022703  000204                  CMP     #204,R3                 ;VERIFY STATUS
13317 065212  001403                          BEQ     15$                     ;BRANCH IF GOOD
13318 065214  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
13319 065220  104003                          ERROR   .3                      ;FPP ERROR
13320                                                                         ;BAD FPS
13321 065222  012704  003314          15$:    MOV     #TAB6,R4                        ;POINT TO EXPECTED DATA
13322 065226  004767  052646                  JSR     R7,DATVER                       ;VERIFY DATA
13323 065232  005767  115662                  TST     COUNT
13324 065236  001403                          BEQ     16$                     ;BRANCH IF GOOD
13325 065240  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
13326 065244  104003                          ERROR   .3                      ;FPP ERROR
13327                                                                         ;BAD AC0
13328                                 ;UNERFLOW WITH TRAPS ENABLED - NON-ZERO  RESULT
13329                                 ;
13330 065246  012702  102200          16$:    MOV     #102200,R2                      ;SETUP FLOATING POINT STATUS
13331 065252  170102                          LDFPS   R2                      ;LOAD FPS
13332 065254  012737  065306  000244          MOV     #18$,@#FPVEC            ;RESTORE TRAP VECTOR
13333 065262  012704  003714                  MOV     #TAB41,R4                       ;POINT TO FSRC DATA
13334 065266  012701  003724                  MOV     #TAB42,R1                       ;POINT TO AC0 DATA
13335 065272  172411                          LDD     (R1),AC0                        ;LOAD AC0 WITH TEST DATA
13336 065274  172014                          ADDD    (R4),AC0                ;*TEST INSTRUCTION
13337 065276  170000                  17$:    CFCC
13338 065300  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
13339 065304  104003                          ERROR   .3                      ;FPP ERROR
13340                                                                         ;NO TRAP ON UNDERFLOW
13341 065306  170203                  18$:    STFPS   R3                      ;SAVE FPS
13342 065310  012701  003142                  MOV     #RECDST,R1              ;POINT TO RECEIVED DATA TABLE
13343 065314  174011                          STD     AC0,(R1)                ;SAVE AC0  RESULT
13344 065316  012600                          MOV     (SP)+,R0                ;SAVE STACK CONTENTS
13345 065320  005726                          TST     (SP)+                   ;CLEAN UP STACK
13346 065322  022700  065276                  CMP     #17$,R0                 ;VERIFY RETURN ADDRESS
13347 065326  001403                          BEQ     19$                     ;BRANCH IF GOOD
```

D3

FLOATING POINT TESTS

```
13348 065330  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
13349 065334  104003                        ERROR   +3              ;FPP ERROR
13350                                                               ;BAD RETURN ADDRESS
13351 065336  022703  102204          19$:  CMP     #102204,R3                      ;VERIFY STATUS
13352 065342  001403                        BEQ     20$             ;BRANCH IF GOOD
13353 065344  004737  140132                CALL    @#DETFPA        ;DETERMINE FLOATING POINT FAULT. $$$
13354 065350  104003                        ERROR   +3              ;FPP ERROR
13355                                                               ;BAD FPS
13356 065352  012704  003734          20$:  MOV     #TAB43,R4                       ;POINT TO EXPECTED DATA
13357 065356  004767  052516                JSR     R7,DATVER                       ;VERIFY DATA
13358 065362  005767  115532                TST     COUNT
13359 065366  001403                        BEQ     21$             ;BRANCH IF GOOD
13360 065370  004737  140132                CALL    @#DETFPA        ;DETERMINE FLOATING POINT FAULT. $$$
13361 065374  104003                        ERROR   +3              ;FPP ERROR
13362                                                               ;BAD ACO
13363 065376                          21$:
13364                                 ;
13367                                 ;
13368                                 ; - ----------- -- --- ----- ---
13369                                 ;TEST LDCFD, LDCDF
13370                                 ;
13371 065376                          MLDC:
13372                                 ;
13373                                 ;TRUNCATE
13374 065376  012702  000300                MOV     #300,R2                 ;SETUP FLOATING POINT STATUS
13375 065402  170102                        LDFPS   R2                      ;LOAD FPS
13376 065404  012704  003744                MOV     #TAB45,R4                       ;POINT TO FSRC DATA
13377 065410  012701  003314                MOV     #TAB6,R1                        ;POINT TO ACO DATA
13378 065414  172411                        LDD     (R1),AC0                        ;LOAD ACO WITH TEST DATA
13379 065416  177424                        LDCFD   (R4)+,AC0               ;*TEST INSTRUCTION
13380 065420  012701  003142                MOV     #RECDST,R1              ;POINT TO RECEIVED DATA TABLE
13381 065424  174011                        STD     AC0,(R1)                ;SAVE ACO  RESULT
13382 065426  022704  003750                CMP     #TAB45+4,R4             ;VERIFY AUTO-INC
13383 065432  001403                        BEQ     1$              ;BRANCH IF GOOD AUTO-INC
13384 065434  004737  140132                CALL    @#DETFPA        ;DETERMINE FLOATING POINT FAULT. $$$
13385 065440  104003                        ERROR   +3              ;FPP ERROR
13386                                                               ;BAD AUTO-INC
13387 065442  012704  003754          1$:   MOV     #TAB46,R4                       ;POINT TO EXPECTED DATA
13388 065446  004767  052426                JSR     R7,DATVER                       ;VERIFY DATA
13389 065452  005767  115442                TST     COUNT
13390 065456  001403                        BEQ     2$              ;BRANCH IF GOOD
13391 065460  004737  140132                CALL    @#DETFPA        ;DETERMINE FLOATING POINT FAULT. $$$
13392 065464  104003                        ERROR   +3              ;FPP ERROR
13393                                                               ;BAD ACO
13394                                 ;AUTO-INC DOUBLE MODE
13395                                 ;
13396 065466  005002                  2$:   CLR     R2                      ;SETUP FLOATING POINT STATUS
13397 065470  170102                        LDFPS   R2                      ;LOAD FPS
13398 065472  012704  003744                MOV     #TAB45,R4                       ;POINT TO FSRC DATA
13399 065476  012701  003464                MOV     #TAB16,R1                       ;POINT TO ACO DATA
13400 065502  172411                        LDD     (R1),AC0                        ;LOAD ACO WITH TEST DATA
13401 065504  177424                        LDCDF   (R4)+,AC0               ;*TEST INSTRUCTION
13402 065506  020427  003754                CMP     R4,#TAB45+10            ;VERIFY AUTO-INC
13403 065512  001403                        BEQ     3$              ;BRANCH IF GOOD
13404 065514  004737  140132                CALL    @#DETFPA        ;DETERMINE FLOATING POINT FAULT. $$$
13405 065520  104003                        ERROR   +3              ;FPP ERROR
13406                                                               ;BAD AUTO INC ON DOUBLE
```

E3

FLOATING POINT TESTS

```
13407 065522  170203                3$:     STFPS   R3              ;SAVE FPS
13408 065524  012701  003142                MOV     #RECDST,R1      ;POINT TO RECEIVED DATA TABLE
13409 065530  174011                        STD     AC0,(R1)        ;SAVE AC0  RESULT
13410 065532  022703  000000                CMP     #0,R3           ;VERIFY STATUS
13411 065536  001403                        BEQ     4$              ;BRANCH IF GOOD
13412 065540  004737  140132                CALL    @#DETFPA        ;DETERMINE FLOATING POINT FAULT. $$$
13413 065544  104003                        ERROR   +3              ;FPP ERROR
13414                                                               ;BAD FPS
13415 065546  012704  004014          4$:     MOV     #TAB49,R4               ;POINT TO EXPECTED DATA
13416 065552  004767  052322                JSR     R7,DATVER               ;VERIFY DATA
13417 065556  005767  115336                TST     COUNT
13418 065562  001403                        BEQ     5$              ;BRANCH IF GOOD
13419 065564  004737  140132                CALL    @#DETFPA        ;DETERMINE FLOATING POINT FAULT. $$$
13420 065570  104003                        ERROR   +3              ;FPP ERROR
13421                                                               ;BAD AC0
13422
13423                               ;LDCFD GR7
                                    ;
13424 065572  012702  000200          5$:     MOV     #200,R2         ;SETUP FLOATING POINT STATUS
13425 065576  170102                        LDFPS   R2              ;LOAD FPS
13426 065600  005003                        CLR     R3
13427 065602  177427  043243                LDCFD   #5203,AC0       ;*TEST INSTRUCTION
13428 065606  005203                        INC     R3
13429 065610  005203                        INC     R3
13430 065612  005203                        INC     R3              ;IF LDCFD WORKED, R3 SHOULD=3
13431 065614  022703  000003                CMP     #3,R3           ;VERIFY CORRECT PROGRAM FLOW
13432 065620  001403                        BEQ     6$              ;BRANCH IF GOOD
13433 065622  004737  140132                CALL    @#DETFPA        ;DETERMINE FLOATING POINT FAULT. $$$
13434 065626  104003                        ERROR   +3              ;FPP ERROR
13435                                                               ;BAD PROGRAM FLOW
13436
13437                               ;NEGATIVE OPERANDS
                                    ;
13438 065630  012702  000200          6$:     MOV     #200,R2         ;SETUP FLOATING POINT STATUS
13439 065634  170102                        LDFPS   R2              ;LOAD FPS
13440 065636  012704  003764                MOV     #TAB47,R4               ;POINT TO FSRC DATA
13441 065642  012701  003744                MOV     #TAB45,R1               ;POINT TO AC0 DATA
13442 065646  172411                        LDD     (R1),AC0                ;LOAD AC0 WITH TEST DATA
13443 065650  177414                        LDCFD   (R4),AC0        ;*TEST INSTRUCTION
13444 065652  170203                        STFPS   R3              ;SAVE FPS
13445 065654  012701  003142                MOV     #RECDST,R1      ;POINT TO RECEIVED DATA TABLE
13446 065660  174011                        STD     AC0,(R1)        ;SAVE AC0  RESULT
13447 065662  022703  000210                CMP     #210,R3         ;VERIFY STATUS
13448 065666  001403                        BEQ     7$              ;BRANCH IF GOOD
13449 065670  004737  140132                CALL    @#DETFPA        ;DETERMINE FLOATING POINT FAULT. $$$
13450 065674  104003                        ERROR   +3              ;FPP ERROR
13451                                                               ;BAD FPS
13452 065676  012704  004004          7$:     MOV     #TAB48,R4               ;POINT TO EXPECTED DATA
13453 065702  004767  052172                JSR     R7,DATVER               ;VERIFY DATA
13454 065706  005767  115206                TST     COUNT
13455 065712  001403                        BEQ     8$              ;BRANCH IF GOOD
13456 065714  004737  140132                CALL    @#DETFPA        ;DETERMINE FLOATING POINT FAULT. $$$
13457 065720  104003                        ERROR   +3              ;FPP ERROR
13458                                                               ;BAD AC0
13459
13460                               ;LOAD A ZERO
                                    ;
13461 065722  012702  000200          8$:     MOV     #200,R2         ;SETUP FLOATING POINT STATUS
13462 065726  170102                        LDFPS   R2              ;LOAD FPS
13463 065730  012704  003314                MOV     #TAB6,R4                ;POINT TO FSRC DATA
```

F3

FLOATING POINT TESTS

```
13464 065734  012701  004004                MOV     #TAB48,R1                       ;POINT TO ACO DATA
13465 065740  172411                        LDD     (R1),AC0                        ;LOAD AC0 WITH TEST DATA
13466 065742  177414                        LDCFD   (R4),AC0                        ;*TEST INSTRUCTION
13467 065744  170203                        STFPS   R3                      ;SAVE FPS
13468 065746  012701  003142                MOV     #RECDST,R1              ;POINT TO RECEIVED DATA TABLE
13469 065752  174011                        STD     AC0,(R1)                ;SAVE AC0  RESULT
13470 065754  022703  000204                CMP     #204,R3                 ;VERIFY STATUS
13471 065760  001403                        BEQ     9$                      ;BRANCH IF GOOD
13472 065762  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
13473 065766  104003                        ERROR   +3                      ;FPP ERROR
13474                                                                       ;BAD FPS
13475 065770  012704  003314          9$:   MOV     #TAB6,R4                        ;POINT TO EXPECTED DATA
13476 065774  004767  052100                JSR     R7,DATVER                       ;VERIFY DATA
13477 066000  005767  115114                TST     COUNT
13478 066004  001403                        BEQ     10$                     ;BRANCH IF GOOD
13479 066006  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
13480 066012  104003                        ERROR   +3                      ;FPP ERROR
13481                                                                       ;BAD AC0
13482 066014                          10$:
13483                                 ;
13486                                 ;
13487                                 ;-------------------------- ----
13488                                 ;TEST CMPD
13489                                 ;
13490 066014                          MCMPD:
13491                                 ;
13492                                 ;  CMPD WITH FRSC=AC0=0
13493                                 ;
13494 066014  005037  003030                CLR     @#FLAG                  ;SIGNAL THAT AC0 REMAINS CONSTANT
13495 066020  004767  000152                JSR     R7,CMPRTN               ;ROUTINE TO TEST DATA
13496 066024  000000  000000  000000        .WORD   0,0,0,0                 ;AC0 AT START
      066032  000000
13497 066034  000000  000000  000000        .WORD   0,0,0,0                 ;FSRC AT START
      066042  000000
13498 066044  000200                        .WORD   200                     ;FPS AT START (D)
13499 066046  000204                        .WORD   204                     ;FPS AT END
13500                                 ;  CMPD WITH EXP[FSRC]=0, EXP[AC0]=0
13501 066050  012737  000001  003030        MOV     #1,@#FLAG               ;SIGNAL THAT AC0 WILL = 0
13502 066056  004767  000114                JSR     R7,CMPRTN               ;ROUTINE TO TEST DATA
13503 066062  000000  000000  000000        .WORD   0,0,0,125252            ;AC0 AT START
      066070  125252
13504 066072  000100  000022  000123        .WORD   100,22,123,123          ;FSRC AT START
      066100  000123
13505 066102  000200                        .WORD   200                     ;FPS AT START (D)
13506 066104  000204                        .WORD   204                     ;FPS AT END
13507                                 ;  CMPD FSRC>EXP[AC0]=0
13508 066106  005037  003030                CLR     @#FLAG                  ;AC0 REMAINS UNCHANGED
13509 066112  004767  000060                JSR     R7,CMPRTN               ;ROUTINE TO TEST DATA
13510 066116  000400  012346  012346        .WORD   400,12346,12346,23      ;AC0 AT START
      066124  000023
13511 066126  000200  000000  000000        .WORD   200,0,0,0               ;FSRC AT START
      066134  000000
1351. 066136  000200                        .WORD   200                     ;FPS AT START (D)
13513 066140  000210                        .WORD   210                     ;FPS AT END
13514                                 ; CMPD FSRC=AC0>0
13515 066142  004767  000030                JSR     R7,CMPRTN               ;ROUTINE TO TEST DATA
13516 066146  077777  177777  177777        .WORD   77777,-1,-1,-1          ;AC0 AT START
```

FLOATING POINT TESTS

```
        066154  177777
13517   066156  077777  177777  177777          .WORD   77777,-1,-1,-1          ;FSRC AT START
        066164  177777
13518   066166  000200                          .WORD   200                     ;FPS AT START (D)
13519   066170  000204                          .WORD   204                     ;FPS AT END
13520   066172  000167  000126                 JMP     HOP44                   ;HOP OVER SUBROUTINE
13521
13522                                   ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
13523                                   ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
13524                                   ;COMPARE ROUTINE DATA TABLES
13525                                   ;
13526                                   ;               AC0
13527                                   ;               FSRC
13528                                   ;               FPS BEFORE EXECUTION
13529                                   ;               FPS AFTER EXECUTION
13530                                   ;               (FEC)
13531                                   ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
13532                                   ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
13533                                   ;
13534   066176  012605                  CMPRTN: MOV     (SP)+,R5                ; RETURN ADDRESS TO USE AS POINTER
13535   066200  012702  000200                  MOV     #200,R2                 ;SET TO DOUBLE MODE FOR LOAD
13536   066204  170102                          LDFPS   R2                      ;LOAD FPS
13537   066206  010504                          MOV     R5,R4                   ;POINT TO FSRC DATA
13538   066210  062704  000010                  ADD     #10,R4                  ;
13539   066214  010501                          MOV     R5,R1           ;POINT TO AC0 DATA
13540   066216  172411                          LDD     (R1),AC0                ;LOAD AC0 WITH TEST DATA
13541   066220  016502  000020                  MOV     20(R5),R2               ;GET TEST FPS
13542   066224  170102                          LDFPS   R2                      ;LOAD TEST FPS
13543   066226  173414                  1$:     CMPD    (R4),AC0                ;*TEST INSTRUCTION
13544   066230  170203                          STFPS   R3                      ;SAVE FPS
13545   066232  012702  000200                  MOV     #200,R2         ;SET FPP TO DOUBLE
13546   066236  170102                          LDFPS   R2
13547   066240  012701  003142                  MOV     #RECDST,R1              ;POINT TO RECEIVED DATA TABLE
13548   066244  174011                          STD     AC0,(R1)                ;SAVE AC0  RESULT
13549   066246  026503  000022                  CMP     22(R5),R3               ;VERIFY STATUS
13550   066252  001403                          BEQ     2$                      ;BRANCH IF GOOD
13551   066254  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
13552   066260  104003                          ERROR   +3              ;FPP ERROR
13553                                           ;BAD FPS
13554   066262  005737  003030          2$:     TST     @#FLAG                  ;SEE IF AC0 REMAINS UNCHANGED
13555   066266  001403                          BEQ     3$                      ;BRANCH IF AC0 STAYS THE SAME
13556   066270  012704  003314                  MOV     #TAB6,R4                ;AC0=0
13557   066274  000401                          BR      4$                      ;GO VERIFY  DATA
13558   066276  010504                  3$:     MOV     R5,R4                   ;POINT TO EXPECTED DATA
13559   066300  004767  051574          4$:     JSR     R7,DATVER                       ;VERIFY DATA
13560   066304  005767  114610                  TST     COUNT
13561   066310  001403                          BEQ     5$                      ;BRANCH IF GOOD
13562   066312  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
13563   066316  104003                          ERROR   +3              ;FPP ERROR
13564                                           ;BAD AC0
13565   066320  000165  000024          5$:     JMP     24(R5)                  ;RETURN
13566   066324                          HOP44:
13567                                   ;
13570                                   ;
13571                                   ;-------------  ---------- ------
13572                                   ;TEST DIVF
13573                                   ;
```

H3

FLOATING POINT TESTS

```
13574 066324                                    MDIVF:
13575                                           ;
13576                                           ;1/EXP[AC]=FSRC=0
13577 066324  012737  000002  003030                    MOV     #2,@#FLAG               ;NO INTERRUPT, BUT FEC
13578 066332  004767  000706                            JSR     R7,DVFSUB               ;DO TEST
13579 066336  000100  000027                            .WORD   100,27          ;AC0
13580 066342  000000  000000                            .WORD   0,0             ;FSRC
13581 066346  000100  000027                            .WORD   100,27          ;RESULT
13582 066352  040000                                    .WORD   40000           ; TEST FPS
13583 066354  140000                                    .WORD   140000          ;RESULT FPS
13584 066356  000004                                    .WORD   4               ;FEC
13585                                           ;2/AC-EXP[FSRC]=0
13586                                           ;TRAPS ENABLED
13587 066360  012737  000001  003030                    MOV     #1,@#FLAG               ;INTERRUPT
13588 066366  004767  000652                            JSR     R7,DVFSUB               ;DO TEST
13589 066372  000000  000000                            .WORD   0,0             ;AC0
13590 066376  000100  000000                            .WORD   100,0           ;FSRC
13591 066402  000000  000000                            .WORD   0,0             ;RESULT
13592 066406  000000                                    .WORD   0               ; TEST FPS
13593 066410  100000                                    .WORD   100000          ;RESULT FPS
13594 066412  000004                                    .WORD   4               ;FEC
13595                                           ;3/FSRC>AC0=0
13596 066414  005037  003030                            CLR     @#FLAG          ;NO INTERRUPT
13597 066420  004767  000620                            JSR     R7,DVFSUB               ;DO TEST
13598 066424  000177  000234                            .WORD   177,234         ;AC0
13599 066430  004100  000000                            .WORD   4100,0          ;FSRC
13600 066434  000000  000000                            .WORD   0,0             ;RESULT
13601 066440  007400                                    .WORD   7400            ; TEST FPS
13602 066442  007404                                    .WORD   7404            ;RESULT FPS
13603                                           ;4/AC0>EXP[FSRC]=0
13604 066444  012737  000001  003030                    MOV     #1,@#FLAG               ;INTERRUPT
13605 066452  004767  000566                            JSR     R7,DVFSUB               ;DO TEST
13606 066456  040200  104210                            .WORD   40200,104210    ;AC0
13607 066462  000125  025252                            .WORD   125,25252       ;FSRC
13608 066466  040200  104210                            .WORD   40200,104210    ;RESULT
13609 066472  007557                                    .WORD   7557            ; TEST FPS
13610 066474  107557                                    .WORD   107557          ;RESULT FPS
13611 066476  000004                                    .WORD   4               ;FEC
13612                                           ;5/EXP[AC]=EXP[FSRC]
13613 066500  005037  003030                            CLR     @#FLAG          ;NO INTERRUPT
13614 066504  004767  000534                            JSR     R7,DVFSUB               ;DO TEST
13615 066510  077760  177777                            .WORD   77760,-1                ;AC0
13616 066514  077760  000000                            .WORD   77760,0         ;FSRC
13617 066520  040200  104210                            .WORD   40200,104210    ;RESULT
13618 066524  007414                                    .WORD   7414            ; TEST FPS
13619 066526  007400                                    .WORD   7400            ;RESULT FPS
13620                                           ;6/AC=FSRC
13621 066530  005037  003030                            CLR     @#FLAG          ;NO INTERRUPT
13622 066534  004767  000504                            JSR     R7,DVFSUB               ;DO TEST
13623 066540  052525  052525                            .WORD   52525,52525             ;AC0
13624 066544  052525  052525                            .WORD   52525,52525             ;FSRC
13625 066550  040200  000000                            .WORD   40200,0         ;RESULT
13626 066554  007400                                    .WORD   7400            ; TEST FPS
13627 066556  007400                                    .WORD   7400            ;RESULT FPS
13628                                           ;7/FSRC>0<AC0, ROUND
13629 066560  005037  003030                            CLR     @#FLAG          ;NO INTERRUPT
13630 066564  004767  000454                            JSR     R7,DVFSUB               ;DO TEST
```

FLOATING POINT TESTS

```
13631 066570  077777  125252                .WORD   77777,125252    ;AC0
13632 066574  040300  000000                .WORD   40300,0         ;FSRC
13633 066600  077652  070707                .WORD   77652,070707            ;RESULT
13634 066604  007400                        .WORD   7400            ; TEST FPS
13635 066606  007400                        .WORD   7400            ;RESULT FPS
13636                             ;8/AC>0<FSRC
13637 066610  005037  003030                CLR     @#FLAG          ;NO INTERRUPT
13638 066614  004767  000424                JSR     R7,DVFSUB               ;DO TEST
13639 066620  055377  177777                .WORD   55377,-1        ;AC0
13640 066624  055300  000000                .WORD   55300,0         ;FSRC
13641 066630  040252  125252                .WORD   40252,125252    ;RESULT
13642 066634  000000                        .WORD   0               ; TEST FPS
13643 066636  000000                        .WORD   0               ;RESULT FPS
13644                             ;9/FSRC>AC>0
13645 066640  005037  003030                CLR     @#FLAG          ;NO INTERRUPT
13646 066644  004767  000374                JSR     R7,DVFSUB               ;DO TEST
13647 066650  064600  000001                .WORD   64600,1         ;AC0
13648 066654  066600  000000                .WORD   66600,0         ;FSRC
13649 066660  036200  000001                .WORD   36200,1         ;RESULT
13650 066664  000000                        .WORD   0               ; TEST FPS
13651 066666  000000                        .WORD   0               ;RESULT FPS
13652                             ;10/AC>FSRC>0
13653 066670  005037  003030                CLR     @#FLAG          ;NO INTERRUPT
13654 066674  004767  000344                JSR     R7,DVFSUB               ;DO TEST
13655 066700  012345  156024                .WORD   12345,156024    ;AC0
13656 066704  005600  000000                .WORD   05600,0         ;FSRC
13657 066710  044745  156024                .WORD   44745,156024            ;RESULT
13658 066714  000017                        .WORD   17              ; TEST FPS
13659 066716  000000                        .WORD   0               ;RESULT FPS
13660                             ;11/FSRC<0
13661 066720  005037  003030                CLR     @#FLAG          ;NO INTERRUPT
13662 066724  004767  000314                JSR     R7,DVFSUB               ;DO TEST
13663 066730  040422  101010                .WORD   40422,101010    ;AC0
13664 066734  140511  101010                .WORD   140511,101010   ;FSRC
13665 066740  140072  020167                .WORD   140072,20167            ;RESULT
13666 066744  000057                        .WORD   57              ; TEST FPS
13667 066746  000050                        .WORD   50              ;RESULT FPS
13668                             ;12/AC<0
13669 066750  005037  003030                CLR     @#FLAG          ;NO INTERRUPT
13670 066754  004767  000264                JSR     R7,DVFSUB               ;DO TEST
13671 066760  160077  000101                .WORD   160077,101      ;AC0
13672 066764  040417  177777                .WORD   40417,-1        ;FSRC
13673 066770  157651  143527                .WORD   157651,143527           ;RESULT
13674 066774  000007                        .WORD   7               ; TEST FPS
13675 066776  000010                        .WORD   10              ;RESULT FPS
13676                             ;13/TRUNCATE TEST
13677 067000  005037  003030                CLR     @#FLAG          ;NO INTERRUPT
13678 067004  004767  000234                JSR     R7,DVFSUB               ;DO TEST
13679 067010  060100  000177                .WORD   60100,177       ;AC0
13680 067014  040300  000000                .WORD   40300,0         ;FSRC
13681 067020  060000  000124                .WORD   60000,124               ;RESULT
13682 067024  000040                        .WORD   40              ; TEST FPS
13683 067026  000040                        .WORD   40              ;RESULT FPS
13684                             ;14/ROUND TEST
13685 067030  005037  003030                CLR     @#FLAG          ;NO INTERRUPT
13686 067034  004767  000204                JSR     R7,DVFSUB               ;DO TEST
13687 067040  060100  000177                .WORD   60100,177       ;AC0
```

```
067044  040300  000000                  .WORD   40300,0 ;FSRC
067050  060000  000125                  .WORD   60000,125               ;RESULT
067054  000000                          .WORD   0               ; TEST FPS
067056  000000                          .WORD   0               ;RESULT FPS
                                ;15/OVERFLOW, INTERRUPTS ENABLED
067060  012737  000001  003030          MOV     #1,@#FLAG               ;INTERRUPT
067066  004767  000152                  JSR     R7,DVFSUB               ;DO TEST
067072  177700  000000                  .WORD   177700,0                ;AC0
067076  000200  000000                  .WORD   200,0           ;FSRC
067102  137700  000000                  .WORD   137700,0                ;RESULT
067106  001100                          .WORD   1100            ; TEST FPS
067110  101112                          .WORD   101112          ;RESULT FPS
067112  000010                          .WORD   10              ;FEC
                                ;16/OVERFLOW, TRAPS DISABLED
067114  012737  000002  003030          MOV     #2,@#FLAG               ;NO INTERRUPT
067122  004767  000116                  JSR     R7,DVFSUB               ;DO TEST
067126  000200  000000                  .WORD   200,0           ;AC0
067132  177700  000000                  .WORD   177700,0                ;FSRC
067136  000000  000000                  .WORD   0,0             ;RESULT
067142  041100                          .WORD   41100           ; TEST FPS
067144  041104                          .WORD   41104           ;RESULT FPS
067146  000010                          .WORD   10              ;FEC OVERFLOW
                                ;17/UNDERFLOW, TRAPS ENABLED, UV RESULT
067150  012737  000001  003030          MOV     #1,@#FLAG               ;INTERRUPT
067156  004767  000062                  JSR     R7,DVFSUB               ;DO TEST
067162  100200  000000                  .WORD   100200,0                ;AC0
067166  040377  177777                  .WORD   40377,-1                ;FSRC
067172  100000  000001                  .WORD   100000,1                ;RESULT
067176  002000                          .WORD   2000            ; TEST FPS
067200  102014                          .WORD   102014          ;RESULT FPS
067202  000012                          .WORD   12              ;FEC
                                ;18/UNDERFLOW,TRAPS ENABLED, ROUND
067204  012737  000001  003030          MOV     #1,@#FLAG               ;INTERRUPT
067212  004767  000026                  JSR     R7,DVFSUB               ;DO TEST
067216  030325  025252                  .WORD   30325,25252             ;AC0
067222  076777  023456                  .WORD   76777,23456             ;FSRC
067226  071525  157716                  .WORD   71525,157716            ;RESULT
067232  002537                          .WORD   2537            ; TEST FPS
067234  102500                          .WORD   102500          ;RESULT FPS
067236  000012                          .WORD   12              ;FEC
                                ;
067240  000167  000242                  JMP     HOP10           ;GO TO NEXT TEST
                                ;*X*X*X*X*X*X*Y*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
                                ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
                                ;DIVF SUBROUTINE:
                                ;                       AC0
                                ;                       FSRC
                                ;                       FPS BEFORE EXECUTION
                                ;                       FPS AFTER EXECUTION
                                ;                       (FEC)
                                ;
                                ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
                                ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
                                ;
067244  012605                  DVFSUB: MOV     (SP)+,R5                ; RETURN ADDRESS TO USE AS POINTER
067246  012737  067326  000244          MOV     #50$,@#FPVEC            ;REDIRECT TRAP VECTOR
067254  012702  000200                  MOV     #200,R2                 ;SET TO DOUBLE MODE FOR LOAD
```

FLOATING POINT TESTS

```
13745 067260  170102                        LDFPS   R2                      ;LOAD FPS
13746 067262  010504                        MOV     R5,R4                   ;POINT TO FSRC DATA
13747 067264  062704  000004                ADD     #4,R4
13748 067270  172415                        LDD     (R5),AC0                ;LOAD AC0 WITH TEST DATA
13749 067272  016502  000014                MOV     14(R5),R2               ;GET TEST FPS
13750 067276  170102                        LDFPS   R2                      ;LOAD TEST FPS
13751                               ;
13752 067300  174414                        DIVF    (R4),AC0                ;*TEST INSTRUCTION
13753 067302  170001                1$:     SETF                            ;WAIT FOR POSSIBLE FPA TRAP.
13754                               ;
13755                               ;INSTRUCTION DIDNT TRAP
13756                               ;
13757 067304  032737  000001  003030        BIT     #1,@#FLAG                       ;VERIFY A NO TRAP CONDITION
13758 067312  001426                        BEQ     2$                      ;BRANCH IF GOOD
13759 067314  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
13760 067320  104003                        ERROR   +3                  ;FPP ERROR
13761                                                                   ;INSTRUCTION SHOULD HAVE TRAPPED
13762 067322  000167  000042                JMP     2$                      ;REJOIN CODE
13763                               ;
13764                               ;INSTRUCTION TRAPPED
13765 067326  032737  000001  003030  50$:   BIT     #1,@#FLAG               ;SEE IF EXPECTING A TRAP
13766 067334  001005                        BNE     51$                     ;BRANCH IF EXPECTING A TRAP
13767 067336  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
13768 067342  104003                        ERROR   +3                  ;FPP ERROR
13769                                                                   ;INSTRUCTION WASNT SUPPOSE TO TRAP
13770 067344  000167  000020                JMP     2$                      ;REJOIN CODE
13771 067350  012604                51$:    MOV     (SP)+,R4                ;SEE IF PC = INSTRUCTION
13772 067352  005726                        TST     (SP)+                   ;CLEAN UP STACK
13773 067354  022704  067302                CMP     #1$,R4                  ;
13774 067360  001403                        BEQ     2$                      ;BRANCH IF GOOD COMPARE
13775 067362  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
13776 067366  104003                        ERROR   +3                  ;FPP ERROR
13777                                                                   ;PC WAS INCORRECT
13778                               ;
13779                               ;COMMON CODE FOR TRAP AND NO TRAP
13780                               ;
13781 067370  170203                2$:     STFPS   R3                      ;SAVE FPS
13782 067372  012702  000200                MOV     #200,R2                 ;SET FPP TO DOUBLE
13783 067376  170102                        LDFPS   R2
13784 067400  012701  003142                MOV     #RECDST,R1              ;POINT TO RECEIVED DATA TABLE
13785 067404  174011                        STD     AC0,(R1)                ;SAVE AC0  RESULT
13786 067406  026503  000016                CMP     16(R5),R3               ;VERIFY STATUS
13787 067412  001403                        BEQ     3$                      ;BRANCH IF GOOD
13788 067414  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
13789 067420  104003                        ERROR   +3                  ;FPP ERROR
13790                                                                   ;BAD FPS
13791 067422  010504                3$:     MOV     R5,R4                   ;POINT TO EXPECTED DATA
13792 067424  062704  000010                ADD     #10,R4
13793 067430  004767  050426          4$:   JSR     R7,DATVFR                       ;VERIFY DATA
13794 067434  005767  113460                TST     COUNT
13795 067440  001403                        BEQ     5$                      ;BRANCH IF GOOD
13796 067442  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
13797 067446  104003                        ERROR   +3                  ;FPP ERROR
13798                                                                   ;BAD AC0
13799 067450  005737  003030          5$:   TST     @#FLAG                  ;SEE IF NEED TO CHECK FEC
13800 067454  001002                        BNE     6$                      ;BRANCH IF NEED TO CHECK
13801 067456  000165  000020                JMP     20(R5)                  ;RETURN FROM TEST
```

FLOATING POINT TESTS

```
13802 067462  170301                  6$:     STST    R1                      ;SAVE FEC
13803 067464  016504  000020                  MOV     20(R5),R4               ;GET FEC
13804 067470  020401                          CMP     R4,R1                   ;VERIFY FEC
13805 067472  001403                          BEQ     7$                      ;BRANCH IF GOOD
13806 067474  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
13807 067500  104003                          ERROR   +3              ;FPP ERROR
13808                                                                 ;BAD FEC
13809 067502  000165  000022          7$:     JMP     22(R5)                  ;RETURN FROM TEST
13810                                 ;
13811 067506                          HOP10:
13814                                 ;
13815                                 ;-- --------- ------   -----   ---
13816                                 ;TEST DIVD
13817                                 ;
13818 067506                          MDIVD:
13819                                 ;
13820                                 ;1/AC=FSRC=0 TRAPS DISABLED
13821 067506  012737  000002  003030          MOV     #2,@#FLAG               ;NO INTERRUPT
13822 067514  004767  000516                  JSR     R7,DVDSUB               ;DO TEST
13823 067520  000000  000000  000000          .WORD   0,0,0,1         ;AC0
      067526  000001
13824 067530  000100  000000  000000          .WORD   100,0,0,0       ;FSRC
      067536  000000
13825 067540  000000  000000  000000          .WORD   0,0,0,1         ;RESULT
      067546  000001
13826 067550  040000                          .WORD   40000           ; TEST FPS
13827 067552  140000                          .WORD   140000          ;RESULT FPS
13828 067554  000004                          .WORD   4               ;FEC
13829                                 ;2/FSRC=0, TRAPS ENABLED
13830 067556  012737  000001  003030          MOV     #1,@#FLAG               ;INTERRUPT
13831 067564  004767  000446                  JSR     R7,DVDSUB               ;DO TEST
13832 067570  000402  000000  000000          .WORD   402,0,0,0       ;AC0
      067576  000000
13833 067600  000000  000000  000000          .WORD   0,0,0,0         ;FSRC
      067606  000000
13834 067610  000402  000000  000000          .WORD   402,0,0,0               ;RESULT
      067616  000000
13835 067620  000200                          .WORD   200             ; TEST FPS
13836 067622  100200                          .WORD   100200          ;RESULT FPS
13837 067624  000004                          .WORD   4                       ;FEC
13838                                 ;3/ROUND
13839 067626  005037  003030                  CLR     @#FLAG          ;NO INTERRUPT
13840 067632  004767  000400                  JSR     R7,DVDSUB               ;DO TEST
13841 067636  034300  000000  000000          .WORD   34300,0,0,1     ;AC0
      067644  000001
13842 067646  140300  000000  000000          .WORD   140300,0,0,0    ;FSRC
      067654  000000
13843 067656  134200  000000  000000          .WORD   134200,0,0,1    ;RESULT
      067664  000001
13844 067666  000200                          .WORD   200             ; TEST FPS
13845 067670  000210                          .WORD   210             ;RESULT FPS
13846                                 ;4/TRUNCATE
13847 067672  005037  003030                  CLR     @#FLAG          ;NO INTERRUPT
13848 067676  004767  000334                  JSR     R7,DVDSUB               ;DO TEST
13849 067702  034300  000000  000000          .WORD   34300,0,0,1     ;AC0
      067710  000001
13[illegible] 067712  140300  000000  000000          .WORD   140300,0,0,0    ;FSRC
```

M3

FLOATING POINT TESTS

```
                067720  000000
13851 067722    134200  000000  000000          .WORD   134200,0,0,0    ;RESULT
                067730  000000
13852 067732    000240                          .WORD   240             ; TEST FPS
13853 067734    000250                          .WORD   250             ;RESULT FPS
13854                           ;5/ROUND NEGATIVE AC, FSRC
13855 067736    005037  003030                  CLR     @#FLAG          ;NO INTERRUPT
13856 067742    004767  000270                  JSR     R7,DVDSUB               ;DO TEST
13857 067746    177642  000000  000000          .WORD   177642,0,0,151  ;AC0
                067754  000151
13858 067756    166600  000000  000000          .WORD   166600,0,0,123  ;FSRC
                067764  000123
13859 067766    051242  000000  000000          .WORD   51242,0,0,0             ;RESULT
                067774  000000
13860 067776    000200                          .WORD   200             ; TEST FPS
13861 070000    000200                          .WORD   200             ;RESULT FPS
13862                           ;6/TRUNCATE NEAGTIVE AC, FSRC
13863 070002    005037  003030                  CLR     @#FLAG          ;NO INTERRUPT
13864 070006    004767  000224                  JSR     R7,DVDSUB               ;DO TEST
13865 070012    177642  000000  000000          .WORD   177642,0,0,151  ;AC0
                070020  000151
13866 070022    166600  000000  000000          .WORD   166600,0,0,123  ;FSRC
                070030  000123
13867 070032    051241  177777  177777          .WORD   51241,-1,-1,-1          ;RESULT
                070040  177777
13868 070042    000240                          .WORD   240             ; TEST FPS
13869 070044    000240                          .WORD   240             ;RESULT FPS
13870                           ;7/AC=FSRC
13871 070046    005037  003030                  CLR     @#FLAG          ;NO INTERRUPT
13872 070052    004767  000160                  JSR     R7,DVDSUB               ;DO TEST
13873 070056    055521  047621  100333          .WORD   55521,47621,100333,-1   ;AC0
                070064  177777
13874 070066    055521  047621  100333          .WORD   55521,47621,100333,-1   ;FSRC
                070074  177777
13875 070076    040200  000000  000000          .WORD   40200,0,0,0             ;RESULT
                070104  000000
13876 070106    007717                          .WORD   7717            ; TEST FPS
13877 070110    007700                          .WORD   7700            ;RESULT FPS
13878                           ;8/UNDERFLOW TRAPS ENABLED, UV RESULT
13879 070112    012737  000001  003030          MOV     #1,@#FLAG               ;INTERRUPT
13880 070120    004767  000112                  JSR     R7,DVDSUB               ;DO TEST
13881 070124    100200  000000  000000          .WORD   100200,0,0,0            ;AC0
                070132  000000
13882 070134    077777  000000  000000          .WORD   77777,0,0,0             ;FSRC
                070142  000000
13883 070144    140400  100200  100200          .WORD   140400,100200,100200,100201     ;RESULT
                070152  100201
13884 070154    002200                          .WORD   2200            ; TEST FPS
13885 070156    102210                          .WORD   102210          ;RESULT FPS
13886 070160    000012                          .WORD   12              ;FEC
13887                           ;9/OVERFLOW TRAPS ENABLED
13888 070162    012737  000001  003030          MOV     #1,@#FLAG               ;INTERRUPT
13889 070170    004767  000042                  JSR     R7,DVDSUB               ;DO TEST
13890 070174    077000  123465  012346          .WORD   77000,123465,12346,525          ;AC0
                070202  000525
13891 070204    000303  000001  140000          .WORD   303,1,140000,140001             ;FSRC
                070212  140001
```

FLOATING POINT TESTS

```
13892 070214  036650  163002  103645          .WORD   36650,163002,103645,64003       ;RESULT
      070222  064003
13893 070224  001700                          .WORD   1700                ; TEST FPS
13894 070226  101702                          .WORD   101702              ;RESULT FPS
13895 070230  000010                          .WORD   10                  ;FEC
13896                                 ;
13897 070232  000167  000242                    JMP     HOP11                       ;HOP OVER SUBROUTINE
13898                                 ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
13899                                 ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
13900                                 ;
13901                                 ;DIVD SUBROUTINE:
13902                                 ;               AC0
13903                                 ;               FSRC
13904                                 ;               FPS BEFORE EXECUTION
13905                                 ;               FPS AFTER EXECUTION
13906                                 ;               (FEC)
13907                                 ;
13908                                 ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
13909                                 ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
13910                                 ;
13911 070236  012605                  DVDSUB: MOV     (SP)+,R5                    ; RETURN ADDRESS TO USE AS POINTER
13912 070240  012737  070320  000244          MOV     #50$,@#FPVEC                ;REDIRECT TRAP VECTOR
13913 070246  012702  000200                  MOV     #200,R2                     ;SET TO DOUBLE MODE FOR LOAD
13914 070252  170102                          LDFPS   R2                          ;LOAD FPS
13915 070254  010504                          MOV     R5,R4                       ;POINT TO FSRC DATA
13916 070256  062704  000010                  ADD     #10,R4
13917 070262  172415                          LDD     (R5),AC0                    ;LOAD AC0 WITH TEST DATA
13918 070264  016502  000030                  MOV     30(R5),R2                   ;GET TEST FPS
13919 070270  170102                          LDFPS   R2                          ;LOAD TEST FPS
13920                                 ;
13921 070272  174414                          DIVD    (R4),AC0                    ;*TEST INSTRUCTION
13922 070274  170000                  1$:     CFCC                                ;WAIT FOR POSSIBLE FPA TRAP.
13923                                 ;
13924                                 ;INSTRUCTION DIDNT TRAP
13925                                 ;
13926 070276  032737  000001  003030          BIT     #1,@#FLAG                           ;VERIFY A NO TRAP CONDITION
13927 070304  001426                          BEQ     2$                          ;BRANCH IF GOOD
13928 070306  004737  140132                  CALL    @#DETFPA                    ;DETERMINE FLOATING POINT FAULT. $$$
13929 070312  104003                          ERROR   +3                  ;FPP ERROR
13930                                                                     ;INSTRUCTION SHOULD HAVE TRAPPED
13931 070314  000167  000042                  JMP     2$                          ;REJOIN CODE
13932                                 ;
13933                                 ;INSTRUCTION TRAPPED
13934                                 ;
13935 070320  032737  000001  003030  50$:     BIT     #1,@#FLAG                   ;SEE IF EXPECTING A TRAP
13936 070326  001005                          BNE     51$                         ;BRANCH IF EXPECTING A TRAP
13937 070330  004737  140132                  CALL    @#DETFPA                    ;DETERMINE FLOATING POINT FAULT. $$$
13938 070334  104003                          ERROR   +3                  ;FPP ERROR
13939                                                                     ;INSTRUCTION WASNT SUPPOSE TO TRAP
13940 070336  000167  000020                  JMP     2$                          ;REJOIN CODE
13941 070342  012604                  51$:     MOV     (SP)+,R4                    ;SEE IF PC = INSTRUCTION
13942 070344  005726                          TST     (SP)+                       ;CLEAN UP STACK
13943 070346  022704  070274                  CMP     #1$,R4                      ;
13944 070352  001403                          BEQ     2$                          ;BRANCH IF GOOD COMPARE
13945 070354  004737  140132                  CALL    @#DETFPA                    ;DETERMINE FLOATING POINT FAULT. $$$
13946 070360  104003                          ERROR   +3                  ;FPP ERROR
13947                                                                     ;PC WAS INCORRECT
```

FLOATING POINT TESTS

```
13948                                   ;
13949                                   ;COMMON CODE FOR TRAP AND NO TRAP
13950                                   ;
13951 070362  170203                    2$:     STFPS   R3                      ;SAVE FPS
13952 070364  012702  000200                    MOV     #200,R2                 ;SET FPP TO DOUBLE
13953 070370  170102                            LDFPS   R2
13954 070372  012701  003142                    MOV     #RECDST,R1              ;POINT TO RECEIVED DATA TABLE
13955 070376  174011                            STD     AC0,(R1)                ;SAVE AC0  RESULT
13956 070400  026503  000032                    CMP     32(R5),R3               ;VERIFY STATUS
13957 070404  001403                            BEQ     3$                      ;BRANCH IF GOOD
13958 070406  004737  140132                    CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
13959 070412  104003                            ERROR   +3              ;FPP ERROR
13960                                                                   ;BAD FPS
13961 070414  010504                    3$:     MOV     R5,R4                   ;POINT TO EXPECTED DATA
13962 070416  062704  000020                    ADD     #20,R4
13963 070422  004767  047452            4$:     JSR     R7,DATVER                       ;VERIFY DATA
13964 070426  005767  112466                    TST     COUNT
13965 070432  001403                            BEQ     5$                      ;BRANCH IF GOOD
13966 070434  004737  140132                    CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
13967 070440  104003                            ERROR   +3              ;FPP ERROR
13968                                                                   ;BAD AC0
13969 070442  005737  003030            5$:     TST     @#FLAG                  ;SEE IF NEED TO CHECK FEC
13970 070446  001002                            BNE     6$                      ;BRANCH IF NEED TO CHECK
13971 070450  000165  000034                    JMP     34(R5)                  ;RETURN FROM TEST
13972 070454  170301                    6$:     STST    R1                      ;SAVE FEC
13973 070456  016504  000034                    MOV     34(R5),R4               ;GET FEC
13974 070462  020401                            CMP     R4,R1                   ;VERIFY FEC
13975 070464  001403                            BEQ     7$                      ;BRANCH IF GOOD
13976 070466  004737  140132                    CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
13977 070472  104003                            ERROR   +3              ;FPP ERROR
13978                                                                           ;BAD FEC
13979 070474  000165  000036            7$:     JMP     36(R5)                  ;RETURN FROM TEST
13980                                   ;
13981 070500                            HOP11:
13984                                   ;
13985                                   ;--------------- ----------------
13986                                   ;TEST MULF
13987                                   ;
13988 070500                            MMULF:
13989                                   ;
13990                                   ;1/AC0=FSRC=0 -INTERRUPTS DISABLED
13991 070500  005037  003030                    CLR     @#FLAG                  ;NO INTERRUPT
13992 070504  004767  000564                    JSR     R7,MLFSUB                       ;DO TEST
13993 070510  000000  000000                    .WORD   0,0     ;AC0
13994 070514  000000  000000                    .WORD   0,0     ;FSRC
13995 070520  000000  000000                    .WORD   0,0                     ;RESULT
13996 070524  007517                            .WORD   7517                    ; TEST FPS
13997 070526  007504                            .WORD   7504                    ;RESULTANT FPS
13998                                   ;2/AC>FSRC=0 -  INTERRUPTS ON
13999 070530  005037  003030                    CLR     @#FLAG                  ;NO INTERRUPT
14000 070534  004767  000534                    JSR     R7,MLFSUB                       ;DO TEST
14001 070540  000200  000000                    .WORD   200,0   ;AC0
14002 070544  000000  000000                    .WORD   0,0     ;FSRC
14003 070550  000000  000000                    .WORD   0,0                     ;RESULT
14004 070554  000013                            .WORD   13                      ; TEST FPS
14005 070556  000004                            .WORD   4                       ;RESULTANT FPS
14006                                   ;3/AC=0 FSRC>0
```

FLOATING POINT TESTS

```
14007 070560  005037  003030                CLR     @#FLAG          ;NO INTERRUPT
14008 070564  004767  000504                JSR     R7,MLFSUB               ;DO TEST
14009 070570  000100  000000                .WORD   100,0   ;AC0
14010 070574  000300  000000                .WORD   300,0   ;FSRC
14011 070600  000000  000000                .WORD   0,0             ;RESULT
14012 070604  007500                        .WORD   7500            ; TEST FPS
14013 070606  007504                        .WORD   7504            ;RESULTANT FPS
14014                           ;4/AC=1 >FSRC   ROUND
14015 070610  005037  003030                CLR     @#FLAG          ;NO INTERRUPT
14016 070614  004767  000454                JSR     R7,MLFSUB               ;DO TEST
14017 070620  040200  000000                .WORD   40200,0 ;AC0
14018 070624  040177  177777                .WORD   40177,-1        ;FSRC
14019 070630  040177  177777                .WORD   40177,-1                ;RESULT
14020 070634  000000                        .WORD   0               ; TEST FPS
14021 070636  000000                        .WORD   0               ;RESULTANT FPS
14022                           ;5/TRUNCATE
14023 070640  005037  003030                CLR     @#FLAG          ;NO INTERRUPT
14024 070644  004767  000424                JSR     R7,MLFSUB               ;DO TEST
14025 070650  040177  177777                .WORD   40177,-1        ;AC0
14026 070654  040200  000000                .WORD   40200,0 ;FSRC
14027 070660  040177  177777                .WORD   40177,-1                ;RESULT
14028 070664  000040                        .WORD   40              ; TEST FPS
14029 070666  000040                        .WORD   40              ;RESULTANT FPS
14030                           ;6/NORMALIZE
14031 070670  005037  003030                CLR     @#FLAG          ;NO INTERRUPT
14032 070674  004767  000374                JSR     R7,MLFSUB               ;DO TEST
14033 070700  040100  000000                .WORD   40100,0 ;AC0
14034 070704  040100  000000                .WORD   40100,0 ;FSRC
14035 070710  040020  000000                .WORD   40020,0         ;RESULT
14036 070714  000012                        .WORD   12              ; TEST FPS
14037 070716  000000                        .WORD   0               ;RESULTANT FPS
14038                           ;7/ROUND
14039 070720  005037  003030                CLR     @#FLAG          ;NO INTERRUPT
14040 070724  004767  000344                JSR     R7,MLFSUB               ;DO TEST
14041 070730  017500  000000                .WORD   17500,0 ;AC0
14042 070734  023652  125252                .WORD   23652,125252    ;FSRC
14043 070740  003177  177777                .WORD   3177,-1         ;RESULT
14044 070744  007417                        .WORD   7417            ; TEST FPS
14045 070746  007400                        .WORD   7400            ;RESULTANT FPS
14046                           ;8/AC>0>FSRC ROUND
14047 070750  005037  003030                CLR     @#FLAG          ;NO INTERRUPT
14048 070754  004767  000314                JSR     R7,MLFSUB               ;DO TEST
14049 070760  040342  177777                .WORD   40342,-1        ;AC0
14050 070764  176543  025252                .WORD   176543,025252   ;FSRC
14051 070770  176711  067324                .WORD   176711,67324            ;RESULT
14052 070774  007500                        .WORD   7500            ; TEST FPS
14053 070776  007510                        .WORD   7510            ;RESULTANT FPS
14054                           ;9/IAC<FSRC<0, ROUND
14055 071000  005037  003030                CLR     @#FLAG          ;NO INTERRUPT
14056 071004  004767  000264                JSR     R7,MLFSUB               ;DO TEST
14057 071010  144600  000000                .WORD   144600,0                ;AC0
14058 071014  154000  000000                .WORD   154000,0                ;FSRC
14059 071020  060400  000000                .WORD   60400,0         ;RESULT
14060 071024  000017                        .WORD   17              ; TEST FPS
14061 071026  000000                        .WORD   0               ;RESULT FPS
14062                           ;10/AC<FSRC, ROUND
14063 071030  005037  003030                CLR     @#FLAG          ;NO INTERRUPT
```

```
071034  004767  000234                  JSR     R7,MLFSUB               ;DO TEST
071040  060000  000000                  .WORD   60000,0         ;ACO
071044  140377  177776                  .WORD   140377,177776           ;FSRC
071050  160177  177776                  .WORD   160177,177776           ;RESULT
071054  000017                          .WORD   17              ; TEST FPS
071056  000010                          .WORD   10              ;RESULT FPS
                                ;11/AC>0>FSRC, TRUNCATE
071060  005037  003030                  CLR     @#FLAG          ;NO INTERRUPT
071064  004767  000204                  JSR     R7,MLFSUB               ;DO TEST
071070  060000  000000                  .WORD   60000,0         ;ACO
071074  140377  177776                  .WORD   140377,177776           ;FSRC
071100  160177  177776                  .WORD   160177,177776           ;RESULT
071104  007547                          .WORD   7547            ; TEST FPS
071106  007550                          .WORD   7550            ;RESULT FPS
                                ;12/UNDERFLOW, NO INTERRUPTS
071110  012737  000002  003030          MOV     #2,@#FLAG               ;NO INTERRUPT
071116  004767  000152                  JSR     R7,MLFSUB               ;DO TEST
071122  000200  000001                  .WORD   200,1           ;ACO
071126  000200  000001                  .WORD   200,1           ;FSRC
071132  040200  000002                  .WORD   40200,2         ;RESULT
071136  042117                          .WORD   42117           ; TEST FPS
071140  142100                          .WORD   142100          ;RESULT FPS
071142  000012                          .WORD   12              ;FEC
                                ;13/OVERFLOW, TRAP
071144  012737  000001  003030          MOV     #1,@#FLAG               ;INTERRUPT
071152  004767  000116                  JSR     R7,MLFSUB               ;DO TEST
071156  177777  177777                  .WORD   177777, 1               ;ACO
071162  040300  000000                  .WORD   40300,0         ;FSRC
071166  100077  177777                  .WORD   100077,-1               ;RESULT
071172  001117                          .WORD   1117            ; TEST FPS
071174  101116                          .WORD   101116          ;RESULT FPS
071176  000010                          .WORD   10              ;FEC
                                ;14/OVERFLOW NO TRAP
071200  012737  000002  003030          MOV     #2,@#FLAG               ;NO INTERRUPT
071206  004767  000062                  JSR     R7,MLFSUB               ;DO TEST
071212  077700  000000                  .WORD   77700,0         ;ACO
071216  077700  000000                  .WORD   77700,0         ;FSRC
071222  000000  000000                  .WORD   0,0             ;RESULT
071226  040117                          .WORD   40117           ; TEST FPS
071230  040106                          .WORD   40106           ;RESULT FPS
071232  000010                          .WORD   10              ;FEC
                                ;15/UNDEFINED VARIABLE IN FSRC, TRAP ENABLED
071234  012737  000001  003030          MOV     #1,@#FLAG               ;INTERRUPT
071242  004767  000026                  JSR     R7,MLFSUB               ;DO TEST
071246  123465  000000                  .WORD   123465,0                ;ACO
071252  100022  000000                  .WORD   100022,0                ;FSRC
071256  123465  000000                  .WORD   123465,0                ;RESULT
071262  004000                          .WORD   4000            ; TEST FPS
071264  104000                          .WORD   104000          ;RESULT FPS
071266  000014                          .WORD   14              ;FEC
                                ;
071270  000167  000242                  JMP     HOP12
                                ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
                                ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
                                ;
                                ;               ACO
                                ;               FSRC
```

FLOATING POINT TESTS

```
14121                                   ;               FPS BEFORE EXECUTION
14122                                   ;               FPS AFTER EXECUTION
14123                                   ;               (FEC)
14124                                   ;
14125                                   ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
14126                                   ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
14127                                   ;
14128 071274  012605                  MLFSUB: MOV     (SP)+,R5                ; RETURN ADDRESS TO USE AS POINTER
14129 071276  012737  071356  000244          MOV     #50$,@#FPVEC            ;REDIRECT TRAP VECTOR
14130 071304  012702  000200                  MOV     #200,R2                 ;SET TO DOUBLE MODE FOR LOAD
14131 071310  170102                          LDFPS   R2                      ;LOAD FPS
14132 071312  172415                          LDD     (R5),AC0                ;LOAD AC0 WITH TEST DATA
14133 071314  010504                          MOV     R5,R4                   ;POINT TO FSRC DATA
14134 071316  062704  000004                  ADD     #4,R4
14135 071322  016502  000014                  MOV     14(R5),R2               ;GET TEST FPS
14136 071326  170102                          LDFPS   R2                      ;LOAD TEST FPS
14137                                   ;
14138 071330  171014                          MULF    (R4),AC0                ;*TEST INSTRUCTION
14139 071332  170001                  1$:     SETF                            ;WAIT FOR POSSIBLE FPA TRAP.
14140                                   ;
14141                                   ;INSTRUCTION DIDNT TRAP
14142                                   ;
14143 071334  032737  000001  003030          BIT     #1,@#FLAG                       ;VERIFY A NO TRAP CONDITION
14144 071342  001426                          BEQ     2$                      ;BRANCH IF GOOD
14145 071344  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
14146 071350  104003                          ERROR   +3                      ;FPP ERROR
14147                                                                         ;INSTRUCTION SHOULD HAVE TRAPPED
14148 071352  000167  000042                  JMP     2$                      ;REJOIN CODE
14149                                   ;
14150                                   ;INSTRUCTION TRAPPED
14151                                   ;
14152 071356  032737  000001  003030  50$:    BIT     #1,@#FLAG               ;SEE IF EXPECTING A TRAP
14153 071364  001005                          BNE     51$                     ;BRANCH IF EXPECTING A TRAP
14154 071366  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
14155 071372  104003                          ERROR   +3                      ;FPP ERROR
14156                                                                         ;INSTRUCTION WASNT SUPPOSE TO TRAP
14157 071374  000167  000020                  JMP     2$                      ;REJOIN CODE
14158 071400  012604                  51$:    MOV     (SP)+,R4                ;SEE IF PC = INSTRUCTION
14159 071402  005726                          TST     (SP)+                   ;CLEAN UP STACK
14160 071404  022704  071332                  CMP     #1$,R4                  ;
14161 071410  001403                          BEQ     2$                      ;BRANCH IF GOOD COMPARE
14162 071412  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
14163 071416  104003                          ERROR   +3                      ;FPP ERROR
14164                                                                         ;PC WAS INCORRECT
14165                                   ;
14166                                   ;COMMON CODE FOR TRAP AND NO TRAP
14167                                   ;
14168 071420  170203                  2$:     STFPS   R3                      ;SAVE FPS
14169 071422  012702  000200                  MOV     #200,R2                 ;SET FPP TO DOUBLE
14170 071426  170102                          LDFPS   R2
14171 071430  012701  003142                  MOV     #RECDST,R1              ;POINT TO RECEIVED DATA TABLE
14172 071434  174011                          STD     AC0,(R1)                ;SAVE AC0  RESULT
14173 071436  026503  000016                  CMP     16(R5),R3               ;VERIFY STATUS
14174 071442  001403                          BEQ     3$                      ;BRANCH IF GOOD
14175 071444  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
14176 071450  104003                          ERROR   +3                      ;FPP ERROR
14177                                                                         ;BAD FPS
```

FLOATING POINT TESTS

```
14178 071452  010504                3$:     MOV     R5,R4                   ;POINT TO EXPECTED DATA
14179 071454  062704  000010                ADD     #10,R4
14180 071460  004767  046376        4$:     JSR     R7,DATVFR                       ;VERIFY DATA
14181 071464  005767  111430                TST     COUNT
14182 071470  001403                        BEQ     5$                      ;BRANCH IF GOOD
14183 071472  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
14184 071476  104003                        ERROR   .+3             ;FPP ERROR
14185                                                               ;BAD ACO
14186 071500  005737  003030        5$:     TST     @#FLAG                  ;SEE IF NEED TO CHECK FEC
14187 071504  001002                        BNE     6$                      ;BRANCH IF NEED TO CHECK
14188 071506  000165  000020                JMP     20(R5)                  ;RETURN FROM TEST
14189                                       ;
14190                                       ;VERIFY ERROR STATUS
14191                                       ;
14192 071512  170301                6$:     STST    R1                      ;SAVE FEC
14193 071514  016504  000020                MOV     20(R5),R4               ;GET FEC
14194 071520  020401                        CMP     R4,R1                   ;VERIFY FEC
14195 071522  001403                        BEQ     7$                      ;BRANCH IF GOOD
14196 071524  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
14197 071530  104003                        ERROR   .+3             ;FPP ERROR
14198                                                               ;BAD FEC
14199 071532  000165  000022        7$:     JMP     22(R5)          ;RETURN FROM TEST
14200 071536                        HOP12:
14203                                       ;
14204                                       ;-------------------- -------  --
14205                                       ;TEST MULD
14206                                       ;
14207 071536                        MMULD:
14208                                       ;
14209                                       ;1/AC=0
14210 071536  005037  003030                CLR     @#FLAG                  ;NO INTERRUPT
14211 071542  004767  000554                JSR     R7,MLDSUB                       ;DO TEST
14212 071546  000100  000000  000000        .WORD   100,0,0,0               ;ACO
      071554  000000
14213 071556  000411  177777  000000        .WORD   411,-1,0,1              ;FSRC
      071564  000001
14214 071566  000000  000000  000000        .WORD   0,0,0,0                 ;RESULT
      071574  000000
14215 071576  000200                        .WORD   200                     ; TEST FPS
14216 071600  000204                        .WORD   204                     ;RESULTANT FPS
14217                                       ;2/FSRC=0
14218 071602  005037  003030                CLR     @#FLAG                  ;NO INTERRUPT
14219 071606  004767  000510                JSR     R7,MLDSUB                       ;DO TEST
14220 071612  077777  000000  000000        .WORD   77777,0,0,0             ;ACO
      071620  000000
14221 071622  000000  000000  000000        .WORD   0,0,0,0 ;FSRC
      071630  000000
14222 071632  000000  000000  000000        .WORD   0,0,0,0                 ;RESULT
      071640  000000
14223 071642  007700                        .WORD   7700                    ; TEST FPS
14224 071644  007704                        .WORD   7704                    ;RESULTANT FPS
 4225                                       ;3/AC=1
14226 071646  005037  003030                CLR     @#FLAG                  ;NO INTERRUPT
14227 071652  004767  000444                JSR     R7,MLDSUB                       ;DO TEST
14228 071656  040200  000000  000000        .WORD   40200,0,0,0             ;ACO
      071664  000000
14229 071666  000277  177777  177777        .WORD   277,-1,-1,-1            ;FSRC
```

FLOATING POINT TESTS

```
        071674  177777
14230   071676  000277  177777  177777          .WORD   277,-1,-1, 1             ;RESULT
        071704  177777
14231   071706  007717                          .WORD   7717              ; TEST FPS
14232   071710  007700                          .WORD   7700              ;RESULTANT FPS
14233                           ;4/AC>FSRC>0, TRUNCATE
14234   071712  005037  003030                  CLR     @#FLAG            ;NO INTERRUPT
14235   071716  004767  000400                  JSR     R7,MLDSUB                 ;DO TEST
14236   071722  065500  000000  000000          .WORD   65500,0,0,1       ;AC0
        071730  000001
14237   071732  037577  177777  177777          .WORD   37577, 1, 1, 2    ;FSRC
        071740  177776
14238   071742  065077  177777  177777          .WORD   65077,-1, 1,-1            ;RESULT
        071750  177777
14239   071752  007717                          .WORD   7717              ; TEST FPS
14240   071754  007700                          .WORD   7700              ;RESULTANT FPS
14241                           ;5/AC<FSRC<0
14242   071756  005037  003030                  CLR     @#FLAG            ;NO INTERRUPT
14243   071762  004767  000334                  JSR     R7,MLDSUB                 ;DO TEST
14244   071766  137577  177777  177777          .WORD   137577,-1,-1,-2 ;AC0
        071774  177776
14245   071776  165400  000000  000000          .WORD   165400,0,0,1      ;FSRC
        072004  000001
14246   072006  065000  000000  000000          .WORD   65000,0,0,0               ;RESULT
        072014  000000
14247   072016  007717                          .WORD   7717              ; TEST FPS
14248   072020  007700                          .WORD   7700              ;RESULTANT FPS
14249                           ;6/AC>FSRC>0
14250   072022  005037  003030                  CLR     @#FLAG            ;NO INTERRUPT
14251   072026  004767  000270                  JSR     R7,MLDSUB                 ;DO TEST
14252   072032  017500  000000  000000          .WORD   17500,0,0,0       ;AC0
        072040  000000
14253   072042  123652  125252  125252          .WORD   123652,125252,125252,125252     ;FSRC
        072050  125252
14254   072052  103177  177777  177777          .WORD   103177,-1,-1,-1           ;RESULT
        072060  177777
14255   072062  000200                          .WORD   200               ; TEST FPS
14256   072064  000210                          .WORD   210               ;RESULTANT FPS
14257                           ;7/UNDERFLOW, TRAPS DISABLED
14258   072066  005037  003030                  CLR     @#FLAG                    ;NO INTERRUPT
14259   072072  004767  000224                  JSR     R7,MLDSUB                 ;DO TEST
14260   072076  000300  000000  000000          .WORD   300,0,0,252               ;AC0
        072104  000252
14261   072106  000377  000001  000002          .WORD   377,1,2,3                 ;FSRC
        072114  000003
14262   072116  000000  000000  000000          .WORD   0,0,0,0           ;RESULT
        072124  000000
14263   072126  005740                          .WORD   5740              ; TEST FPS
14264   072130  005744                          .WORD   5744              ;RESULT FPS
14265                           ;8/UNDERFLOW, TRAP ENABLED
14266   072132  012737  000001  003030          MOV     #1,@#FLAG                 ;INTERRUPT
14267   072140  004767  000156                  JSR     R7,MLDSUB                 ;DO TEST
14268   072144  100277  000001  000002          .WORD   100277,1,2,-1             ;AC0
        072152  177777
14269   072154  100300  000001  000001          .WORD   100300,1,1,1              ;FSRC
        072162  000001
14270   072164  040417  040001  077403          .WORD   40417,40001,77403,0               ;RESULT
```

FLOATING POINT TESTS

```
        072172  000000
14271   072174  002217                  .WORD   2217            ; TEST FPS
14272   072176  102200                  .WORD   102200          ;RESULT FPS
14273   072200  000012                  .WORD   12              ;FEC
14274                           ;9/OVERFLOW, TRAPS DISABLED
14275   072202  005037  003030          CLR     @#FLAG                  ;NO INTERRUPT
14276   072206  004767  000110          JSR     R7,MLDSUB               ;DO TEST
14277   072212  177777  177777  177777  .WORD   -1,-1,-1,-1             ;AC0
        072220  177777
14278   072222  040200  177777  177777  .WORD   40200, 1, 1, 1          ;FSRC
        072230  177777
14279   072232  000000  000000  000000  .WORD   0,0,0,0          ;RESULT
        072240  000000
14280   072242  006740                  .WORD   6740            ; TEST FPS
14281   072244  006746                  .WORD   6746            ;RESULT FPS
14282                           ;10/OVERFLOW, TRAPS ENABLED
14283   072246  012737  000001  003030  MOV     #1,@#FLAG               ;INTERRUPT
14284   072254  004767  000042          JSR     R7,MLDSUB               ;DO TEST
14285   072260  157700  025252  025252  .WORD   157700,25252,25252,25252                 ;AC0
        072266  025252
14286   072270  167700  000000  000000  .WORD   167700,0,0,0            ;FSRC
        072276  000000
14287   072300  007420  017777  117777  .WORD   7420,017777,117777,117777               ;RESULT
        072306  117777
14288   072310  001240                  .WORD   1240            ; TEST FPS
14289   072312  101242                  .WORD   101242          ;RESULT FPS
14290   072314  000010                  .WORD   10              ;FEC
14291                           ;
14292   072316  000167  000242          JMP     HOP13
14293                           ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
14294                           ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
14295                           ;
14296                           ;               AC0
14297                           ;               FSRC
14298                           ;               FPS BEFORE EXECUTION
14299                           ;               FPS AFTER EXECUTION
14300                           ;               (FEC)
14301                           ;
14302                           ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
14303                           ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
14304                           ;
14305   072322  012605          MLDSUB: MOV     (SP)+,R5                ; RETURN ADDRESS TO USE AS POINTER
14306   072324  012737  072404  000244  MOV     #50$,@#FPVEC            ;REDIRECT TRAP VECTOR
14307   072332  012702  000200          MOV     #200,R2                 ;SET TO DOUBLE MODE FOR LOAD
14308   072336  170102                  LDFPS   R2                      ;LOAD FPS
14309   072340  172415                  LDD     (R5),AC0                ;LOAD AC0 WITH TEST DATA
14310   072342  010501                  MOV     R5,R1                   ;POINT TO FSRC DATA
14311   072344  062701  000010          ADD     #10,R1
14312   072350  016502  000030          MOV     30(R5),R2               ;GET TEST FPS
14313   072354  170102                  LDFPS   R2                      ;LOAD TEST FPS
14314                           ;
14315   072356  171011                  MULD    (R1),AC0                ;*TEST INSTRUCTION
14316   072360  170011          1$:     SETD                            ;WAIT FOR POSSIBLE FPA TRAP.
14317                           ;
14318                           ;INSTRUCTION DIDNT TRAP
14319                           ;
14320   072362  032737  000001  003030          BIT     #1,@#FLAG                       ;VERIFY A NO TRAP CONDITION
```

FLOATING POINT TESTS

```
14321 072370  001426                    BEQ     2$                      ;BRANCH IF GOOD
14322 072372  004737  140132              CALL    @#DETFPA                ;DETER'INE FLOATING POINT FAULT. $$$
14323 072376  104003                    ERROR   ;3              ;FPP ERROR
14324                                           ;INSTRUCTION SHOULD HAVE TRAPPED
14325 072400  000167  000042              JMP     2$                      ;REJOIN CODE
14326                                 ;
14327                                 ;INSTRUCTION TRAPPED
14328                                 ;
14329 072404  032737  000001  003030  50$:   BIT     #1,@#FLAG               ;SEE IF EXPECTING A TRAP
14330 072412  001005                    BNE     51$                     ;BRANCH IF EXPECTING A TRAP
14331 072414  004737  140132              CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
14332 072420  104003                    ERROR   ;3              ;FPP ERROR
14333                                           ;INSTRUCTION WASNT SUPPOSE TO TRAP
14334 072422  000167  000020              JMP     2$                      ;REJOIN CODE
14335 072426  012604                51$:   MOV     (SP)+,R4                ;SEE IF PC = INSTRUCTION
14336 072430  005726                    TST     (SP)+                   ;CLEAN UP STACK
14337 072432  022704  072360              CMP     #1$,R4                  ;
14338 072436  001403                    BEQ     2$                      ;BRANCH IF GOOD COMPARE
14339 072440  004737  140132              CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
14340 072444  104003                    ERROR   ;3              ;FPP ERROR
14341                                           ;PC WAS INCORRECT
14342                                 ;
14343                                 ;COMMON CODE FOR TRAP AND NO TRAP
14344                                 ;
14345 072446  170203                2$:    STFPS   R3                      ;SAVE FPS
14346 072450  012702  000200              MOV     #200,R2                 ;SET FPP TO DOUBLE
14347 072454  170102                    LDFPS   R2
14348 072456  012701  003142              MOV     #RECDST,R1              ;POINT TO RECEIVED DATA TABLE
14349 072462  174011                    STD     AC0,(R1)                ;SAVE AC0  RESULT
14350 072464  026503  000032              CMP     32(R5),R3               ;VERIFY STATUS
14351 072470  001403                    BEQ     3$                      ;BRANCH IF GOOD
14352 072472  004737  140132              CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
14353 072476  104003                    ERROR   ;3              ;FPP ERROR
14354                                           ;BAD FPS
14355 072500  010504                3$:    MOV     R5,R4                   ;POINT TO EXPECTED DATA
14356 072502  062704  000020              ADD     #20,R4
14357 072506  004767  045366        4$:    JSR     R7,DATVER                       ;VERIFY DATA
14358 072512  005767  110402              TST     COUNT
14359 072516  001403                    BEQ     5$                      ;BRANCH IF GOOD
14360 072520  004737  140132              CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
14361 072524  104003                    ERROR   ;3              ;FPP ERROR
14362                                           ;BAD AC0
14363 072526  005737  003030        5$:    TST     @#FLAG                  ;SEE IF NEED TO CHECK FEC
14364 072532  001002                    BNE     6$                      ;BRANCH IF NEED TO CHECK
14365 072534  000165  000034              JMP     34(R5)                  ;RETURN FROM TEST
14366                                 ;VERIFY ERROR STATUS
14367 072540  170301                6$:    STST    R1                      ;SAVE FEC
14368 072542  016504  000034              MOV     34(R5),R4               ;GET FEC
14369 072546  020401                    CMP     R4,R1                   ;VERIFY FEC
14370 072550  001403                    BEQ     7$                      ;BRANCH IF GOOD
14371 072552  004737  140132              CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
14372 072556  104003                    ERROR   ;3              ;FPP ERROR
14373                                           ;BAD FEC
14374 072560  000165  000036        7$:    JMP     36(R5)                  ;RETURN FROM TEST
14375 072564                        HOP13:
14378                                 ;
14379                                 ;---    -       -
```

FLOATING POINT TESTS

```
14380                                   ;TEST MODF
14381                                   ;
14382 072564                            MMODF:
14383                                   ;
14384                                   ;1/AC=0 FSRC=0
14385 072564  005037  003030                      CLR     @#FLAG          ;NO INTERRUPT
14386 072570  004767  000554                      JSR     R7,MDFSUB               ;DO TEST
14387 072574  000100  000000                      .WORD   100,0   ;AC0
14388 072600  012346  177777                      .WORD   12346,-1        ;FSRC
14389 072604  000000  000000                      .WORD   0,0             ;FRACTIONAL RESULT
14390 072610  000000  000000                      .WORD   0,0             ;INTEGER RESULT
14391 072614  000013                              .WORD   13              ; TEST FPS
14392 072616  000004                              .WORD   4               ;RESULTANT FPS
14393                                   ;2/FSRC=0
14394 072620  005037  003030                      CLR     @#FLAG          ;NO INTERRUPT
14395 072624  004767  000520                      JSR     R7,MDFSUB               ;DO TEST
14396 072630  012356  177777                      .WORD   12356,-1        ;AC0
14397 072634  000000  000000                      .WORD   0,0     ;FSRC
14398 072640  000000  000000                      .WORD   0,0             ;FRACTIONAL RESULT
14399 072644  000000  000000                      .WORD   0,0             ;INTEGER RESULT
14400 072650  000003                              .WORD   3               ; TEST FPS
14401 072652  000004                              .WORD   4       ;RESULTANT FPS
14402                                   ;3/AC=0
14403 072654  005037  003030                      CLR     @#FLAG          ;NO INTERRUPT
14404 072660  004767  000464                      JSR     R7,MDFSUB               ;DO TEST
14405 072664  000000  000000                      .WORD   0,0             ;AC0
14406 072670  177777  177777                      .WORD   -1,-1           ;FSRC
14407 072674  000000  000000                      .WORD   0,0             ;FRACTIONAL  RESULT
14408 072700  000000  000000                      .WORD   0,0             ;INTEGER RESULT
14409 072704  007500                              .WORD   7500            ; TEST FPS
14410 072706  007504                              .WORD   7504            ;RESULT FPS
14411                                   ;4/AC>FSRC>0
14412 072710  005037  003030                      CLR     @#FLAG          ;NO INTERRUPT
14413 072714  004767  000430                      JSR     R7,MDFSUB               ;DO TEST
14414 072720  046252  125252                      .WORD   46252,125252    ;AC0
14415 072724  040300  000000                      .WORD   40300,0 ;FSRC
14416 072730  000000  000000                      .WORD   0,0             ;FRACTIONAL RESULT
14417 072734  046377  177777                      .WORD   46377,-1                ;INTEGER RESULT
14418 072740  000013                              .WORD   13              ; TEST FPS
14419 072742  000004                              .WORD   4               ;RESULTANT FPS
14420                                   ;5/AC>FSRC>0
14421 072744  005037  003030                      CLR     @#FLAG          ;NO INTERRUPT
14422 072750  004767  000374                      JSR     R7,MDFSUB               ;DO TEST
14423 072754  077652  125252                      .WORD   77652,125252    ;AC0
14424 072760  040300  000000                      .WORD   40300,0 ;FSRC
14425 072764  000000  000000                      .WORD   0,0             ;FRACTIONAL RESULT
14426 072770  077777  177777                      .WORD   77777,-1                ;INTEGER RESULT
14427 072774  000000                              .WORD   0               ; TEST FPS
14428 072776  000004                              .WORD   4               ;RESULTANT FPS
14429                                   ;6/AC>0<FSRC, INTEGERS
14430 073000  005037  003030                      CLR     @#FLAG          ;NO INTERRUPT
14431 073004  004767  000340                      JSR     R7,MDFSUB               ;DO TEST
14432 073010  060600  000000                      .WORD   60600,0         ;AC0
14433 073014  147400  025700                      .WORD   147400,25700            ;FSRC
14434 073020  000000  000000                      .WORD   0,0             ;FRACTIONAL  RESULT
14435 073024  170000  025700                      .WORD   170000,25700            ;INTEGER RESULT
14436 073030  007400                              .WORD   7400            ; TEST FPS
```

FLOATING POINT TESTS

```
14437 073032  007404                        .WORD   7404            ;RESULT FPS
14438                               ;7/AC<0<FSRC, FRACTIONAL
14439 073034  005037  003030                CLR     @#FLAG          ;NO INTERRUPT
14440 073040  004767  000304                JSR     R7,MDFSUB               ;DO TEST
14441 073044  100227  177777                .WORD   100227, 1               ;AC0
14442 073050  044025  025252                .WORD   44025,25252             ;FSRC
14443 073054  104061  021251                .WORD   104061,21251            ;FRACTIONAL  RESULT
14444 073060  000000  000000                .WORD   0,0             ;INTEGER RESULT
14445 073064  000000                        .WORD   0               ; TEST FPS
14446 073066  000010                        .WORD   10              ;RESULT FPS
14447                               ;8/AC<0>FSRC, TRUNCATE
14448 073070  005037  003030                CLR     @#FLAG          ;NO INTERRUPT
14449 073074  004767  000250                JSR     R7,MDFSUB               ;DO TEST
14450 073100  046252  125252                .WORD   46252,125252            ;AC0
14451 073104  040300  000000                .WORD   40300,0         ;FSRC
14452 073110  000000  000000                .WORD   0,0             ;FRACTIONAL  RESULT
14453 073114  046377  177777                .WORD   46377,-1                ;INTEGER RESULT
14454 073120  000053                        .WORD   53              ; TEST FPS
14455 073122  000044                        .WORD   44              ;RESULT FPS
14456                               ;9/ROUND INTEGER
14457 073124  005037  003030                CLR     @#FLAG          ;NO INTERRUPT
14458 073130  004767  000214                JSR     R7,MDFSUB               ;DO TEST
14459 073134  046252  125252                .WORD   46252,125252            ;AC0
14460 073140  040300  000000                .WORD   40300,0         ;FSRC
14461 073144  000000  000000                .WORD   0,0             ;FRACTIONAL  RESULT
14462 073150  046377  177777                .WORD   46377,-1                ;INTEGER RESULT
14463 073154  000013                        .WORD   13              ; TEST FPS
14464 073156  000004                        .WORD   4               ;RESULT FPS
14465                               ;10/TRUNCATE FRACTION
14466 073160  005037  003030                CLR     @#FLAG          ;NO INTERRUPT
14467 073164  004767  000160                JSR     R7,MDFSUB               ;DO TEST
14468 073170  040777  177777                .WORD   40777,-1                ;AC0
14469 073174  040200  000000                .WORD   40200,0         ;FSRC
14470 073200  040177  177770                .WORD   40177,177770            ;FRACTIONAL  RESULT
14471 073204  040740  000000                .WORD   40740,0         ;INTEGER RESULT
14472 073210  000000                        .WORD   0               ; TEST FPS
14473 073212  000000                        .WORD   0               ;RESULT FPS
14474                               ;11/ROUND INTEGER
14475 073214  005037  003030                CLR     @#FLAG          ;NO INTERRUPT
14476 073220  004767  000124                JSR     R7,MDFSUB               ;DO TEST
14477 073224  000000  000000                .WORD   0,0             ;AC0
14478 073230  000000  000000                .WORD   0,0             ;FSRC
14479 073234  000000  000000                .WORD   0,0             ;FRACTIONAL  RESULT
14480 073240  000000  000000                .WORD   0,0             ;INTEGER RESULT
14481 073244  000000                        .WORD   0               ; TEST FPS
14482 073246  000004                        .WORD   4               ;RESULT FPS
14483                               ;12/ROUND FRACTION
14484 073250  005037  003030                CLR     @#FLAG          ;NO INTERRUPT
14485 073254  004767  000070                JSR     R7,MDFSUB               ;DO TEST
14486 073260  040225  125252                .WORD   40225,125252            ;AC0
14487 073264  066652  052525                .WORD   66652,52525             ;FSRC
14488 073270  000000  000000                .WORD   0,0             ;FRACTIONAL  RESULT
14489 073274  066707  025160                .WORD   66707,25160             ;INTEGER RESULT
14490 073300  007027                        .WORD   7027            ; TEST FPS
14491 073302  007004                        .WORD   7004            ;RESULT FPS
14492                               ;/OVERFLOW
14493 073304  012737  000001  003030        MOV     #1,@#FLAG               ;INTERRUPT
```

FLOATING POINT TESTS

```
14494 073312  004767  000032                    JSR     R7,MDFSUB               ;DO TEST
14495 073316  076000  000000                    .WORD   76000,0         ;AC0
14496 073322  076000  000000                    .WORD   76000,0         ;FSRC
14497 073326  000000  000000                    .WORD   0,0             ;FRACTIONAL  RESULT
14498 073332  033600  000000                    .WORD   33600,0         ;INTEGER RESULT
14499 073336  001000                            .WORD   1000            ; TEST FPS
14500 073340  101006                            .WORD   101006          ;RESULT FPS
14501 073342  000010                            .WORD   10              ;FEC
14502                                 ;
14503 073344  000167  000310                    JMP     HOP14
14504                                 ;
14505                                 ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
14506                                 ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
14507                                 ;
14508                                 ;               AC0
14509                                 ;               FSRC
14510                                 ;               FRACTIONAL RESULT
14511                                 ;               INTEGER RESULT
14512                                 ;               FPS BEFORE EXECUTION
14513                                 ;               FPS AFTER EXECUTION
14514                                 ;               (FEC)
14515                                 ;
14516                                 ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
14517                                 ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
14518                                 ;
14519 073350  012605                  MDFSUB: MOV     (SP)+,R5                ; RETURN ADDRESS TO USE AS POINTER
14520 073352  012737  073440  000244          MOV     #50$,@#FPVEC            ;REDIRECT TRAP VECTOR
14521 073360  012702  000200                  MOV     #200,R2                 ;SET TO DOUBLE MODE FOR LOAD
14522 073364  170102                          LDFPS   R2                      ;LOAD FPS
14523 073366  172415                          LDD     (R5),AC0                ;LOAD AC0 WITH TEST DATA
14524 073370  012701  073650                  MOV     #MODGAR,R1              ;LOAD KNOWN INTO AC1
14525 073374  172511                          LDD     (R1),AC1                ;
14526 073376  010501                          MOV     R5,R1                   ;POINT TO FSRC DATA
14527 073400  062701  000004                  ADD     #4,R1
14528 073404  016502  000020                  MOV     20(R5),R2               ;GET TEST FPS
14529 073410  170102                          LDFPS   R2                      ;LOAD TEST FPS
14530                                 ;
14531 073412  171411                          MODF    (R1),AC0                ;*TEST INSTRUCTION
14532 073414  170001                  1$:     SETF                            ;WAIT FOR POSSIBLE FPA TRAP.
14533                                 ;
14534                                 ;INSTRUCTION DIDNT TRAP
14535                                 ;
14536 073416  032737  000001  003030          BIT     #1,@#FLAG                       ;VERIFY A NO TRAP CONDITION
14537 073424  001426                          BEQ     2$                      ;BRANCH IF GOOD
14538 073426  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
14539 073432  104003                          ERROR   .3                      ;FPP ERROR
14540                                                                         ;INSTRUCTION SHOULD HAVE TRAPPED
14541 073434  000167  000042                  JMP     2$                      ;REJOIN CODE
14542                                 ;
14543                                 ;INSTRUCTION TRAPPED
14544                                 ;
14545 073440  032737  000001  003030  50$:     BIT     #1,@#FLAG               ;SEE IF EXPECTING A TRAP
14546 073446  001005                          BNE     51$                     ;BRANCH IF EXPECTING A TRAP
14547 073450  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
14548 073454  104003                          ERROR   .3                      ;FPP ERROR
14549                                                                         ;INSTRUCTION WASNT SUPPOSE TO TRAP
14550 073456  000167  000020                  JMP     2$                      ;REJOIN CODE
```

M4

FLOATING POINT TESTS

```
14551 073462  012604                  51$:   MOV    (SP)+,R4              ;SEE IF PC = INSTRUCTION
14552 073464  005726                         TST    (SP)+                 ;CLEAN UP STACK
14553 073466  022704  073414                 CMP    #1$,R4                ;
14554 073472  001403                         BEQ    2$                    ;BRANCH IF GOOD COMPARE
14555 073474  004737  140132                 CALL   @#DETFPA              ;DETERMINE FLOATING POINT FAULT. $$$
14556 073500  104003                         ERROR  +3                  ;FPP ERROR
14557                                                                   ;PC WAS INCORRECT
14558                                        ;
14559                                        ;COMMON CODE FOR TRAP AND NO TRAP
14560                                        ;
14561 073502  170203                  2$:    STFPS  R3                    ;SAVE FPS
14562 073504  012702  000200                 MOV    #200,R2               ;SET FPP TO DOUBLE
14563 073510  170102                         LDFPS  R2
14564 073512  012701  003142                 MOV    #RECDST,R1            ;POINT TO RECEIVED DATA TABLE
14565                                        ;SAVE FRACTIONAL RESULT
14566 073516  174011                         STD    AC0,(R1)              ;SAVE AC0  RESULT
14567 073520  026503  000022                 CMP    22(R5),R3             ;VERIFY STATUS
14568 073524  001403                         BEQ    3$                    ;BRANCH IF GOOD
14569 073526  004737  140132                 CALL   @#DETFPA              ;DETERMINE FLOATING POINT FAULT. $$$
14570 073532  104003                         ERROR  +3                  ;FPP ERROR
14571                                                                   ;BAD FPS
14572 073534  010504                  3$:    MOV    R5,R4                 ;POINT TO EXPECTED DATA
14573 073536  062704  000010                 ADD    #10,R4
14574 073542  004767  044314          4$:    JSR    R7,DATVFR                     ;VERIFY DATA
14575 073546  005767  107346                 TST    COUNT
14576 073552  001403                         BEQ    5$                    ;BRANCH IF GOOD
14577 073554  004737  140132                 CALL   @#DETFPA              ;DETERMINE FLOATING POINT FAULT. $$$
14578 073560  104003                         ERROR  +3                  ;FPP ERROR
14579                                                                   ;BAD AC0
14580                                        ;SAVE INTEGER RESULT
14581 073562  174111                  5$:    STD    AC1,(R1)                ;SAVE AC1 RESULT
14582 073564  010504                         MOV    R5,R4               ;POINT TO EXPECTED
14583 073566  062704  000014                 ADD    #14,R4
14584 073572  004767  044264                 JSR    R7,DATVFR             ;VERIFY DATA
14585 073576  005767  107316                 TST    COUNT                 ;
14586 073602  001403                         BEQ    6$                    ;BRANCH IF GOOD
14587 073604  004737  140132                 CALL   @#DETFPA              ;DETERMINE FLOATING POINT FAULT. $$$
14588 073610  104003                         ERROR  +3                  ;FPP ERROR
14589                                                                   ;BAD AC1
14590 073612  005737  003030          6$:    TST    @#FLAG                ;SEE IF NEED TO CHECK FEC
14591 073616  001002                         BNE    7$                    ;BRANCH IF NEED TO CHECK
14592 073620  000165  000024                 JMP    24(R5)                ;RETURN FROM TEST
14593 073624  170301                  7$:    STST   R1                    ;SAVE FEC
14594 073626  016504  000024                 MOV    24(R5),R4             ;GET FEC
14595 073632  020401                         CMP    R4,R1                 ;VERIFY FEC
14596 073634  001403                         BEQ    8$                    ;BRANCH IF GOOD
14597 073636  004737  140132                 CALL   @#DETFPA              ;DETERMINE FLOATING POINT FAULT. $$$
14598 073642  104003                         ERROR  +3                  ;FPP ERROR
14599                                                                   ;BAD FEC
14600 073644  000165  000026          8$:    JMP    26(R5)              ;RETURN FROM TEST
14601                                        ;
14602 073650  177777  177777  177777 MODGAR: .WORD    1,-1,-1,-1                ;KNOWN DATA FOR AC1
      073656  177777
14603 073660                         HOP14:
14604
14607                                        ;
14608                                        ;- --------  -------  --- -  ---
```

FLOATING POINT TESTS

```
14609                                   ;TEST MODD
14610                                   ;
14611 073660                            MMODD:
14612                                   ;
14613                                   ;1/AC>FSRC=0
14614 073660  005037  003030                      CLR     @#FLAG          ;NO INTERRUPT
14615 073664  004767  001164                      JSR     R7,MDDSUB               ;DO TEST
14616 073670  012345  177777  177777              .WORD   12345,-1,-1, 1  ;AC0
      073676  177777
14617 073700  000100  000000  000000              .WORD   100,0,0,0       ;FSRC
      073706  000000
14618 073710  000000  000000  000000              .WORD   0,0,0,0         ;FRACTIONAL RESULT
      073716  000000
14619 073720  000000  000000  000000              .WORD   0,0,0,0         ;INTEGER RESULT
      073726  000000
14620 073730  000200                            .WORD   200             ; TEST FPS
14621 073732  000204                            .WORD   204             ;RESULTANT FPS
14622                                   ;2/AC=0<FSRC
14623 073734  005037  003030                      CLR     @#FLAG          ;NO INTERRUPT
14624 073740  004767  001110                      JSR     R7,MDDSUB               ;DO TEST
14625 073744  000000  000000  000000              .WORD   0,0,0,0 ;AC0
      073752  000000
14626 073754  001234  177777  000000              .WORD   1234,-1,0,0     ;FSRC
      073762  000000
14627 073764  000000  000000  000000              .WORD   0,0,0,0         ;FRACTIONAL RESULT
      073772  000000
14628 073774  000000  000000  000000              .WORD   0,0,0,0         ;INTEGER RESULT
      074002  000000
14629 074004  007717                            .WORD   7717            ; TEST FPS
14630 074006  007704                            .WORD   7704            ;RESULTANT FPS
14631                                   ;3/AC>FSRC>0
14632 074010  005037  003030                      CLR     @#FLAG          ;NO INTERRUPT
14633 074014  004767  001034                      JSR     R7,MDDSUB               ;DO TEST
14634 074020  056252  125252  125252              .WORD   56252,125252,125252,125250      ;AC0
      074026  125250
14635 074030  040300  000000  000000              .WORD   40300,0,0,0     ;FSRC
      074036  000000
14636 074040  000000  000000  000000              .WORD   0,0,0,0         ;FRACTIONAL RESULT
      074046  000000
14637 074050  056377  177777  177777              .WORD   56377,-1,-1,-4          ;INTEGER RESULT
      074056  177774
14638 074060  000213                            .WORD   213             ; TEST FPS
14639 074062  000204                            .WORD   204             ;RESULTANT FPS
14640                                   ;4/AC<0>FSRC
14641 074064  005037  003030                      CLR     @#FLAG          ;NO INTERRUPT
14642 074070  004767  000760                      JSR     R7,MDDSUB               ;DO TEST
14643 074074  140240  000000  000000              .WORD   140240,0,0,0    ;AC0
      074102  000000
14644 074104  063714  146314  133572              .WORD   63714,146314,133572,167737      ;FSRC
      074112  167737
14645 074114  000000  000000  000000              .WORD   0,0,0,0         ;FRACTIONAL RESULT
      074122  000000
14646 074124  163777  177777  162531              .WORD   163777,-1,162531,125726         ;INTEGER RESULT
      074132  125726
14647 074134  000210                            .WORD   210             ; TEST FPS
14648 074136  000204                            .WORD   204             ;RESULTANT FPS
14649                                   ;5/AC>FSRC>0
```

FLOATING POINT TESTS

```
14650 074140  005037  003030                CLR     @#FLAG          ;NO INTERRUPT
14651 074144  004767  000704                JSR     R7,MDDSUB               ;DO TEST
14652 074150  056200  000000  000000        .WORD   56200,0,0,1     ;AC0
      074156  000001
14653 074160  040340  000000  000000        .WORD   40340,0,0,0     ;FSRC
      074166  000000
14654 074170  000000  000000  000000        .WORD   0,0,0,0         ;FRACTIONAL RESULT
      074176  000000
14655 074200  056340  000000  000000        .WORD   56340,0,0,1             ;INTEGER RESULT
      074206  000001
14656 074210  000213                       .WORD   213             ; TEST FPS
14657 074212  000204                       .WORD   204             ;RESULTANT FPS
14658                             ;6/TRUNCATE
14659 074214  005037  003030                CLR     @#FLAG          ;NO INTERRUPT
14660 074220  004767  000630                JSR     R7,MDDSUB               ;DO TEST
14661 074224  056252  125252  125252        .WORD   56252,125252,125252,125252              ;AC0
      074232  125252
14662 074234  040300  000000  000000        .WORD   40300,0,0,0             ;FSRC
      074242  000000
14663 074244  000000  000000  000000        .WORD   0,0,0,0         ;FRACTIONAL  RESULT
      074252  000000
14664 074254  056377  177777  177777        .WORD   56377,-1,-1,-1          ;INTEGER RESULT
      074262  177777
14665 074264  000253                       .WORD   253             ; TEST FPS
14666 074266  000244                       .WORD   244             ;RESULT FPS
14667                             ;7/TRUNCATE FRACTION
14668 074270  005037  003030                CLR     @#FLAG          ;NO INTERRUPT
14669 074274  004767  000554                JSR     R7,MDDSUB               ;DO TEST
14670 074300  023252  125252  125252        .WORD   23252,125252,125252,125252              ;AC0
      074306  125252
14671 074310  040300  000000  000000        .WORD   40300,0,0,0             ;FSRC
      074316  000000
14672 074320  023377  177777  177777        .WORD   23377,-1,-1,-1          ;FRACTIONAL  RESULT
      074326  177777
14673 074330  000000  000000  000000        .WORD   0,0,0,0         ;INTEGER RESULT
      074336  000000
14674 074340  000253                       .WORD   253             ; TEST FPS
14675 074342  000240                       .WORD   240             ;RESULT FPS
14676                             ;8/ROUND INTEGER
14677 074344  005037  003030                CLR     @#FLAG          ;NO INTERRUPT
14678 074350  004767  000500                JSR     R7,MDDSUB               ;DO TEST
14679 074354  076600  000000  000000        .WORD   76600,0,0,125252                ;AC0
      074362  125252
14680 074364  040300  000000  000000        .WORD   40300,0,0,0             ;FSRC
      074372  000000
14681 074374  000000  000000  000000        .WORD   0,0,0,0         ;FRACTIONAL  RESULT
      074402  000000
14682 074404  076700  000000  000000        .WORD   76700,0,0,-1            ;INTEGER RESULT
      074412  177777
14683 074414  000200                       .WORD   200             ; TEST FPS
14684 074416  000204                       .WORD   204             ;RESULT FPS
14685                             ;9/ROUND THROUGH FRACTION
14686 074420  005037  003030                CLR     @#FLAG          ;NO INTERRUPT
14687 074424  004767  000424                JSR     R7,MDDSUB               ;DO TEST
14688 074430  041525  052525  052525        .WORD   41525,052525,52525,52525                ;AC0
      074436  052525
14689 074440  040300  000000  000000        .WORD   40300,0,0,0             ;FSRC
```

FLOATING POINT TESTS

```
        074446  000000
14690   074450  040177  177777  177777          .WORD   40177, 1,-1,177740            ;FRACTIONAL  RESULT
        074456  177740
14691   074460  041636  000000  000000          .WORD   41636,0,0,0                    ;INTEGER RESULT
        074466  000000
14692   074470  007700                          .WORD   7700                   ; TEST FPS
14693   074472  007700                          .WORD   7700                   ;RESULT FPS
14694                           ;/OVERFLOW, TRAPS ENABLED
14695   074474  012737  000001  003030          MOV     #1,@#FLAG                      ;INTERRUPT
14696   074502  004767  000346                  JSR     R7,MDDSUB                      ;DO TEST
14697   074506  177777  177777  177777          .WORD   -1,-1,-1, 1                    ;AC0
        074514  177777
14698   074516  040400  000000  000000          .WORD   40400,0,0,0                    ;FSRC
        074524  000000
14699   074526  000000  000000  000000          .WORD   0,0,0,0                ;FRACTIONAL  RESULT
        074534  000000
14700   074536  100177  177777  177777          .WORD   100177, 1,-1,-1                ;INTEGER RESULT
        074544  177777
14701   074546  007700                          .WORD   7700                   ; TEST FPS
14702   074550  107706                          .WORD   107706                 ;RESULT FPS
14703   074552  000010                          .WORD   10                     ;FEC
14704                           ;/INTEGER CHOPPED TO 56 BITS
14705   074554  005037  003030                  CLR     @#FLAG                 ;NO INTERRUPT
14706   074560  004767  000270                  JSR     R7,MDDSUB                      ;DO TEST
14707   074564  056700  000000  000000          .WORD   56700,0,0,-1                   ;AC0
        074572  177777
14708   074574  044440  177777  177777          .WORD   44440,-1,-1,-1                 ;FSRC
        074602  177777
14709   074604  000000  000000  000000          .WORD   0,0,0,0                ;FRACTIONAL  RESULT
        074612  000000
14710   074614  063161  100000  000001          .WORD   63161,100000,1,40775           ;INTEGER RESULT
        074622  040775
14711   074624  000200                          .WORD   200                    ; TEST FPS
14712   074626  000204                          .WORD   204                    ;RESULT FPS
14713                           ;/OVERFLOW, TRAPS DISABLED
14714   074630  012737  000002  003030          MOV     #2,@#FLAG                      ;NO INTERRUPT
14715   074636  004767  000212                  JSR     R7,MDDSUB                      ;DO TEST
14716   074642  066600  000000  000000          .WORD   66600,0,0,0                    ;AC0
        074650  000000
14717   074652  066600  000000  000000          .WORD   66600,0,0,0                    ;FSRC
        074660  000000
14718   074662  000000  000000  000000          .WORD   0,0,0,0                ;FRACTIONAL  RESULT
        074670  000000
14719   074672  015200  000000  000000          .WORD   15200,0,0,0                    ;INTEGER RESULT
        074700  000000
14720   074702  047700                          .WORD   47700                  ; TEST FPS
14721   074704  147706                          .WORD   147706                 ;RESULT FPS
14722   074706  000010                          .WORD   10                     ;FEC
14723                           ;/UNDERFLOW, TRAPS DISABLED
14724   074710  012737  000002  003030          MOV     #2,@#FLAG                      ;NO INTERRUPT
14725   074716  004767  000132                  JSR     R7,MDDSUB                      ;DO TEST
14726   074722  100277  000001  000002          .WORD   100277,1,2,-1                  ;AC0
        074730  177777
14727   074732  100300  000001  000001          .WORD   100300,1,1,1                   ;FSRC
        074740  000001
14728   074742  000000  000000  000000          .WORD   0,0,0,0                ;FRACTIONAL  RESULT
        074750  000000
```

FLOATING POINT TESTS

```
14729 074752  000000  000000  000000          .WORD   0,0,0,0          ;INTEGER RESULT
      074760  000000
14730 074762  005200                          .WORD   5200             ; TEST FPS
14731 074764  005204                          .WORD   5204             ;RESULT FPS
14732 074766  000010                          .WORD   10               ;FEC
14733                               ;/UNDERFLOW TRAPS ENABLED, UV AS RESULT
14734 074770  012737  000001  003030          MOV     #1,@#FLAG                ;INTERRUPT
14735 074776  004767  000052                  JSR     R7,MODSUB                ;DO TEST
14736 075002  100277  000001  000002          .WORD   100277,1,2,-1            ;AC0
      075010  177777
14737 075012  100300  000001  000001          .WORD   100300,1,1,1             ;FSRC
      075020  000001
14738 075022  040417  040001  077403          .WORD   40417,40001,77403,0              ;FRACTIONAL  RESULT
      075030  000000
14739 075032  000000  000000  000000          .WORD   0,0,0,0          ;INTEGER RESULT
      075040  000000
14740 075042  002200                          .WORD   2200             ; TEST FPS
14741 075044  102200                          .WORD   102200           ;RESULT FPS
14742 075046  000012                          .WORD   12               ;FEC
14743                               ;
14744 075050  000167  000300                  JMP     HOP15            ;JUMP OVER SUBROUTINE
14745                               ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
14746                               ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
14747                               ;
14748                               ;               AC0
14749                               ;               FSRC
14750                               ;               FRACTIONAL RESULT
14751                               ;               INTEGER RESULT
14752                               ;               FPS BEFORE EXECUTION
14753                               ;               FPS AFTER EXECUTION
14754                               ;               (FEC)
14755                               ;
14756                               ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
14757                               ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
14758                               ;
14759 075054  012605                  MODSUB: MOV     (SP)+,R5                 ; RETURN ADDRESS TO USE AS POINTER
14760 075056  012737  075144  000244          MOV     #50$,@#FPVEC             ;REDIRECT TRAP VECTOR
14761 075064  012702  000200                  MOV     #200,R2                  ;SET TO DOUBLE MODE FOR LOAD
14762 075070  170102                          LDFPS   R2                       ;LOAD FPS
14763 075072  172415                          LDD     (R5),AC0                 ;LOAD AC0 WITH TEST DATA
14764 075074  012701  073650                  MOV     #MODGAR,R1               ;LOAD KNOWN INTO AC1
14765 075100  172511                          LDD     (R1),AC1                 ;
14766 075102  010501                          MOV     R5,R1                    ;POINT TO FSRC DATA
14767 075104  062701  000010                  ADD     #10,R1
14768 075110  016502  000040                  MOV     40(R5),R2                ;GET TEST FPS
14769 075114  170102                          LDFPS   R2                       ;LOAD TEST FPS
14770                               ;
14771 075116  171411                          MODD    (R1),AC0                 ;*TEST INSTRUCTION
14772 075120  170011                  1$:     SETD                             ;WAIT FOR POSSIBLE FPA TRAP.
14773                               ;
14774                               ;INSTRUCTION DIDNT TRAP
14775                               ;
14776 075122  032737  000001  003030          BIT     #1,@#FLAG                        ;VERIFY A NO TRAP CONDITION
14777 075130  001426                          BEQ     2$                       ;BRANCH IF GOOD
14778 075132  004737  140132                  CALL    @#DETFPA                 ;DETERMINE FLOATING POINT FAULT. $$$
14779 075136  104003                          ERROR   +3                  ;FPP ERROR
14780                                                                     ;INSTRUCTION SHOULD HAVE TRAPPED
```

FLOATING POINT TESTS

```
14781 075140  000167  000042                    JMP     2$                      ;REJOIN CODE
14782                                           ;
14783                                           ;INSTRUCTION TRAPPED
14784                                           ;
14785 075144  032737  000001  003030  50$:      BIT     #1,@#FLAG               ;SEE IF EXPECTING A TRAP
14786 075152  001005                            BNE     51$                     ;BRANCH IF EXPECTING A TRAP
14787 075154  004737  140132                    CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
14788 075160  104003                            ERROR   .3                  ;FPP ERROR
14789                                                                       ;INSTRUCTION WASNT SUPPOSE TO TRAP
14790 075162  000167  000020                    JMP     2$                      ;REJOIN CODE
14791 075166  012604                    51$:    MOV     (SP)+,R4                ;SEE IF PC = INSTRUCTION
14792 075170  005726                            TST     (SP)+                   ;CLEAN UP STACK
14793 075172  022704  075120                    CMP     #1$,R4                  ;
14794 075176  001403                            BEQ     2$                      ;BRANCH IF GOOD COMPARE
14795 075200  004737  140132                    CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
14796 075204  104003                            ERROR   .3                  ;FPP ERROR
14797                                                                       ;PC WAS INCORRECT
14798                                           ;
14799                                           ;COMMON CODE FOR TRAP AND NO TRAP
14800                                           ;
14801 075206  170203                    2$:     STFPS   R3                      ;SAVE FPS
14802 075210  012702  000200                    MOV     #200,R2                 ;SET FPP TO DOUBLE
14803 075214  170102                            LDFPS   R2
14804 075216  012701  003142                    MOV     #RECDST,R1              ;POINT TO RECEIVED DATA TABLE
14805                                   ;SAVE FRACTIONAL RESULT
14806 075222  174011                            STD     AC0,(R1)                ;SAVE AC0  RESULT
14807 075224  026503  000042                    CMP     42(R5),R3               ;VERIFY STATUS
14808 075230  001403                            BEQ     3$                      ;BRANCH IF GOOD
14809 075232  004737  140132                    CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
14810 075236  104003                            ERROR   .3                  ;FPP ERROR
14811                                                                       ;BAD FPS
14812 075240  010504                    3$:     MOV     R5,R4                   ;POINT TO EXPECTED DATA
14813 075242  062704  000020                    ADD     #20,R4
14814 075246  004767  042626            4$:     JSR     R7,DATVER                       ;VERIFY DATA
14815 075252  005767  105642                    TST     COUNT
14816 075256  001403                            BEQ     5$                      ;BRANCH IF GOOD
14817 075260  004737  140132                    CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
14818 075264  104003                            ERROR   .3                  ;FPP ERROR
14819                                                                       ;BAD AC0
14820                                   ;SAVE INTEGER RESULT
14821                                   ;
14822 075266  174111                    5$:     STD     AC1,(R1)                ;SAVE AC1 RESULT
14823 075270  010504                            MOV     R5,R4                   ;POINT TO EXPECTED
14824 075272  062704  000030                    ADD     #30,R4
14825 075276  004767  042576                    JSR     R7,DATVER               ;VERIFY DATA
14826 075302  005767  105612                    TST     COUNT                   ;
14827 075306  001403                            BEQ     6$                      ;BRANCH IF GOOD
14828 075310  004737  140132                    CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
14829 075314  104003                            ERROR   .3                  ;FPP ERROR
14830                                                                       ;BAD AC1
14831 075316  005737  003030  6$:      TST     @#FLAG                  ;SEE IF NEED TO CHECK FEC
14832 075322  001002                            BNE     7$                      ;BRANCH IF NEED TO CHECK
14833 075324  000165  000044                    JMP     44(R5)                  ;RETURN FROM TEST
14834 075330  170301                    7$:     STST    R1                      ;SAVE FEC
14835 075332  016504  000044                    MOV     44(R5),R4               ;GET FEC
14836 075336  020401                            CMP     R4,R1                   ;VERIFY FEC
14837 075340  001403                            BEQ     8$                      ;BRANCH IF GOOD
```

F5

FLOATING POINT TESTS

```
14838 075342  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATINC POINT FAULT. $$$
14839 075346  104003                          ERROR   +3              ;FPP ERROR
14840                                                                 ;BAD FEC
14841 075350  000165  000046          8$:     JMP     46(R5)          ;RETURN FROM TEST
14842                                 ;
14843 075354                          HOP15:
14846                                 ;
14847                                 ;-----------   -------- ----------
14848                                 ;TEST STCFD
14849                                 ;
14850 075354                          MSFD:
14851                                 ;
14852                                 ;1/AC=0
14853 075354  004767  000170                  JSR     R7,SFDSUB               ;DO TEST
14854 075360  000177  000000  000000          .WORD   0177,0,0,1              ;AC0
      075366  000001
14855 075370  000000  000000  000000          .WORD   0,0,0,0         ;RESULT
      075376  000000
14856 075400  047557                          .WORD   47557           ; TEST FPS
14857 075402  047544                          .WORD   47544           ;RESULT FPS
14858                                 ;2/AC>0, TRUNCATE
14859 075404  004767  000140                  JSR     R7,SFDSUB       ;DO TEST
14860 075410  077577  177777  177777          .WORD   77577,-1,-1, 1          ;AC0
      075416  177777
14861 075420  077577  177777  000000          .WORD   77577,-1,0,0            ;RESULT
      075426  000000
14862 075430  007540                          .WORD   7540            ; TEST FPS
14863 075432  007540                          .WORD   7540            ;RESULT FPS
14864                                 ;3/AC<0, ROUND
14865 075434  004767  000110                  JSR     R7,SFDSUB               ;DO TEST
14866 075440  100377  177777  100000          .WORD   100377,-1,100000,0              ;AC0
      075446  000000
14867 075450  100377  177777  000000          .WORD   100377,-1,0,0   ;RESULT
      075456  000000
14868 075460  007517                          .WORD   7517            ; TEST FPS
14869 075462  007510                          .WORD   7510            ;RESULT FPS
14870                                 ;4/AC=-0
14871 075464  004767  000060                  JSR     R7,SFDSUB               ;DO TEST
14872 075470  100000  000000  000000          .WORD   100000,0,0,0            ;AC0
      075476  000000
14873 075500  000000  000000  000000          .WORD   0,0,0,0         ;RESULT
      075506  000000
14874 075510  007757                          .WORD   7757            ; TEST FPS
14875 075512  007744                          .WORD   7744            ;RESULT FPS
14876                                 ;5/AC<0
14877 075514  004767  000030                  JSR     R7,SFDSUB               ;DO TEST
14878 075520  125252  125252  125252          .WORD   125252,125252,125252,125252             ;AC0
      075526  125252
14879 075530  125252  125252  000000          .WORD   125252,125252,0,0               ;RESULT
      075536  000000
14880 075540  000000                          .WORD   0               ; TEST FPS
14881 075542  000010                          .WORD   10              ;RESULT FPS
14882                                 ;
14883 075544  000167  000120                  JMP     HOP16                   ;GET OVER SUBROUTINE
14884                                 ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
14885                                 ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
14886                                 ;STCFD
```

FLOATING POINT TESTS

```
14887                                   ;               AC0
14888                                   ;               RESULT
14889                                   ;               FPS BEFORE EXECUTION
14890                                   ;               FPS AFTER EXECUTION
14891                                   ;
14892                                   ;THERE CAN BE NO TRAPS WITH THE STCFD INSTRUCTION
14893                                   ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
14894                                   ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
14895                                   ;
14896 075550  012605                   SFDSUB: MOV     (SP)+,R5                ; RETURN ADDRESS TO USE AS POINTER
14897 075552  012737  075656  000244           MOV     #50$,@#FPVEC            ;REDIRECT TRAP VECTOR
14898 075560  012702  000200                   MOV     #200,R2                 ;SET TO DOUBLE MODE FOR LOAD
14899 075564  170102                           LDFPS   R2                      ;LOAD FPS
14900 075566  172415                           LDD     (R5),AC0                ;LOAD AC0 WITH TEST DATA
14901 075570  012701  003142                   MOV     #RECDST,R1              ;POINT TO RESULT AREA
14902 075574  016502  000020                   MOV     20(R5),R2               ;GET TEST FPS
14903 075600  170102                           LDFPS   R2                      ;LOAD TEST FPS
14904                                   ;
14905 075602  176011                   40$:    STCFD   AC0,(R1)                ;*TEST INSTRUCTION
14906                                   ;
14907                                   ;INSTRUCTION DIDNT TRAP
14908                                   ;VERIFY STATUS
14909                                   ;
14910 075604  170203                   2$:     STFPS   R3                      ;SAVE FPS
14911 075606  016502  000022                   MOV     22(R5),R2                       ;GET EXPECTED STATUS
14912 075612  020203                           CMP     R2,R3                   ;VERIFY STATUS
14913 075614  001403                           BEQ     3$                      ;BRANCH IF GOOD
14914 075616  004737  140132                   CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
14915 075622  104003                           ERROR   +3              ;FPP ERROR
14916                                                                  ;BAD FPS
14917 075624  010504                   3$:     MOV     R5,R4                   ;POINT TO EXPECTED DATA
14918 075626  062704  000010                   ADD     #10,R4
14919 075632  004767  042242           4$:     JSR     R7,DATVER                       ;VERIFY DATA
14920 075636  005767  105256                   TST     COUNT
14921 075642  001403                           BEQ     5$                      ;BRANCH IF GOOD
14922 075644  004737  140132                   CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
14923 075650  104003                           ERROR   +3              ;FPP ERROR
14924                                                                  ;BAD AC0
14925 075652  000165  000024           5$:     JMP     24(R5)                  ;RETURN FROM TEST
14926                                   ;
14927                                   ;INSTRUCTION TRAPPED
14928                                   ;
14929 075656  004737  140132           50$:    CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
14930 075662  104003                           ERROR   +3              ;FPP ERROR
14931                                                                  ;INSTRUCTION WASNT SUPPOSE TO TRAP
14932 075664  000165  000024                   JMP     24(R5)          ;RETURN FROM TEST
14933                                   ;
14934 075670                           HOP16:
14937                                   ;
14938                                   ;------------ ---------- - ---
14939                                   ;TEST STCDF
14940                                   ;
14941 075670                           MSDF:
14942                                   ;
14943                                   ;1/AC=0
14944 075670  005037  003030                   CLR     @#FLAG          ;NO INTERRUPT
14945 075674  004767  000220                   JSR     R7,SDFSUB               ;DO TEST
```

```
075700  000177  000000  000000          .WORD   177,0,0,0                       ;ACO
075706  000000
075710  000000  000000                  .WORD   0,0                     ;RESULT
075714  000200                          .WORD   200                     ; TEST FPS
075716  000204                          .WORD   204                     ;RESULT FPS
                                ;2/AC=-0
075720  005037  003030                  CLR     @#FLAG                  ;NO INTERRUPT
075724  004767  000170                  JSR     R7,SDFSUB                       ;DO TEST
075730  100000  000300  000200          .WORD   100000,300,200,100                      ;ACO
075736  000100
075740  000000  000000                  .WORD   0,0                     ;RESULT
075744  007777                          .WORD   7777                    ; TEST FPS
075746  007744                          .WORD   7744                    ;RESULT FPS
                                ;3/AC>0, TRUNCATE
075750  005037  003030                  CLR     @#FLAG                  ;NO INTERRUPT
075754  004767  000140                  JSR     R7,SDFSUB                       ;DO TEST
075760  055555  055555  177777          .WORD   55555,55555,-1,-1                       ;ACO
075766  177777
075770  055555  055555                  .WORD   55555,55555                     ;RESULT
075774  000240                          .WORD   240                     ; TEST FPS
075776  000240                          .WORD   240                     ;RESULT FPS
                                ;4/AC<0, ROUND TO UNDEFINED VARIABLE
076000  012737  000001  003030          MOV     #1,@#FLAG                       ;INTERRUPT
076006  004767  000106                  JSR     R7,SDFSUB                       ;DO TEST
076012  077777  177777  100000          .WORD   77777,-1,100000,0                       ;ACO
076020  000000
076022  000000  000000                  .WORD   0,0                     ;RESULT
076026  001200                          .WORD   1200                    ; TEST FPS
076030  101206                          .WORD   101206                  ;RESULT FPS
                                ;5/AC<0, ROUND
076032  005037  003030                  CLR     @#FLAG                  ;NO INTERRUPT
076036  004767  000056                  JSR     R7,SDFSUB                       ;DO TEST
076042  125252  125252  125252          .WORD   125252,125252,125252,125252                     ;ACO
076050  125252
076052  125252  125253                  .WORD   125252,125253                   ;RESULT
076056  007700                          .WORD   7700                    ; TEST FPS
076060  007710                          .WORD   7710                    ;RESULT FPS
                                ;6/ROUND TO UV, TRAPS DISABLED
076062  012737  000002  003030          MOV     #2,@#FLAG                       ;INTERRUPT
076070  004767  000024                  JSR     R7,SDFSUB                       ;DO TEST
076074  077777  177777  177777          .WORD   77777,-1,-1,0                   ;ACO
076102  000000
076104  000000  000000                  .WORD   0,0                     ;RESULT
076110  006700                          .WORD   6700                    ; TEST FPS
076112  006706                          .WORD   6706                    ;RESULT FPS
                                ;
076114  000167  000232                  JMP     HOP17                   ;GET OVER SUBROUTINE
                                ;
                                ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
                                ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
                                ;STCDF
                                ;                       ACO
                                ;                       RESULT
                                ;                       FPS BEFORE EXECUTION
                                ;                       FPS AFTER EXECUTION
                                ;
                                ;A TRAP CAN ONLY OCCUR IF ROUNDING CAUSES OVERFLOW
```

FLOATING POINT TESTS

```
14997                                   ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
14998                                   ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
14999                                   ;
15000 076120  012605                    SDFSUB: MOV     (SP)+,R5                ; RETURN ADDRESS TO USE AS POINTER
15001 076122  012737  076202  000244            MOV     #50$,@#FPVEC            ;REDIRECT TRAP VECTOR
15002 076130  012702  000200                    MOV     #200,R2                 ;SET TO DOUBLE MODE FOR LOAD
15003 076134  170102                            LDFPS   R2                      ;LOAD FPS
15004 076136  172415                            LDD     (R5),AC0                ;LOAD AC0 WITH TEST DATA
15005 076140  012701  003142                    MOV     #RECDST,R1              ;POINT TO RESULT AREA
15006 076144  016502  000014                    MOV     14(R5),R2               ;GET TEST FPS
15007 076150  170102                            LDFPS   R2                      ;LOAD TEST FPS
15008                                   ;
15009 076152  176011                    40$:    STCDF   AC0,(R1)                ;*TEST INSTRUCTION
15010 076154  170327                    1$:     STST    (PC)+                   ;WAIT FOR POSSIBLE FPA TRAP.
15011 076156  000000                            .WORD   0                       ;STORE STATUS HERE.
15012                                   ;
15013                                   ;INSTRUCTION DIDNT TRAP
15014                                   ;
15015 076160  032737  000001  003030            BIT     #1,@#FLAG                       ;VERIFY A NO TRAP CONDITION
15016 076166  001426                            BEQ     2$                      ;BRANCH IF GOOD
15017 076170  004737  140132                    CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
15018 076174  104003                            ERROR   +3              ;FPP ERROR
15019                                                                   ;INSTRUCTION SHOULD HAVE TRAPPED
15020 076176  000167  000042                    JMP     2$                      ;REJOIN CODE
15021                                   ;
15022                                   ;INSTRUCTION TRAPPED
15023                                   ;
15024 076202  032737  000001  003030    50$:    BIT     #1,@#FLAG               ;SEE IF EXPECTING A TRAP
15025 076210  001005                            BNE     51$                     ;BRANCH IF EXPECTING A TRAP
15026 076212  004737  140132                    CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
15027 076216  104003                            ERROR   +3              ;FPP ERROR
15028                                                                   ;INSTRUCTION WASNT SUPPOSE TO TRAP
15029 076220  000167  000020                    JMP     2$                      ;REJOIN CODE
15030 076224  012604                    51$:    MOV     (SP)+,R4                ;SEE IF PC - INSTRUCTION
15031 076226  005726                            TST     (SP)+                   ;CLEAN UP STACK
15032 076230  022704  076154                    CMP     #1$,R4                  ;
15033 076234  001403                            BEQ     2$                      ;BRANCH IF GOOD COMPARE
15034 076236  004737  140132                    CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
15035 076242  104003                            ERROR   +3              ;FPP ERROR
15036                                                                   ;PC WAS INCORRECT
15037                                   ;
15038                                   ;COMMON CODE FOR TRAP AND NO TRAP
15039                                   ;VERIFY STATUS
15040                                   ;
15041 076244  170203                    2$:     STFPS   R3                      ;SAVE FPS
15042 076246  016502  000016                    MOV     16(R5),R2                       ;GET EXPECTED STATUS
15043 076252  020203                            CMP     R2,R3                   ;VERIFY STATUS
15044 076254  001403                            BEQ     3$                      ;BRANCH IF GOOD
15045 076256  004737  140132                    CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
15046 076262  104003                            ERROR   +3              ;FPP ERROR
15047                                                                   ;BAD FPS
15048 076264  010504                    3$:     MOV     R5,R4                   ;POINT TO EXPECTED DATA
15049 076266  062704  000010                    ADD     #10,R4
15050 076272  004767  041564            4$:     JSR     R7,DATVFR                       ;VERIFY DATA
15051 076276  005767  104616                    TST     COUNT
15052 076302  001403                            BEQ     5$                      ;BRANCH IF GOOD
15053 076304  004737  140132                    CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
```

```
FLOATING POINT TESTS

15054 076310  104003                    ERROR   +3              ;FPP ERROR
15055                                                           ;BAD AC0
15056 076312  005737  003030      5$:     TST     @#FLAG                  ;SEE IF NEED TO CHECK FEC
15057 076316  001002                    BNE     7$                      ;BRANCH IF NEED TO CHECK
15058 076320  000165  000020            JMP     20(R5)                  ;RETURN FROM TEST
15059                                   ;
15060                                   ;VERIFY FEC
15061                                   ;
15062 076324  012704  003122      7$:     MOV     #RECFEC,R4              ;POINT TO FEC AREA
15063 076330  170314                    STST    (R4)                    ;SAVE FEC
15064 076332  021427  000010            CMP     (R4),#10                        ;VERIFY FEC FOR OVERFLOW
15065 076336  001403                    BEQ     8$                      ;BRANCH IF GOOD
15066 076340  004737  140132            CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
15067 076344  104003                    ERROR   +3              ;FPP ERROR
15068                                                           ;BAD FEC
15069 076346  000165  000020      8$:     JMP     20(R5)          ;RETURN FROM TEST
15070                                   ;
15071 076352                            HOP17:
15074                                   ;
15075                                   ;------------ - ---  -  -----  --
15076                                   ;TEST STCFD   USING ILLEGAL ACCUMULATOR
15077                                   ;
15078 076352                            MSFDI:
15079                                   ;
15080 076352  012701  040000                    MOV     #40000,R1               ;DISABLE INTERRUPTS
15081 076356  170101                            LDFPS   R1                      ;
15082 076360  176006                            STCFD   AC0,AC6                 ;*TEST ILLEGAL INSTRUCTION
15083 076362  170202                            STFPS   R2                      ;SAVE STATUS
15084 076364  170303                            STST    R3                      ;SAVE FEC
15085 076366  022702  140000                    CMP     #140000,R2              ;VERIFY FER SET
15086 076372  001403                            BEQ     1$                      ;BRANCH IF ERROR RECEIVED
15087 076374  004737  140132                    CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
15088 076400  104003                            ERROR   +3              ;FPP ERROR
15089                                                                   ;FER BIT NOT SET ON ILLEGAL INST.
15090 076402  022703  000002      1$:           CMP     #2,R3                   ;VERIFY FEC = FLOATING OPCDOE ERROR
15091 076406  001403                            BEQ     2$                      ;BRANCH IF GOOD
15092 076410  004737  140132                    CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
15093 076414  104003                            ERROR   +3              ;FPP ERROR
15094                                                                   ;FEC INCORRECT
15095 076416                            2$:
15096                                   ;
15099                                   ;
15100                                   ;------ -  -------------- ----- -
15101                                   ;TEST CLRD
15102                                   ;
15103 076416                            MCLRD:
15104                                   ;
15105 076416  012701  003764                    MOV     #TAB47,R1               ;POINT TO DATA
15106 076422  012704  000200                    MOV     #200,R4                 ;SET FPP STATUS TO DOUBLE
15107 076426  170104                            LDFPS   R4
15108 076430  172411                            LDD     (R1),AC0                ;
15109 076432  012701  003142                    MOV     #RECDST,R1              ;POINT TO DATA BUFFER
15110 076436  174011                            STD     AC0,(R1)                ;STORE GARBAGE
15111 076440  170411                            CLRD    (R1)                    ;CLEAR DATA BUFFER
15112 076442  012704  003314                    MOV     #TAB6,R4                ;VERIFY BUFFER =0
15113 076446  004767  041426                    JSR     R7,DATVER               ;
15114 076452  005767  104442                    TST     COUNT                   ;
```

```
FLOATING POINT TESTS

15115 076456  001403                        BEQ     1$                      ;BRANCH I RECDST = 0
15116 076460  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
15117 076464  104003                        ERROR   .3              ;FPP ERROR
15118                                                                 ;RECDST NOT CLEARED
15119 076466  170202                1$:     STFPS   R2                      ;SAVE STATUS
15120 076470  020227  000204                CMP     R2,#204                 ;VERIFY STATUS
15121 076474  001403                        BEQ     2$                      ;BRANCH IF GOOD
15122 076476  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
15123 076502  104003                        ERROR   .3              ;FPP ERROR
15124                                                                 ;BAD STATUS
15125 076504                        2$:
15128                                       ;
15129                                       ;- - - - - - - - - - - - - - - - -
15130                                       ;TEST CLRD, ILLEGAL ACCUMULATOR
15131                                       ;
15132 076504                        MCLRI:
15133                                       ;
15134 076504  012704  040200                MOV     #40200,R4               ;DISABLE INTERRUPTS
15135 076510  170104                        LDFPS   R4                      ;LOAD STATUS
15136 076512  170406                        CLRD    R6                      ;*TEST INSTRUCTION WITH ILLEGAL ACC
15137 076514  170203                        STFPS   R3                      ;SAVE STATUS
15138 076516  170305                        STST    R5                      ;SAVE FEC
15139 076520  022703  140200                CMP     #140200,R3              ;VERIFY ERROR
15140 076524  001403                        BEQ     1$                      ;BRANCH IF FER SET
15141 076526  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
15142 076532  104003                        ERROR   .3              ;FPP ERROR
15143                                                                 ;ERROR IN FPS
15144 076534  022705  000002          1$:   CMP     #2,R5                   ;VERIFY FEC =2 OPCODE ERROR
15145 076540  001403                        BEQ     2$                      ;BRANCH IF GOOD
15146 076542  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
15147 076546  104003                        ERROR   .3              ;FPP ERROR
15148                                                                 ;BAD FEC
15149 076550                        2$:
15152                                       ;
15153                                       ;------------------ ---------- -----
15154                                       ;TEST LDFPS, STFPS MODE 1
15155                                       ;
15156 076550                        MLS1:
15157                                       ;
15158 076550  012704  003162                MOV     #TSTLOC,R4              ;POINT R4 TO RAM
15159 076554  012714  147757                MOV     #147757,(R4)            ;SETUP EXPECTED STATUS
15160 076560  012701  003132                MOV     #RECST,R1               ;SET BUFFER FOR RECEIVED STATUS
15161 076564  012737  076650  000244        MOV     #10$,@#FPVEC            ;SETUP TRAP VECTOR
15162 076572  170114                        LDFPS   (R4)                    ;*TEST INSTRUCTION
15163 076574  170211                        STFPS   (R1)                    ;*TEST INSTRUCTION
15164 076576  020427  003162                CMP     R4,#TSTLOC              ;VERIFY R4
15165 076602  001403                        BEQ     1$                      ;BRANCH IF GOOD
15166 076604  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
15167 076610  104003                        ERROR   .3              ;FPP ERROR
15168                                                                 ;
15169 076612  020127  003132          1$:   CMP     R1,#RECST               ;VERIFY R1
15170 076616  001403                        BEQ     2$                      ;BRANCH IF GOOD
15171 076620  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
15172 076624  104003                        ERROR   .3              ;FPP ERROR
15173                                                                 ;BAD R1
15174 076626  023727  003132  147757  2$:   CMP     @#RECST,#147757         ;VERIFY STATUS
15175 076634  001412                        BEQ     3$                      ;BRANCH F GOOD
```

```
FLOATING POINT TESTS

 15176 076636  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
 15177 076642  104003                          ERROR   .3                      ;FPP ERROR
 15178                                                                         ;BAD STATUS
 15179 076644  000167  000012                  JMP     3$                      ;GET OVER TRAP
 15180                                 ;
 15181                                 ;UNEXPECTED TRAP
 15182                                 ;
 15183 076650  012600                  10$:    MOV     (SP)+,R0                        ;SAVE PC
 15184 076652  012605                          MOV     (SP)+,R5                        ;SAVE PS
 15185 076654  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
 15186 076660  104003                          ERROR   .3                      ;FPP ERROR
 15187                                                                         ;UNEXPECTED TRAP
 15188 076662                          3$:
 15191                                 ;
 15192                                 ;-------  ------  ------  -------
 15193                                 ;TEST LDFPS, STFPS MODE 2
 15194                                 ;
 15195 076662                          MLS2:
 15196                                 ;
 15197 076662  012704  003162                  MOV     #TSTLOC,R4              ;POINT R4 TO RAM
 15198 076666  012714  145557                  MOV     #145557,(R4)            ;SETUP EXPECTED STATUS
 15199 076672  012701  003132                  MOV     #RECST,R1               ;SET BUFFER FOR RECEIVED STATUS
 15200 076676  012737  076762  000244          MOV     #10$,@#FPVEC            ;SETUP TRAP VECTOR
 15201 076704  170124                          LDFPS   (R4)+                   ;*TEST INSTRUCTION
 15202 076706  170221                          STFPS   (R1)+                   ;*TEST INSTRUCTION
 15203 076710  020427  003164                  CMP     R4,#TSTLOC+2            ;VERIFY R4
 15204 076714  001403                          BEQ     1$                      ;BRANCH IF GOOD
 15205 076716  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
 15206 076722  104003                          ERROR   .3                      ;FPP ERROR
 15207                                                                         ;
 15208 076724  020127  003134          1$:     CMP     R1,#RECST+2                     ;VERIFY R1
 15209 076730  001403                          BEQ     2$                              ;BRANCH IF GOOD
 15210 076732  004737  140132                  CALL    @#DETFPA                        ;DETERMINE FLOATING POINT FAULT. $$$
 15211 076736  104003                          ERROR   .3                      ;FPP ERROR
 15212                                                                         ;BAD R1
 15213 076740  023727  003132  145557  2$:     CMP     @#RECST,#145557         ;VERIFY STATUS
 15214 076746  001412                          BEQ     3$                      ;BRANCH F GOOD
 15215 076750  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
 15216 076754  104003                          ERROR   .3                      ;FPP ERROR
 15217                                                                         ;BAD STATUS
 15218 076756  000167  000012                  JMP     3$                      ;GET OVER TRAP
 15219                                 ;
 15220                                 ;UNEXPECTED TRAP
 15221                                 ;
 15222 076762  012600                  10$:    MOV     (SP)+,R0                        ;SAVE PC
 15223 076764  012605                          MOV     (SP)+,R5                        ;SAVE PS
 15224 076766  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
 15225 076772  104003                          ERROR   .3                      ;FPP ERROR
 15226                                                                         ;UNEXPECTED TRAP
 15227 076774                          3$:
 15230                                 ;
 15231                                 ;---------------------------------
 15232                                 ;TEST LDFPS, STFPS MODE 3
 15233                                 ;
 15234 076774                          MLS3:
 15235                                 ;
 15236 076774  012704  003162                  MOV     #TSTLOC,R4              ;POINT R4 TO RAM
```

M5

FLOATING POINT TESTS

```
15237 077000  012737  003166  003162          MOV     #TSTLOC+4,@#TSTLOC      ;TSTLOC= DEFERRED ADDRESS
15238 077006  012737  147501  003166          MOV     #147501,@#TSTLOC+4      ;SETUP EXPECTED STATUS
15239 077014  012701  003172                  MOV     #TSTLOC+10,R1           ;R1 POINTS TO TSTLOC+10
15240 077020  012737  003132  003172          MOV     #RECST,@#TSTLOC+10      ;SET DEFERRED BUFFER FOR RECEIVED STATUS
15241 077026  012737  077112  000244          MOV     #10$,@#FPVEC            ;SETUP TRAP VECTOR
15242 077034  170134                          LDFPS   @(R4)+                  ;*TEST INSTRUCTION
15243 077036  170231                          STFPS   @(R1)+                  ;*TEST INSTRUCTION
15244 077040  020427  003164                  CMP     R4,#TSTLOC+2            ;VERIFY R4
15245 077044  001403                          BEQ     1$                      ;BRANCH IF GOOD
15246 077046  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
15247 077052  104003                          ERROR   +3              ;FPP ERROR
15248                                                                 ;
15249 077054  020127  003174          1$:     CMP     R1,#TSTLOC+12           ;VERIFY R1
15250 077060  001403                          BEQ     2$                      ;BRANCH IF GOOD
15251 077062  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
15252 077066  104003                          ERROR   +3              ;FPP ERROR
15253                                                                 ;BAD R1
15254 077070  023727  003132  147501  2$:     CMP     @#RECST,#147501         ;VERIFY STATUS
15255 077076  001412                          BEQ     3$                      ;BRANCH F GOOD
15256 077100  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
15257 077104  104003                          ERROR   +3              ;FPP ERROR
15258                                                                 ;BAD STATUS
15259 077106  000167  000012                  JMP     3$                      ;GET OVER TRAP
15260                                 ;
15261                                 ;UNEXPECTED TRAP
15262                                 ;
15263 077112  012600                  10$:     MOV     (SP)+,R0                        ;SAVE PC
15264 077114  012605                           MOV     (SP)+,R5                        ;SAVE PS
15265 077116  004737  140132                   CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
15266 077122  104003                           ERROR   +3              ;FPP ERROR
15267                                                                 ;UNEXPECTED TRAP
15268 077124                          3$:
15271                                 ;
15272                                 ;--------------------------------
15273                                 ;TEST LDFPS, STFPS MODE 4
15274                                 ;
15275 077124                          MLS4:
15276                                 ;
15277 077124  012704  003164                   MOV     #TSTLOC+2,R4            ;POINT R4 TO RAM
15278 077130  012737  147757  003162           MOV     #147757,@#TSTLOC        ;TSTLOC= STATUS ADDRESS
15279 077136  012701  003134                   MOV     #RECST+2,R1             ;SET BUFFER FOR RECEIVED STATUS
15280 077142  012737  077226  000244           MOV     #10$,@#FPVEC            ;SETUP TRAP VECTOR
15281 077150  170144                           LDFPS   -(R4)                   ;*TEST INSTRUCTION
15282 077152  170241                           STFPS   -(R1)                   ;*TEST INSTRUCTION
15283 077154  020427  003162                   CMP     R4,#TSTLOC              ;VERIFY R4
15284 077160  001403                           BEQ     1$                      ;BRANCH IF GOOD
15285 077162  004737  140132                   CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
15286 077166  104003                           ERROR   +3              ;FPP ERROR
15287                                                                 ;
15288 077170  020127  003132          1$:      CMP     R1,#RECST               ;VERIFY R1
15289 077174  001403                           BEQ     2$                      ;BRANCH IF GOOD
15290 077176  004737  140132                   CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
15291 077202  104003                           ERROR   +3              ;FPP ERROR
15292                                                                 ;BAD R1
15293 077204  023727  003132  147757  2$:      CMP     @#RECST,#147757         ;VERIFY STATUS
15294 077212  001412                           BEQ     3$                      ;BRANCH F GOOD
15295 077214  004737  140132                   CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
```

FLOATING POINT TESTS

```
15296 077220  104003                          ERROR   +3              ;FPP ERROR
15297                                                                 ;BAD STATUS
15298 077222  000167  000012                  JMP     3$                      ;GET OVER TRAP
15299                                 ;
15300                                 ;UNEXPECTED TRAP
15301                                 ;
15302 077226  012600                  10$:    MOV     (SP)+,R0                        ;SAVE PC
15303 077230  012605                          MOV     (SP)+,R5                        ;SAVE PS
15304 077232  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
15305 077236  104003                          ERROR   +3              ;FPP ERROR
15306                                                                 ;UNEXPECTED TRAP
15307 077240                          3$:
15310                                 ;
15311                                 ;-------- ---------------- - -
15312                                 ;TEST LDFPS, STFPS MODE 5
15313                                 ;
15314 077240                          MLS5:
15315                                 ;
15316 077240  012704  003164                  MOV     #TSTLOC+2,R4            ;POINT R4 TO RAM
15317 077244  012737  003166  003162          MOV     #TSTLOC+4,@#TSTLOC      ;TSTLOC= DEFERRED ADDRESS
15318 077252  012737  147501  003166          MOV     #147501,@#TSTLOC+4      ;SETUP EXPECTED STATUS
15319 077260  012701  003174                  MOV     #TSTLOC+12,R1           ;R1 POINTS TO 412
15320 077264  012737  003132  003172          MOV     #RECST,@#TSTLOC+10      ;SET DEFERRED BUFFER FOR RECEIVED STATUS
15321 077272  012737  077356  000244          MOV     #10$,@#FPVEC            ;SETUP TRAP VECTOR
15322 077300  170154                          LDFPS   @-(R4)                  ;*TEST INSTRUCTION
15323 077302  170251                          STFPS   @-(R1)                  ;*TEST INSTRUCTION
15324 077304  020427  003162                  CMP     R4,#TSTLOC              ;VERIFY R4
15325 077310  001403                          BEQ     1$                      ;BRANCH IF GOOD
15326 077312  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
15327 077316  104003                          ERROR   +3              ;FPP ERROR
15328                                                                 ;
15329 077320  020127  003172          1$:     CMP     R1,#TSTLOC+10                   ;VERIFY R1
15330 077324  001403                          BEQ     2$                      ;BRANCH IF GOOD
15331 077326  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
15332 077332  104003                          ERROR   +3              ;FPP ERROR
15333                                                                 ;BAD R1
15334 077334  023727  003132  147501  2$:     CMP     @#RECST,#147501         ;VERIFY STATUS
15335 077342  001412                          BEQ     3$                      ;BRANCH F GOOD
15336 077344  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
15337 077350  104003                          ERROR   +3              ;FPP ERROR
15338                                                                 ;BAD STATUS
15339 077352  000167  000012                  JMP     3$                      ;GET OVER TRAP
15340                                 ;
15341                                 ;UNEXPECTED TRAP
15342                                 ;
15343 077356  012600                  10$:    MOV     (SP)+,R0                        ;SAVE PC
15344 077360  012605                          MOV     (SP)+,R5                        ;SAVE PS
15345 077362  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
15346 077366  104003                          ERROR   +3              ;FPP ERROR
15347                                                                 ;UNEXPECTED TRAP
15348 077370                          3$:
15351                                 ;
15352                                 ; ---- ------ ------  ----   -----
15353                                 ;TEST LDFPS, STFPS MODE 6
15354                                 ;
15355 077370                          MLS6:
15356                                 ;
```

FLOATING POINT TESTS

```
15357 077370  012704  003162                  MOV     #TSTLOC,R4              ;POINT R4 TO RAM
15358 077374  012737  140001  003166          MOV     #140001,@#TSTLOC+4      ;SETUP EXPECTED STATUS
15359 077402  012701  003272                  MOV     #TSTLOC+110,R1          ;R1 WILL POINT TO TESTLOC+10
15360 077406  012737  077476  000244          MOV     #10$,@#FPVEC            ;SETUP TRAP VECTOR
15361 077414  170164  000004                  LDFPS   4(R4)                   ;*TEST INSTRUCTION
15362 077420  170261  177700                  STFPS   -100(R1)                        ;*TEST INSTRUCTION
15363 077424  020427  003162                  CMP     R4,#TSTLOC              ;VERIFY R4
15364 077430  001403                          BEQ     1$                      ;BRANCH IF GOOD
15365 077432  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
15366 077436  104003                          ERROR   +3              ;FPP ERROR
15367                                                                 ;
15368 077440  020127  003272          1$:     CMP     R1,#TSTLOC+110          ;VERIFY R1
15369 077444  001403                          BEQ     2$                      ;BRANCH IF GOOD
15370 077446  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
15371 077452  104003                          ERROR   +3              ;FPP ERROR
15372                                                                 ;BAD R1
15373 077454  023727  003172  140001  2$:     CMP     @#TSTLOC+10,#140001     ;VERIFY STATUS
15374 077462  001412                          BEQ     3$                      ;BRANCH F GOOD
15375 077464  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
15376 077470  104003                          ERROR   +3              ;FPP ERROR
15377                                                                 ;BAD STATUS
15378 077472  000167  000012                  JMP     3$                      ;GET OVER TRAP
15379                                 ;
15380                                 ;UNEXPECTED TRAP
15381                                 ;
15382 077476  012600                  10$:    MOV     (SP)+,R0                        ;SAVE PC
15383 077500  012605                          MOV     (SP)+,R5                        ;SAVE PS
15384 077502  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
15385 077506  104003                          ERROR   +3              ;FPP ERROR
15386                                                                 ;UNEXPECTED TRAP
15387 077510                          3$:
15390                                 ;
15391                                 ;---------------------------------
15392                                 ;TEST LDFPS, STFPS MODE 7
15393                                 ;
15394 077510                          MLS7:
15395                                 ;
15396 077510  012704  003262                  MOV     #TSTLOC+100,R4          ;POINT R4 TO RAM
15397 077514  012737  003166  003162          MOV     #TSTLOC+4,@#TSTLOC      ;TSTLOC= DEFERRED ADDRESS
15398 077522  012737  145501  003166          MOV     #145501,@#TSTLOC+4      ;SETUP EXPECTED STATUS
15399 077530  012701  003072                  MOV     #TSTLOC-70,R1           ;R1 POINTS TO TSTLOC+10
15400 077534  012737  003172  003164          MOV     #TSTLOC+10,@#TSTLOC+2   ;
15401 077542  012737  077632  000244          MOV     #10$,@#FPVEC            ;SETUP TRAP VECTOR
15402 077550  170174  177700                  LDFPS   @-100(R4)                       ;*TEST INSTRUCTION
15403 077554  170271  000072                  STFPS   @72(R1)                 ;*TEST INSTRUCTION
15404 077560  020427  003262                  CMP     R4,#TSTLOC+100          ;VERIFY R4
15405 077564  001403                          BEQ     1$                      ;BRANCH IF GOOD
15406 077566  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
15407 077572  104003                          ERROR   +3              ;FPP ERROR
15408                                                                 ;
15409 077574  020127  003072          1$:     CMP     R1,#TSTLOC-70           ;VERIFY R1
15410 077600  001403                          BEQ     2$                      ;BRANCH IF GOOD
1541' 07 '02  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
1541' 077606  104003                          ERROR   +3              ;FPP ERROR
15413                                                                 ;BAD R1
15414 077610  023727  003172  145501  2$:     CMP     @#TSTLOC+10,#145501     ;VERIFY STATUS
15415 077616  001412                          BEQ     3$                      ;BRANCH F GOOD
```

FLOATING POINT TESTS

```
15416 077620  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
15417 077624  104003                          ERROR   +3              ;FPP ERROR
15418                                                                 ;BAD STATUS
15419 077626  000167  000012                  JMP     3$                      ;GET OVER TRAP
15420                                 ;
15421                                 ;UNEXPECTED TRAP
15422                                 ;
15423 077632  012600                  10$:    MOV     (SP)+,R0                        ;SAVE PC
15424 077634  012605                          MOV     (SP)+,R5                        ;SAVE PS
15425 077636  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
15426 077642  104003                          ERROR   +3              ;FPP ERROR
15427                                                                 ;UNEXPECTED TRAP
15428 077644                          3$:
15431                                 ;
15432                                 ;----- -----   -----  ----      -
15433                                 ;TEST LDCLD MODE 27
15434                                 ;
15435 077644                          MLDC2:
15436                                 ;
15437 077644  005001                          CLR     R1                      ;INIT R1
15438 077646  012704  007700                  MOV     #7700,R4                ;FPS=DOUBLE, LONG
15439 077652  170104                          LDFPS   R4                      ;
15440 077654  012737  077714  000244          MOV     #10$,@#FPVEC            ;SETUP WILD TRAP
15441 077662  177027                          LDCLD   (R7)+,AC0               ;*TEST INSTRUCTION
15442 077664  005201                          INC     R1                      ;
15443 077666  005201                          INC     R1                      ;
15444 077670  005201                          INC     R1                      ;
15445 077672  005201                          INC     R1                      ;
15446 077674  020127  000003                  CMP     R1,#3                   ;VERIFY
15447 077700  001412                          BEQ     1$                      ;BRANCH IF GOOD
15448 077702  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
15449 077706  104003                          ERROR   +3              ;FPP ERROR
15450                                                                 ;INSTRUCTION FAILED
15451 077710  000167  000012                  JMP     1$                      ;JUMP OVER WILD TRAP
15452 077714  012600                  10$:    MOV     (SP)+,R0                ;SAVE PC
15453 077716  012605                          MOV     (SP)+,R5                        ;SAVE PS
15454 077720  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
15455 077724  104003                          ERROR   +3              ;FPP ERROR
15456                                                                 ;WILD TRAP ON INSTRUCTION
15457 077726  012704  003324          1$:     MOV     #TAB6A,R4               ;POINT TO EXPECTED DATA
15458 077732  012701  003142                  MOV     #RECDST,R1              ;POINT TO DATA BUFFER
15459 077736  174011                          STD     AC0,(R1)                ;VERIFY DATA
15460 077740  004767  040134                  JSR     R7,DATVER               ;
15461 077744  005767  103150                  TST     COUNT
15462 077750  001403                          BEQ     2$                      ;BRANCH IF GOOD DATA
15463 077752  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
15464 077756  104003                          ERROR   +3              ;FPP ERROR
15465                                                                 ;BAD DATA
15466 077760                          2$:
15469                                 ;
15470                                 ;---------------------- -------- -
15471                                 ;TEST LDCIF, LDCLF
15472                                 ;
15473 077760                          MLCF:
15474                                 ;
15475                                 ;1/INT=0
15476 077760  004767  000500                  JSR     R7,LCFSUB               ;DO TEST
```

FLOATING POINT TESTS

```
15477 077764  000000  000000                .WORD   0,0             ;FSRC
15478 077770  000000  000000                .WORD   0,0             ;RESULT
15479 077774  000000                        .WORD   0               ; TEST FPS
15480 077776  000004                        .WORD   4               ;RESULT FPS
15481                             ;2/INT=0,-1
15482 100000  004767  000460                JSR     R7,LCFSUB               ;DO TEST
15483 100004  000000  177777                .WORD   0,-1            ;FSRC
15484 100010  000000  000000                .WORD   0,0             ;RESULT
15485 100014  007440                        .WORD   7440            ; TEST FPS
15486 100016  007444                        .WORD   7444            ;RESULT FPS
15487                             ;3/LONG=0
15488 100020  004767  000440                JSR     R7,LCFSUB               ;DO TEST
15489 100024  000000  000000                .WORD   0,0             ;FSRC
15490 100030  000000  000000                .WORD   0,0             ;RESULT
15491 100034  000100                        .WORD   100             ; TEST FPS
15492 100036  000104                        .WORD   104             ;RESULT FPS
15493                             ;4/INT=40000
15494 100040  004767  000420                JSR     R7,LCFSUB               ;DO TEST
15495 100044  040000  000000                .WORD   40000,0         ;FSRC
15496 100050  043600  000000                .WORD   43600,0         ;RESULT
15497 100054  000017                        .WORD   17              ; TEST FPS
15498 100056  000000                        .WORD   0               ;RESULT FPS
15499                             ;5/LONG=1
15500 100060  004767  000400                JSR     R7,LCFSUB               ;DO TEST
15501 100064  000000  000001                .WORD   0,1             ;FSRC
15502 100070  040200  000000                .WORD   40200,0         ;RESULT
15503 100074  000117                        .WORD   117             ; TEST FPS
15504 100076  000100                        .WORD   100             ;RESULT FPS
15505                             ;6/INT=PATTERN
15506 100100  004767  000360                JSR     R7,LCFSUB               ;DO TEST
15507 100104  000252  025252                .WORD   252,25252       ;FSRC
15508 100110  042052  000000                .WORD   42052,0         ;RESULT
15509 100114  000000                        .WORD   0               ; TEST FPS
15510 100116  000000                        .WORD   0               ;RESULT FPS
15511                             ;7/INT=-40000
15512 100120  004767  000340                JSR     R7,LCFSUB               ;DO TEST
15513 100124  140000  000000                .WORD    40000,0        ;FSRC
15514 100130  143600  000000                .WORD   143600,0                ;RESULT
15515 100134  000007                        .WORD   7               ; TEST FPS
15516 100136  000010                        .WORD   10              ;RESULT FPS
15517                             ;8/INT=-1
15518 100140  004767  000320                JSR     R7,LCFSUB               ;DO TEST
15519 100144  177777  000000                .WORD   -1,0            ;FSRC
15520 100150  140200  000000                .WORD   140200,0                ;RESULT
15521 100154  000007                        .WORD   7               ; TEST FPS
15522 100156  000010                        .WORD   10              ;RESULT FPS
15523                             ;9/INT=PATTERN
15524 100160  004767  000300                JSR     R7,LCFSUB               ;DO TEST
15525 100164  125252  125252                .WORD   125252,125252           ;FSRC
15526 100170  143652  126000                .WORD   143652,126000           ;RESULT
15527 100174  000007                        .WORD   7               ; TEST FPS
15528 100176  000010                        .WORD   10              ;RESULT FPS
15529                             ;10/LONG=40000
15530 100200  004767  000260                JSR     R7,LCFSUB               ;DO TEST
15531 100204  040000  000000                .WORD   40000,0         ;FSRC
15532 100210  047600  000000                .WORD   47600,0         ;RESULT
15533 100214  000117                        .WORD   117             ; TEST FPS
```

FLOATING POINT TESTS

```
15534 100216  000100                    .WORD   100             ;RESULT FPS
15535                           ;11/LONG=1
15536 100220  004767  000240            JSR     R7,LCFSUB               ;DO TEST
15537 100224  000000  000001            .WORD   0,1             ;FSRC
15538 100230  040200  000000            .WORD   40200,0         ;RESULT
15539 100234  007557                    .WORD   7557            ; TEST FPS
15540 100236  007540                    .WORD   7540            ;RESULT FPS
15541                           ;12/LONG=PATTERN
15542 100240  004767  000220            JSR     R7,LCFSUB               ;DO TEST
15543 100244  000000  000252            .WORD   0,252           ;FSRC
15544 100250  042052  000000            .WORD   42052,0         ;RESULT
15545 100254  007557                    .WORD   7557            ; TEST FPS
15546 100256  007540                    .WORD   7540            ;RESULT FPS
15547                           ;13/LONG = -40000
15548 100260  004767  000200            JSR     R7,LCFSUB               ;DO TEST
15549 100264  140000  000000            .WORD   -40000,0                ;FSRC
15550 100270  147600  000000            .WORD   147600,0                ;RESULT
15551 100274  000107                    .WORD   107             ; TEST FPS
15552 100276  000110                    .WORD   110             ;RESULT FPS
15553                           ;14/LONG=-1
15554 100300  004767  000160            JSR     R7,LCFSUB               ;DO TEST
15555 100304  177777  177777            .WORD   -1,-1           ;FSRC
15556 100310  140200  000000            .WORD   140200,0                ;RESULT
15557 100314  007500                    .WORD   7500            ; TEST FPS
15558 100316  007510                    .WORD   7510            ;RESULT FPS
15559                           ;15/LONG=PATTERN
15560 100320  004767  000140            JSR     R7,LCFSUB               ;DO TEST
15561 100324  125252  125252            .WORD   125252,125252           ;FSRC
15562 100330  147652  125253            .WORD   147652,125253           ;RESULT
15563 100334  000105                    .WORD   105             ; TEST FPS
15564 100336  000110                    .WORD   110             ;RESULT FPS
15565                           ;16/LONG=77777,177500
15566 100340  004767  000120            JSR     R7,LCFSUB               ;DO TEST
15567 100344  077777  177500            .WORD   77777,177500            ;FSRC
15568 100350  047777  177777            .WORD   47777,177777            ;RESULT
15569 100354  000117                    .WORD   117             ; TEST FPS
15570 100356  000100                    .WORD   100             ;RESULT FPS
15571                           ;17/LONG=40000,100
15572 100360  004767  000100            JSR     R7,LCFSUB               ;DO TEST
15573 100364  040000  000100            .WORD   40000,100               ;FSRC
15574 100370  047600  000001            .WORD   47600,1         ;RESULT
15575 100374  007502                    .WORD   7502            ; TEST FPS
15576 100376  007500                    .WORD   7500            ;RESULT FPS
15577                           ;18/LONG=40000,100 - TRUNCATE
15578 100400  004767  000060            JSR     R7,LCFSUB               ;DO TEST
15579 100404  040000  000100            .WORD   40000,100               ;FSRC
15580 100410  047600  000000            .WORD   47600,0         ;RESULT
15581 100414  007557                    .WORD   7557            ; TEST FPS
15582 100416  007540                    .WORD   7540            ;RESULT FPS
15583                           ;19/INT= MOST NEGATIVE
15584 100420  004767  000040            JSR     R7,LCFSUB               ;DO TEST
15585 100424  100000  000000            .WORD   100000,0                ;FSRC
15586 100430  144000  000000            .WORD   144000,0                ;RESULT
15587 100434  000007                    .WORD   7               ; TEST FPS
15588 100436  000010                    .WORD   10              ;RESULT FPS
15589                           ;20/LONG= MOST NEGATIVE
15590 100440  004767  000020            JSR     R7,LCFSUB               ;DO TEST
```

F6

FLOATING POINT TESTS

```
15591 100444  100000  000000                .WORD   100000,0                ;FSRC
15592 100450  150000  000000                .WORD   150000,0                ;RESULT
15593 100454  000107                        .WORD   107             ; TEST FPS
15594 100456  000110                        .WORD   110             ;RESULT FPS
15595                               ;
15596 100460  000167  000126                JMP     HOP18           ;GET OVER SUBROUTINE
15597                               ;
15598                               ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
15599                               ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
15600                               ;LDCIF, LDCLF
15601                               ;
15602                               ;               FSRC
15603                               ;               RESULT
15604                               ;               FPS BEFORE EXECUTION
15605                               ;               FPS AFTER EXECUTION
15606                               ;
15607                               ;NO TRAP CAN OCCUR
15608                               ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
15609                               ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
15610                               ;
15611 100464  012602                LCFSUB: MOV     (SP)+,R2        ; RETURN ADDRESS TO USE AS POINTER
15612 100466  012737  100574  000244        MOV     #50$,@#FPVEC            ;REDIRECT TRAP VECTOR
15613 100474  012701  003142                MOV     #RECDST,R1              ;POINT TO RESULT AREA
15614 100500  016200  000010                MOV     10(R2),R0               ;GET TEST FPS
15615 100504  170100                        LDFPS   R0                      ;LOAD TEST FPS
15616 100506  010204                        MOV     R2,R4                   ;POINT TO TEST DATA
15617                               ;
15618 100510  177014                40$:    LDCIF   (R4),AC0                ;*TEST INSTRUCTION (ACCORDING TO MODE)
15619                               ;
15620                               ;VERIFY STATUS
15621                               ;
15622 100512  170203                2$:     STFPS   R3                      ;SAVE FPS
15623 100514  012700  000200                MOV     #200,R0                 ;SET FPP STATUS TO DOUBLE
15624 100520  170100                        LDFPS   R0                      ;
15625 100522  174011                        STD     AC0,(R1)                ;SAVE TEST RESULT INTO RECDST
15626 100524  016200  000012                MOV     12(R2),R0                       ;GET EXPECTED STATUS
15627 100530  020003                        CMP     R0,R3                   ;VERIFY STATUS
15628 100532  001403                        BEQ     3$                      ;BRANCH IF GOOD
15629 100534  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
15630 100540  104003                        ERROR   .+3             ;FPP ERROR
15631                                                                 ;BAD FPS
15632 100542  010204                3$:     MOV     R2,R4                   ;POINT TO EXPECTED DATA
15633 100544  062704  000004                ADD     #4,R4
15634 100550  004767  037306        4$:     JSR     R7,DATVFR                       ;VERIFY DATA
15635 100554  005767  102340                TST     COUNT
15636 100560  001403                        BEQ     5$                      ;BRANCH IF GOOD
15637 100562  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
15638 100566  104003                        ERROR   .+3             ;FPP ERROR
15639                                                                 ;BAD AC0
15640 100570  000162  000014        5$:     JMP     14(R2)                  ;RETURN FROM TEST
15641                               ;
15642                               ;INSTRUCTION TRAPPED
15643                               ;
15644 100574  012600                50$:    MOV     (SP)+,R0                ;SAVE PC
15645 100576  012605                        MOV     (SP)+,R5                ;SAVE PS
15646 100600  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
15647 100604  104003                        ERROR   .+3             ;FPP ERROR
```

FLOATING PCINT TESTS

```
15648                                                   ;INSTRUCTION WASNT SUPPOSE TO TRAP
15649 100606  000167  177756            JMP     5$                      ;CONTINUE
15650 100612                          HOP18:
15653                                 ;
15654                                 ; ----- ----- -- ------- -------
15655                                 ;TEST LDCID, LDCLD
15656                                 ;
15657 100612                          MLCD:
15658                                 ;
15659                                 ;1/LONG=0
15660 100612  004767  000264                  JSR     R7,LCDSUB               ;DO TEST
15661 100616  000000  000000                  .WORD   0,0             ;FSRC
15662 100622  000000  000000  000000          .WORD   0,0,0,0         ;RESULT
      100630  000000
15663 100632  007313                          .WORD   7313            ; TEST FPS
15664 100634  007304                          .WORD   7304            ;RESULT FPS
15665                                 ;2/INT=0
15666 100636  004767  000240                  JSR     R7,LCDSUB               ;DO TEST
15667 100642  000000  000001                  .WORD   0,1             ;FSRC
15668 100646  040200  000000  000000          .WORD   40200,0,0,0             ;RESULT
      100654  000000
15669 100656  007757                          .WORD   7757            ; TEST FPS
15670 100660  007740                          .WORD   7740            ;RESULT FPS
15671                                 ;3/INT=40000
15672 100662  004767  000214                  JSR     R7,LCDSUB               ;DO TEST
15673 100666  040000  177777                  .WORD   40000,-1                ;FSRC
15674 100672  043600  000000  000000          .WORD   43600,0,0,0             ;RESULT
      100700  000000
15675 100702  007617                          .WORD   7617            ; TEST FPS
15676 100704  007600                          .WORD   7600            ;RESULT FPS
15677                                 ;4/INT=-40000
15678 100706  004767  000170                  JSR     R7,LCDSUB               ;DO TEST
15679 100712  140000  177777                  .WORD   -40000,-1               ;FSRC
15680 100716  143600  000000  000000          .WORD   143600,0,0,0            ;RESULT
      100724  000000
15681 100726  007600                          .WORD   7600            ; TEST FPS
15682 100730  007610                          .WORD   7610            ;RESULT FPS
15683                                 ;5/LONG=40000
15684 100732  004767  000144                  JSR     R7,LCDSUB               ;DO TEST
15685 100736  040000  000000                  .WORD   40000,0         ;FSRC
15686 100742  047600  000000  000000          .WORD   47600,0,0,0             ;RESULT
      100750  000000
15687 100752  007757                          .WORD   7757            ; TEST FPS
15688 100754  007740                          .WORD   7740            ;RESULT FPS
15689                                 ;6/LONG=1
15690 100756  004767  000120                  JSR     R7,LCDSUB               ;DO TEST
15691 100762  000000  000001                  .WORD   0,1             ;FSRC
15692 100766  040200  000000  000000          .WORD   40200,0,0,0             ;RESULT
      100774  000000
15693 100776  000300                          .WORD   300             ; TEST FPS
15694 101000  000300                          .WORD   300             ;RESULT FPS
15695                                 ;7/LONG=-2
15696 101002  004767  000074                  JSR     R7,LCDSUB               ;DO TEST
15697 101006  177777  177776                  .WORD   -1,-2           ;FSRC
15698 101012  140400  000000  000000          .WORD   140400,0,0,0            ;RESULT
      101020  000000
15699 101022  007300                          .WORD   7300            ; TEST FPS
```

H6

FLOATING POINT TESTS

```
15700 101024  007310                          .WORD   7310            ;RESULT FPS
15701                               ;8/INT=PATTERN
15702 101026  004767  000050                  JSR     R7,LCDSUB               ;DO TEST
15703 101032  123456  176543                  .WORD   123456,176543           ;FSRC
15704 101036  143661  122000  000000          .WORD   143661,122000,0,0               ;RESULT
      101044  000000
15705 101046  000200                          .WORD   200             ; TEST FPS
15706 101050  000210                          .WORD   210             ;RESULT FPS
15707                               ;9/LONG=PATTERN
15708 101052  004767  000024                  JSR     R7,LCDSUB               ;DO TEST
15709 101056  125252  125252                  .WORD   125252,125252           ;FSRC
15710 101062  147652  125252  126000          .WORD   147652,125252,126000,0          ;RESULT
      101070  000000
15711 101072  000300                          .WORD   300             ; TEST FPS
15712 101074  000310                          .WORD   310             ;RESULT FPS
15713                               ;
15714 101076  000167  000126                  JMP     HOP19           ;GET OVER SUBROUTINE
15715                               ;
15716                               ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
15717                               ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
15718                               ;LDCID, LDCLD
15719                               ;
15720                               ;               FSRC
15721                               ;               RESULT
15722                               ;               FPS BEFORE EXECUTION
15723                               ;               FPS AFTER EXECUTION
15724                               ;
15725                               ;NO TRAP CAN OCCUR
15726                               ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
15727                               ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
15728                               ;
15729 101102  012602                LCDSUB: MOV     (SP)+,R2                ; RETURN ADDRESS TO USE AS POINTER
15730 101104  012737  101212  000244          MOV     #50$,@#FPVEC            ;REDIRECT TRAP VECTOR
15731 101112  012701  003142                  MOV     #RECDST,R1              ;POINT TO RESULT AREA
15732 101116  016200  000014                  MOV     14(R2),R0               ;GET TEST FPS
15733 101122  170100                          LDFPS   R0                      ;LOAD TEST FPS
15734 101124  010204                          MOV     R2,R4                   ;POINT TO TEST DATA
15735                               ;
15736 101126  177014                40$:      LDCID   (R4),AC0                ;*TEST INSTRUCTION (ACCORDING TO MODE)
15737                               ;
15738                               ;VERIFY STATUS
15739                               ;
15740 101130  170203                2$:       STFPS   R3                      ;SAVE FPS
15741 101132  012700  000200                  MOV     #200,R0                 ;SET FPP STATUS TO DOUBLE
15742 101136  170100                          LDFPS   R0                      ;
15743 101140  174011                          STD     AC0,(R1)                ;SAVE TEST RESULT INTO RECDST
15744 101142  016200  000016                  MOV     16(R2),R0                       ;GET EXPECTED STATUS
15745 101146  020003                          CMP     R0,R3                   ;VERIFY STATUS
15746 101150  001403                          BEQ     3$                      ;BRANCH IF GOOD
15747 101152  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
15748 101156  104003                          ERROR   .+3             ;FPP ERROR
15749                                                                 ;BAD FPS
15750 101160  010204                3$:       MOV     R2,R4                           ;POINT TO EXPECTED DATA
15751 101162  062704  000004                  ADD     #4,R4
15752 101166  004767  036706          4$:     JSR     R7,DATVER                       ;VERIFY DATA
15753 101172  005767  101722                  TST     COUNT
15754 101176  001403                          BEQ     5$                      ;BRANCH IF GOOD
```

FLOATING POINT TESTS

```
15755 101200  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
15756 101204  104003                          ERROR   +3              ;FPP ERROR
15757                                                                 ;BAD ACO
15758 101206  000162  000020          5$:     JMP     20(R2)                  ;RETURN FROM TEST
15759                                       ;
15760                                       ;INSTRUCTION TRAPPED
15761                                       ;
15762 101212  012600                  50$:    MOV     (SP)+,R0                ;SAVE PC
15763 101214  012605                          MOV     (SP)+,R5                ;SAVE PS
15764 101216  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
15765 101222  104003                          ERROR   +3              ;FPP ERROR
15766                                                                 ;INSTRUCTION WASNT SUPPOSE TO TRAP
15767 101224  000167  177756                  JMP     5$                      ;CONTINUE
15768                                       ;
15769 101230                          HOP19:
15772                                       ;
15773                                       ;----   -------- -  ----- -------
15774                                       ;TEST LDEXP
15775                                       ;DOUBLE
15776                                       ;
15777 101230                          MLXP:
15778                                       ;
15779                                       ;1/EXP=10 - AC=NEG
15780 101230  005037  003030                  CLR     @#FLAG                  ;NO INTERRUPTS
15781 101234  004767  001140                  JSR     R7,LXPSUB               ;DO TEST
15782 101240  123456  067012  025252          .WORD   123456,67012,25252,171717       ;ACO
      101246  171717
15783 101250  000010                          .WORD   10                      ;EXP
15784 101252  142056  067012  025252          .WORD   142056,67012,25252,171717               ;RESULT
      101260  171717
15785 101262  007757                          .WORD   7757            ; TEST FPS
15786 101264  007750                          .WORD   7750            ;RESULT FPS
15787                                 ;2/EXP=177 - ACO=POS
15788 101266  005037  003030                  CLR     @#FLAG                  ;NO INTERRUPTS
15789 101272  004767  001102                  JSR     R7,LXPSUB               ;DO TEST
15790 101276  023456  070123  100000          .WORD   23456,70123,100000,1            ;ACO
      101304  000001
15791 101306  000177                          .WORD   177             ;EXP
15792 101310  077656  070123  100000          .WORD   77656,70123,100000,1            ;RESULT
      101316  000001
15793 101320  007700                          .WORD   7700            ; TEST FPS
15794 101322  007700                          .WORD   7700            ;RESULT FPS
15795                                 ;3/EXP=56
15796 101324  005037  003030                  CLR     @#FLAG                  ;NO INTERRUPTS
15797 101330  004767  001044                  JSR     R7,LXPSUB               ;DO TEST
15798 101334  055555  044444  033333          .WORD   55555,44444,33333,22222         ;ACO
      101342  022222
15799 101344  000056                          .WORD   56              ;EXP
15800 101346  053555  044444  033333          .WORD   53555,44444,33333,22222 ;RESULT
      101354  022222
15801 101356  007757                          .WORD   7757            ; TEST FPS
15802 101360  007740                          .WORD   7740            ;RESULT FPS
15803                                 ;4/EXP=-151, ACO=UV
15804 101362  005037  003030                  CLR     @#FLAG                  ;NO INTERRUPTS
15805 101366  004767  001006                  JSR     R7,LXPSUB               ;DO TEST
15806 101372  100077  177777  177777          .WORD   100077, 1,-1, 2         ;ACO
      101400  177776
```

FLOATING POINT TESTS

```
15807 101402  177623                          .WORD   -155              ;EXP
15808 101404  104677  177777  177777          .WORD   104677, 1,-1, 2           ;RESULT
      101412  177776
15809 101414  007757                          .WORD   7757              ; TEST FPS
15810 101416  007750                          .WORD   7750              ;RESULT FPS
15811                               ;5/EXP=-177
15812 101420  005037  003030                  CLR     @#FLAG                    ;NO INTERRUPTS
15813 101424  004767  000750                  JSR     R7,LXPSUB                 ;DO TEST
15814 101430  000177  177777  177777          .WORD   177,-1, 1, 2              ;ACO
      101436  177776
15815 101440  177601                          .WORD   -177              ;EXP
15816 101442  000377  177777  177777          .WORD   377,-1, 1,-2              ;RESULT
      101450  177776
15817 101452  007700                          .WORD   7700              ; TEST FPS
15818 101454  007700                          .WORD   7700              ;RESULT FPS
15819                               ;6/EXP=-200, UNDERFLOW
15820 101456  012737  000001  003030          MOV     #1,@#FLAG                 ; INTERRUPTS
15821 101464  004767  000710                  JSR     R7,LXPSUB                 ;DO TEST
15822 101470  030131  032334  035363          .WORD   30131,32334,35363,73031           ;ACO
      101476  073031
15823 101500  177600                          .WORD   -200              ;EXP
15824 101502  000131  032334  035363          .WORD   131,32334,35363,73031             ;RESULT
      101510  073031
15825 101512  007740                          .WORD   7740              ; TEST FPS
15826 101514  107744                          .WORD   107744            ;RESULT FPS
15827 101516  000012                          .WORD   12                ;FEC
15828                               ;7/EXP=LARGEST NEGATIVE
15829 101520  012737  000001  003030          MOV     #1,@#FLAG                 ;EXPECT  INTERRUPTS
15830 101526  004767  000646                  JSR     R7,LXPSUB                 ;DO TEST
15831 101532  000000  000123  000456          .WORD   0,123,456,1               ;ACO
      101540  000001
15832 101542  100000                          .WORD   100000            ;EXP
15833 101544  040000  000123  000456          .WORD   40000,123,456,1           ;RESULT
      101552  000001
15834 101554  002200                          .WORD   2200              ; TEST FPS
15835 101556  102200                          .WORD   102200            ;RESULT FPS
15836 101560  000012                          .WORD   12                ;FEC
15837                               ;8/EXP=-200, NEG. ACO
15838 101562  012737  000001  003030          MOV     #1,@#FLAG                 ; INTERRUPTS
15839 101570  004767  000604                  JSR     R7,LXPSUB                 ;DO TEST
15840 101574  111111  100000  100000          .WORD   111111,100000,100000,-1           ;ACO
      101602  177777
15841 101604  177600                          .WORD   -200              ;EXP
15842 101606  100111  100000  100000          .WORD   100111,100000,100000,-1           ;RESULT
      101614  177777
15843 101616  002217                          .WORD   2217              ; TEST FPS
15844 101620  102214                          .WORD   102214            ;RESULT FPS
15845 101622  000012                          .WORD   12                ;FEC
15846                               ;9/EXP= 1743, FIU=0
15847 101624  012737  000002  003030          MOV     #2,@#FLAG                 ;NO INTERRUPTS
15848 101632  004767  000542                  JSR     R7,LXPSUB                 ;DO TEST
15849 101636  123456  012346  012346          .WORD   123456,12346,12346,123            ;ACO
      101644  000123
15850 101646  176035                          .WORD    1743             ;EXP
15851 101650  000000  000000  000000          .WORD   0,0,0,0           ;RESULT
      101656  000000
15852 101660  005700                          .WORD   5700              ; TEST FPS
```

FLOATING POINT TESTS

```
15853 101662  005704                          .WORD   5704            ;RESULT FPS
15854 101664  000012                          .WORD   12              ;FEC
15855                           ;10/EXP - 16616, FID=1
15856 101666  012737  000002  003030          MOV     #2,@#FLAG               ;NO INTERRUPTS
15857 101674  004767  000500                  JSR     R7,LXPSUB               ;DO TEST
15858 101700  000377  123456  065432          .WORD   377,123456,65432,1              ;AC0
      101706  000001
15859 101710  161162                          .WORD   -16616          ;EXP
15860 101712  074577  123456  065432          .WORD   74577,123456,65432,1            ;RESULT
      101720  000001
15861 101722  047700                          .WORD   47700           ; TEST FPS
15862 101724  147700                          .WORD   147700          ;RESULT FPS
15863 101726  000012                          .WORD   12              ;FEC
15864                           ;11/EXP=177, AC0=UNDEFINED VARIABLE
15865 101730  005037  003030                  CLR     @#FLAG          ;NO INTERRUPTS
15866 101734  004767  000440                  JSR     R7,LXPSUB               ;DO TEST
15867 101740  100177  177777  177777          .WORD   100177,-1, 1, 1         ;AC0
      101746  177777
15868 101750  000177                          .WORD   177             ;EXP
15869 101752  177777  177777  177777          .WORD   -1,-1, 1,-1             ;RESULT
      101760  177777
15870 101762  007700                          .WORD   7700            ; TEST FPS
15871 101764  007710                          .WORD   7710            ;RESULT FPS
15872                           ;12/EXP=150 AC0=POS
15873 101766  005037  003030                  CLR     @#FLAG          ;NO INTERRUPT
15874 101772  004767  000402                  JSR     R7,LXPSUB               ;DO TEST
15875 101776  000200  000100  000200          .WORD   200,100,200,300         ;AC0
      102004  000300
15876 102006  000150                          .WORD   150             ;EXP
15877 102010  072000  000100  000200          .WORD   72000,100,200,300               ;RESULT
      102016  000300
15878 102020  007717                          .WORD   7717            ; TEST FPS
15879 102022  007700                          .WORD   7700            ;RESULT FPS
15880                           ;13/EXP=200, AC0=NEG
15881 102024  012737  000001  003030          MOV     #1,@#FLAG
15882 102032  004767  000342                  JSR     R7,LXPSUB               ;DO TEST
15883 102036  177777  177777  177777          .WORD   -1,-1,-1,-1             ;AC0
      102044  177777
15884 102046  000200                          .WORD   200             ;EXP
15885 102050  100177  177777  177777          .WORD   100177,-1,-1,-1         ;RESULT
      102056  177777
15886 102060  007705                          .WORD   7705            ; TEST FPS
15887 102062  107716                          .WORD   107716          ;RESULT FPS
15888 102064  000010                          .WORD   10              ;FEC
15889                           ;14/EXP=400, FID
15890 102066  012737  000002  003030          MOV     #2,@#FLAG               ;INTERRUPT
15891 102074  004767  000300                  JSR     R7,LXPSUB               ;DO TEST
15892 102100  000555  177777  177776          .WORD   555, 1,-2, 3            ;AC0
      102106  177775
15893 102110  000400                          .WORD   400                     ;EXP
15894 102112  040155  177777  177776          .WORD   40155,-1, 2,-3          ;RESULT
      102120  177775
15895 102122  047700                          .WORD   47700           ; TEST FPS
15896 102124  147702                          .WORD   147702          ;RESULT FPS
15897 102126  000010                          .WORD   10              ;FEC
15898                           ;15/EXP=11011 FIU=0
15899 102130  012737  000000  003030          MOV     #0,@#FLAG               ;NO INTERRUPT
```

FLOATING POINT TESTS

```
15900 102136  004767  000236                JSR     R7,LXPSUB               ;DO TEST
15901 102142  177773  177777  177776        .WORD   177773, 1,-2,-3         ;ACO
      102150  177775
15902 102152  011011                        .WORD   11011           ;EXP
15903 102154  000000  000000  000000        .WORD   0,0,0,0         ;RESULT
      102162  000000
15904 102164  006700                        .WORD   6700            ; TEST FPS
15905 102166  006706                        .WORD   6706            ;RESULT FPS
15906                       ;16/EXP=LARGEST POSITIVE
15907 102170  012737  000001  003030        MOV     #1,@#FLAG               ;INTERRUPT
15908 102176  004767  000176                JSR     R7,LXPSUB               ;DO TEST
15909 102202  123456  000100  000100        .WORD   123456,100,100,200              ;ACO
      102210  000200
15910 102212  077777                        .WORD   77777           ;EXP
15911 102214  137656  000100  000100        .WORD   137656,100,100,200              ;RESULT
      102222  000200
15912 102224  007740                        .WORD   7740            ; TEST FPS
15913 102226  107752                        .WORD   107752          ;RESULT FPS
15914 102230  000010                        .WORD   10              ;FEC
15915                       ;17/FLOATING
15916 102232  005037  003030                CLR     @#FLAG          ;NO INTERRUPT
15917 102236  004767  000136                JSR     R7,LXPSUB               ;DO TEST
15918 102242  123456  023465  000555        .WORD   123456,23465,555,444            ;ACO
      102250  000444
15919 102252  000050                        .WORD   50              ;EXP
15920 102254  152056  023465  000555        .WORD   152056,23465,555,444            ;RESULT
      102262  000444
15921 102264  007500                        .WORD   7500            ; TEST FPS
15922 102266  007510                        .WORD   7510            ;RESULT FPS
15923                       ;18/FLOATING UNDERFLOW
15924 102270  012737  000001  003030        MOV     #1,@#FLAG               ;INTERRUPT
15925 102276  004767  000076                JSR     R7,LXPSUB               ;DO TEST
15926 102302  000333  000444  000555        .WORD   333,444,555,666         ;ACO
      102310  000666
15927 102312  177600                        .WORD   -200            ;EXP
15928 102314  000133  000444  000555        .WORD   133,444,555,666         ;RESULT
      102322  000666
15929 102324  007500                        .WORD   7500            ; TEST FPS
15930 102326  107504                        .WORD   107504          ;RESULT FPS
15931 102330  000012                        .WORD   12              ;FEC
15932                       ;19/FLOATING OVERFLOW
15933 102332  012737  000001  003030        MOV     #1,@#FLAG               ;INTERRUPT
15934 102340  004767  000034                JSR     R7,LXPSUB               ;DO TEST
15935 102344  012346  000123  000345        .WORD   12346,123,345,456               ;ACO
      102352  000456
15936 102354  000400                        .WORD   400             ;EXP
15937 102356  040146  000123  000345        .WORD   40146,123,345,456               ;RESULT
      102364  000456
15938 102366  007400                        .WORD   7400            ; TEST FPS
15939 102370  107402                        .WORD   107402          ;RESULT FPS
15940 102372  000010                        .WORD   10              ;FEC
15941                       ;
15942 102374  000167  000250                JMP     HOP20                   ;GET OVER SUBROUTINE
15943                       ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
15944                       ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
15945                       ;LDEXP
15946                       ;               ACO
```

```
                                        ;               EXPONENT
                                        ;               RESULT
                                        ;               FPS BEFORE EXECUTION
                                        ;               FPS AFTER EXECUTION
                                        ;               (FEC)
                                        ;
                                        ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
                                        ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
                                        ;
102400  012602                          LXPSUB: MOV     (SP)+,R2                ; RETURN ADDRESS TO USE AS POINTER
102402  012737  102470  000244                  MOV     #50$,@#FPVEC            ;REDIRECT TRAP VECTOR
102410  012701  003142                          MOV     #RECDST,R1              ;POINT TO RESULT AREA
102414  012700  000200                          MOV     #200,R0                 ;SET FPS TO DOUBLE
102420  170100                                  LDFPS   R0                      ;
102422  010204                                  MOV     R2,R4                   ;POINT TO ACO DATA
102424  172414                                  LDD     (R4),ACO                        ;LOAD ACO
102426  016200  000022                          MOV     22(R2),R0               ;GET TEST FPS
102432  170100                                  LDFPS   R0                      ;LOAD TEST FPS
102434  016204  000010                          MOV     10(R2),R4               ;POINT TO TEST DATA
                                        ;
102440  176404                          40$:    LDEXP   R4,ACO                  ;*TEST INSTRUCTION (ACCORDING TO MODE)
102442  170327                          1$:     STST    (PC)+                   ;WAIT FOR POSSIBLE FPA TRAP.
102444  000000                                  .WORD   0                       ;STORE STATUS HERE
                                        ;
                                        ;INSTRUCTION DIDNT TRAP
                                        ;
102446  032737  000001  003030                  BIT     #1,@#FLAG                       ;VERIFY A NO TRAP CONDITION
102454  001426                                  BEQ     2$                      ;BRANCH IF GOOD
102456  004737  140132                          CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
102462  104003                                  ERROR   +3              ;FPP ERROR
                                                                        ;INSTRUCTION SHOULD HAVE TRAPPED
102464  000167  000042                          JMP     2$                      ;REJOIN CODE
                                        ;
                                        ;INSTRUCTION TRAPPED
                                        ;
102470  032737  000001  003030          50$:    BIT     #1,@#FLAG               ;SEE IF EXPECTING A TRAP
102476  001005                                  BNE     51$                     ;BRANCH IF EXPECTING A TRAP
102500  004737  140132                          CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
102504  104003                                  ERROR   +3              ;FPP ERROR
                                                                        ;INSTRUCTION WASNT SUPPOSE TO TRAP
102506  000167  000020                          JMP     2$                      ;REJOIN CODE
102512  012604                          51$:    MOV     (SP)+,R4                ;SEE IF PC = INSTRUCTION
102514  005726                                  TST     (SP)+                   ;CLEAN UP STACK
102516  022704  102442                          CMP     #1$,R4                  ;
102522  001403                                  BEQ     2$                      ;BRANCH IF GOOD COMPARE
102524  004737  140132                          CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
102530  104003                                  ERROR   +3              ;FPP ERROR
                                                                        ;PC WAS INCORRECT
                                        ;
                                        ;COMMON CODE FOR TRAP AND NO TRAP
                                        ;VERIFY STATUS
                                        ;
102532  170203                          2$:     STFPS   R3                      ;SAVE FPS
102534  012700  000200                          MOV     #200,R0                 ;SETUP FPS
102540  170100                                  LDFPS   R0                      ;FPS=200
102542  174011                                  STD     ACO,(R1)                ;GET RESULT
102544  016200  000024                          MOV     24(R2),R0                       ;GET EXPECTED STATUS
```

N6

FLOATING POINT TESTS

```
16004 102550  020003                        CMP     R0,R3                   ;VERIFY STATUS
16005 102552  001403                        BEQ     3$                      ;BRANCH IF GOOD
16006 102554  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
16007 102560  104003                        ERROR   +3              ;FPP ERROR
16008                                                                 ;BAD FPS
16009 102562  010204                3$:     MOV     R2,R4                   ;POINT TO EXPECTED DATA
16010 102564  062704  000012                ADD     #12,R4
16011 102570  004767  035304          4$:   JSR     R7,DATVER                       ;VERIFY DATA
16012 102574  005767  100320                TST     COUNT
16013 102600  001403                        BEQ     5$                      ;BRANCH IF GOOD
16014 102602  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
16015 102606  104003                        ERROR   +3              ;FPP ERROR
16016                                                                 ;BAD ACO
16017 102610  005737  003030          5$:   TST     @#FLAG                  ;SEE IF NEED TO CHECK FEC
16018 102614  001002                        BNE     7$                      ;BRANCH IF NEED TO CHECK
16019 102616  000162  000026                JMP     26(R2)                  ;RETURN FROM TEST
16020                               ;VERIFY FEC
16021 102622  012704  003122          7$:   MOV     #RECFEC,R4              ;POINT TO FEC AREA
16022 102626  170314                        STST    (R4)                    ;SAVE FEC
16023 102630  021462  000026                CMP     (R4),26(R2)                     ;VERIFY FEC FOR OVERFLOW
16024 102634  001403                        BEQ     8$                      ;BRANCH IF GOOD
16025 102636  004737  140132                CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
16026 102642  104003                        ERROR   +3              ;FPP ERROR
16027                                                                 ;BAD FEC
16028 102644  000162  000030          8$:   JMP     30(R2)          ;RETURN FROM TEST
16029                               ;
16030 102650                        HOP20:
16033                               ;
16034                               ;---------------------------------
16035                               ;TEST STCDI, STCDL
16036                               ;
16037 102650                        MSCD:
16038                               ;
16039                               ;1/ACO=0, INT
16040 102650  005037  003030                CLR     @#FLAG          ;NO INTERRUPTS
16041 102654  004767  000610                JSR     R7,SCDSUB                       ;DO TEST
16042 102660  000177  000000  000000        .WORD   0177,0,0,0                      ;ACO
      102666  000000
16043 102670  000000  177777                .WORD   0,-1            ;RESULT
16044 102674  007640                        .WORD   7640            ; TEST FPS
16045 102676  007644                        .WORD   7644            ;RESULT FPS
16046                               ;2/ACO=-0, LONG
16047 102700  005037  003030                CLR     @#FLAG          ;INTERRUPT
16048 102704  004767  000560                JSR     R7,SCDSUB                       ;DO TEST
16049 102710  100177  177777  177777        .WORD   100177,-1,-1,-1                 ;ACO
      102716  177777
16050 102720  000000  000000                .WORD   0,0             ;RESULT
16051 102724  007700                        .WORD   7700            ; TEST FPS
16052 102726  007704                        .WORD   7704            ;RESULT FPS
16053                               ;3/EXP=100, LONG
16054 102730  005037  003030                CLR     @#FLAG                  ;NO INTERRUPT
16055 102734  004767  000530                JSR     R7,SCDSUB                       ;DO TEST
16056 102740  020000  000000  000000        .WORD   20000,0,0,0                     ;ACO
      102746  000000
16057 102750  000000  000000                .WORD   0,0             ;RESULT
16058 102754  000300                        .WORD   300             ; TEST FPS
16059 102756  000304                        .WORD   304             ;RESULT FPS
```

FLOATING POINT TESTS

```
16060                                   ;4/EXP=200, BAISED 0, INT, ROUND
16061 102760  005037  003030                    CLR     @#FLAG          ;INTERRUPT
16062 102764  004767  000500                    JSR     R7,SCDSUB               ;DO TEST
16063 102770  140177  177777  000001            .WORD   140177,177777,1,1               ;AC0
      102776  000001
16064 103000  000000  000000                    .WORD   0,0             ;RESULT
16065 103004  007700                            .WORD   7700            ; TEST FPS
16066 103006  007704                            .WORD   7704            ;RESULT FPS
16067                                   ;5/LONG
16068 103010  005037  003030                    CLR     @#FLAG          ;INTERRUPT
16069 103014  004767  000450                    JSR     R7,SCDSUB               ;DO TEST
16070 103020  047667  075757  157737            .WORD   47667,75757,157737,167773               ;AC0
      103026  167773
16071 103030  055675  173757                    .WORD   55675,173757            ;RESULT
16072 103034  007717                            .WORD   7717            ; TEST FPS
16073 103036  007700                            .WORD   7700            ;RESULT FPS
16074                                   ;6/LONG, EXP=2**32
16075 103040  005037  003030                    CLR     @#FLAG                  ;NO INTERRUPT
16076 103044  004767  000420                    JSR     R7,SCDSUB               ;DO TEST
16077 103050  046400  000000  000000            .WORD   46400,0,0,0             ;AC0
      103056  000000
16078 103060  001000  000000                    .WORD   1000,0          ;RESULT
16079 103064  007700                            .WORD   7700            ; TEST FPS
16080 103066  007700                            .WORD   7700            ;RESULT FPS
16081                                   ;7/LONG, EXP>2**32
16082 103070  012737  000001  003030            MOV     #1,@#FLAG               ;INTERRUPT
16083 103076  004767  000366                    JSR     R7,SCDSUB               ;DO TEST
16084 103102  077607  000000  000000            .WORD   77607,0,0,0             ;AC0
      103110  000000
16085 103112  000000  000000                    .WORD   0,0             ;RESULT
16086 103116  007700                            .WORD   7700            ; TEST FPS
16087 103120  107705                            .WORD   107705          ;RESULT FPS
16088                                   ;8/INT, EXP=2**15
16089 103122  005037  003030                    CLR     @#FLAG          ;NO INTERRUPTS
16090 103126  004767  000336                    JSR     R7,SCDSUB               ;DO TEST
16091 103132  043200  000000  000000            .WORD   43200,0,0,0             ;AC0
      103140  000000
16092 103142  010000  177777                    .WORD   10000,-1                ;RESULT
16093 103146  007600                            .WORD   7600            ; TEST FPS
16094 103150  007600                            .WORD   7600            ;RESULT FPS
16095                                   ;9/INT, EXP>2**15
16096 103152  012737  000001  003030            MOV     #1,@#FLAG               ;INTERRUPT
16097 103160  004767  000304                    JSR     R7,SCDSUB               ;DO TEST
16098 103164  077777  177777  177777            .WORD   77777,-1,-1,-1          ;AC0
      103172  177777
16099 103174  000000  177777                    .WORD   0,-1            ;RESULT
16100 103200  007600                            .WORD   7600            ; TEST FPS
16101 103202  107605                            .WORD   107605          ;RESULT FPS
16102                                   ;10/INT, EXP>2**15, FID
16103 103204  012737  000000  003030            MOV     #0,@#FLAG               ;NO INTERRUPT
16104 103212  004767  000252                    JSR     R7,SCDSUB               ;DO TEST
16105 103216  043300  000000  000000            .WORD   43300,0,0,0             ;AC0
      103224  000000
16106 103226  000000  014000                    .WORD   0,14000         ;RESULT
16107 103232  047700                            .WORD   47700           ; TEST FPS
16108 103234  047700                            .WORD   47700           ;RESULT FPS
16109                                   ;11/INT, EXP>2**15, FIC=0
```

```
16110 103236  012737  000000  003030           MOV     #0,@#FLAG                ;NO INTERRUPT
16111 103244  004767  000220                   JSR     R7,SCDSUB                ;DO TEST
16112 103250  143300  177777  177777           .WORD   143300, 1,-1, 1          ;ACO
      103256  177777
16113 103260  177777  163741                   .WORD   -1,163741                ;RESULT
16114 103264  007300                           .WORD   7300             ; TEST FPS
16115 103266  007310                           .WORD   7310             ;RESULT FPS
16116                                   ;12/LONG, EXP>2**32, FID
16117 103270  012737  000002  003030           MOV     #2,@#FLAG                ;INTERRUPT
16118 103276  004767  000166                   JSR     R7,SCDSUB                ;DO TEST
16119 103302  050100  000000  000000           .WORD   50100,0,0,0              ;ACO
      103310  000000
16120 103312  000000  000000                   .WORD   0,0              ;RESULT
16121 103316  047700                           .WORD   47700            ; TEST FPS
16122 103320  147705                           .WORD   147705           ;RESULT FPS
16123                                   ;13/LONG, EXP>2**32, FIC=0
16124 103322  012737  000000  003030           MOV     #0,@#FLAG                ;NO INTERRUPT
16125 103330  004767  000134                   JSR     R7,SCDSUB                ;DO TEST
16126 103334  050377  177777  177777           .WORD   50377,-1, 1,-1           ;ACO
      103342  177777
16127 103344  000000  000000                   .WORD   0,0              ;RESULT
16128 103350  007300                           .WORD   7300             ; TEST FPS
16129 103352  007305                           .WORD   7305             ;RESULT FPS
16130                                   ;14/LONG, EXP<0
16131 103354  005037  003030                   CLR     @#FLAG           ;NO INTERRUPTS
16132 103360  004767  000104                   JSR     R7,SCDSUB                ;DO TEST
16133 103364  100200  177777  177777           .WORD   100200, 1, 1,-1          ;ACO
      103372  177777
16134 103374  000000  000000                   .WORD   0,0              ;RESULT
16135 103400  007757                           .WORD   7757             ; TEST FPS
16136 103402  007744                           .WORD   7744             ;RESULT FPS
16137                                   ;15/INT, EXP<0
16138 103404  005037  003030                   CLR     @#FLAG           ;NO INTERRUPTS
16139 103410  004767  000054                   JSR     R7,SCDSUB                ;DO TEST
16140 103414  037700  177777  177777           .WORD   37700,-1,-1,-2           ;ACO
      103422  177776
16141 103424  000000  177777                   .WORD   0,-1             ;RESULT
16142 103430  007600                           .WORD   7600             ; TEST FPS
16143 103432  007604                           .WORD   7604             ;RESULT FPS
16144                                   ;16/INT, EXP=10
16145 103434  005037  003030                   CLR     @#FLAG           ;NO INTERRUPTS
16146 103440  004767  000024                   JSR     R7,SCDSUB                ;DO TEST
16147 103444  004377  177777  177777           .WORD   4377,-1,-1,-1            ;ACO
      103452  177777
16148 103454  000000  177777                   .WORD   0,-1             ;RESULT
16149 103460  007600                           .WORD   7600             ; TEST FPS
16150 103462  007604                           .WORD   7604             ;RESULT FPS
16151                                   ;
16152 103464  000167  000244                   JMP     HOP21            ;GET OVER SUBROUTINE
16153                                   ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
16154                                   ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
16155                                   ;STCDI, STCDL, STCFI, STCFL
16156                                   ;
16157                                   ;                       ACO
16158                                   ;                       RESULT
16159                                   ;                       FPS BEFORE EXECUTION
16160                                   ;                       FPS AFTER EXECUTION
```

```
                                          ;               (FEC)
                                          ;
                                          ;TRAP ON CONVERSION FAILURE
                                          ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*x
                                          ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
                                          ;
103470  012602                            SCDSUB: MOV     (SP)+,R2                ; RETURN ADDRESS TO USE AS POINTER
103472  012737  103564  000244                    MOV     #50$,@#FPVEC            ;REDIRECT TRAP VECTOR
103500  012701  003144                            MOV     #RECDST+2,R1            ;POINT TO RESULT AREA
103504  012711  177777                            MOV     # 1,(R1)                ;PRELOAD RECEIVE DATA BUFFER
103510  012741  177777                            MOV     #-1,-(R1)               ;
103514  012700  000200                            MOV     #200,R0                 ;SET FPS TO DOUBLE
103520  170100                                    LDFPS   R0                      ;
103522  010204                                    MOV     R2,R4                   ;POINT TO AC0 DATA
103524  172414                                    LDD     (R4),AC0                        ;LOAD AC0
103526  016200  000014                            MOV     14(R2),R0               ;GET TEST FPS
103532  170100                                    LDFPS   R0                      ;LOAD TEST FPS
                                          ;
103534  175411                            40$:    STCDI   AC0,(R1)                ;*TEST INSTRUCTION(ACCORDING TO MODE)
103536  170327                            1$:     STST    (PC)+                   ;WAIT FOR POSSIBLE FPA TRAP.
103540  000000                                    .WORD   0                       ;STORE STATUS HERE.
                                          ;
                                          ;INSTRUCTION DIDNT TRAP
                                          ;
103542  032737  000001  003030                    BIT     #1,@#FLAG                       ;VERIFY A NO TRAP CONDITION
103550  001426                                    BEQ     2$                      ;BRANCH IF GOOD
103552  004737  140132                            CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
103556  104003                                    ERROR   +3              ;FPP ERROR
                                                                          ;INSTRUCTION SHOULD HAVE TRAPPED
103560  000167  000042                            JMP     2$                      ;REJOIN CODE
                                          ;
                                          ;INSTRUCTION TRAPPED
                                          ;
103564  032737  000001  003030            50$:    BIT     #1,@#FLAG               ;SEE IF EXPECTING A TRAP
103572  001005                                    BNE     51$                     ;BRANCH IF EXPECTING A TRAP
103574  004737  140132                            CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
103600  104003                                    ERROR   +3              ;FPP ERROR
                                                                          ;INSTRUCTION WASNT SUPPOSE TO TRAP
103602  000167  000020                            JMP     2$                      ;REJOIN CODE
103606  012604                            51$:    MOV     (SP)+,R4                ;SEE IF PC = INSTRUCTION
103610  005726                                    TST     (SP)+                   ;CLEAN UP STACK
103612  022704  103536                            CMP     #1$,R4                  ;
103616  001403                                    BEQ     2$                      ;BRANCH IF GOOD COMPARE
103620  004737  140132                            CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
103624  104003                                    ERROR   +3              ;FPP ERROR
                                                                          ;PC WAS INCORRECT
                                          ;
                                          ;COMMON CODE FOR TRAP AND NO TRAP
                                          ;VERIFY STATUS
                                          ;
103626  170203                            2$:     STFPS   R3                      ;SAVE FPS
103630  016200  000016                            MOV     16(R2),R0                       ;GET EXPECTED STATUS
103634  020003                                    CMP     R0,R3                   ;VERIFY STATUS
103636  001403                                    BEQ     3$                      ;BRANCH IF GOOD
103640  004737  140132                            CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
103644  104003                                    ERROR   +3              ;FPP ERROR
                                                                          ;BAD FPS
```

FLOATING POINT TESTS

```
16218 103646  010204                  3$:     MOV     R2,R4                   ;POINT TO EXPECTED DATA
16219 103650  062704  000010                  ADD     #10,R4
16220 103654  004767  034202            4$:     JSR     R7,DATVFR                       ;VERIFY DATA
16221 103660  005767  077234                  TST     COUNT
16222 103664  001403                          BEQ     5$                      ;BRANCH IF GOOD
16223 103666  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
16224 103672  104003                          ERROR   +3              ;FPP ERROR
16225                                                                 ;BAD ACO
16226 103674  005737  003030            5$:     TST     @#FLAG                  ;SEE IF NEED TO CHECK FEC
16227 103700  001002                          BNE     7$                      ;BRANCH IF NEED TO CHECK
16228 103702  000162  000020                  JMP     20(R2)                  ;RETURN FROM TEST
16229                                 ;VERIFY FEC
16230 103706  012704  003122            7$:     MOV     #RECFEC,R4              ;POINT TO FEC AREA
16231 103712  170314                          STST    (R4)                    ;SAVE FEC
16232 103714  021427  000006                  CMP     (R4),#6                 ;VERIFY FEC FOR OVERFLOW
16233 103720  001403                          BEQ     8$                      ;BRANCH IF GOOD
16234 103722  004737  140132                  CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
16235 103726  104003                          ERROR   +3              ;FPP ERROR
16236                                                                 ;BAD FEC
16237 103730  000162  000020            8$:     JMP     20(R2)          ;RETURN FROM TEST
16238                                 ;
16239 103734                          HOP21:
16240                                 ;
16243                                 ;
16244                                 ;-------------------------- -----
16245                                 ;TEST STCFI, STCFL
16246                                 ;
16247 103734                          MSCF:
16248                                 ;
16249                                 ;1/LONG EXP =30
16250 103734  005037  003030                  CLR     @#FLAG          ;NO INTERRUPTS
16251 103740  004767  177524                  JSR     R7,SCDSUB                       ;DO TEST
16252 103744  044541  052525  177777          .WORD   44541,52525,-1, 1                       ;ACO
      103752  177777
16253 103754  000003  102525                  .WORD   3,102525                        ;RESULT
16254 103760  007517                          .WORD   7517            ; TEST FPS
16255 103762  007500                          .WORD   7500            ;RESULT FPS
16256                                 ;2/INT, EXP<0
16257 103764  005037  003030                  CLR     @#FLAG          ;NO INTERRUPTS
16258 103770  004767  177474                  JSR     R7,SCDSUB                       ;DO TEST
16259 103774  002300  177777  177777          .WORD   2300,-1,-1, 1                   ;ACO
      104002  177777
16260 104004  000000  177777                  .WORD   0,-1            ;RESULT
16261 104010  007400                          .WORD   7400            ; TEST FPS
16262 104012  007404                          .WORD   7404            ;RESULT FPS
16263                                 ;3/LONG, EXP
16264 104014  012737  000001  003030          MOV     #1,@#FLAG                       ;INTERRUPT
16265 104022  004767  177442                  JSR     R7,SCDSUB                       ;DO TEST
16266 104026  070000  177777  177777          .WORD   70000,-1,-1,-1                  ;ACO
      104034  177777
16267 104036  000000  000000                  .WORD   0,0             ;RESULT
16268 104042  007540                          .WORD   7540            ; TEST FPS
16269 104044  107545                          .WORD   107545          ;RESULT FPS
16270                                 ;4/INT,EXP=5, FIC=0, FID=1
16271 104046  005037  003030                  CLR     @#FLAG          ;NO INTERRUPTS
16272 104052  004767  177412                  JSR     R7,SCDSUB                       ;DO TEST
16273 104056  052000  000000  177777          .WORD   52000,0,-1, 1                   ;ACO
```

```
104064  177777
104066  000000  177777                  .WORD   0, 1            ;RESULT
104072  047000                          .WORD   47000           ; TEST FPS
104074  047005                          .WORD   47005           ;RESULT FPS
                                ;
                                ;
                                ;---    -----   ----    -----   --
                                ;TEST STEXP
                                ;
104076                          MSXP:
                                ;
                                ;1/EXP=100
104076  004767  000154                  JSR     R7,SXPSUB               ;DO TEST
104102  020000  000000  000000          .WORD   20000,0,0,0             ;ACO
104110  000000
104112  177700                          .WORD   -100            ;RESULT
104114  007740                          .WORD   7740            ; TEST FPS
104116  007750                          .WORD   7750            ;RESULT FPS
                                ;2/EXP=201 FLOAT, NEG
104120  004767  000132                  JSR     R7,SXPSUB               ;DO TEST
104124  140377  177777  177777          .WORD   140377,-1,-1,0          ;ACO
104132  000000
104134  000001                          .WORD   1               ;RESULT
104136  007500                          .WORD   7500            ; TEST FPS
104140  007500                          .WORD   7500            ;RESULT FPS
                                ;3/EXP=-177
104142  004767  000110                  JSR     R7,SXPSUB               ;DO TEST
104146  000177  177777  177777          .WORD   177,-1, 1,-1            ;ACO
104154  177777
104156  177600                          .WORD   177600          ;RESULT
104160  007700                          .WORD   7700            ; TEST FPS
104162  007710                          .WORD   7710            ;RESULT FPS
                                ;4/EXP=-100
104164  004767  000066                  JSR     R7,SXPSUB               ;DO TEST
104170  020000  000000  177777          .WORD   20000,0,-1,-1           ;ACO
104176  177777
104200  177700                          .WORD   -100            ;RESULT
104202  040200                          .WORD   40200           ; TEST FPS
104204  040210                          .WORD   40210           ;RESULT FPS
                                ;5/EXP=200
104206  004767  000044                  JSR     R7,SXPSUB               ;DO TEST
104212  040000  000000  000000          .WORD   40000,0,0,0             ;ACO
104220  000000
104222  000000                          .WORD   0               ;RESULT
104224  007700                          .WORD   7700            ; TEST FPS
104226  007704                          .WORD   7704    ;RESULT FPS
                                ;6/EXP=0
104230  004767  000022                  JSR     R7,SXPSUB               ;DO TEST
104234  000177  177777  177777          .WORD   177,-1,-1,-1            ;ACO
104242  177777
104244  177600                          .WORD   177600          ;RESULT
104246  000000                          .WORD   0               ; TEST FPS
104250  000010                          .WORD   10              ;RESULT FPS
                                ;
104252  000167  000120                  JMP     HOP22           ;GET OVER SUBROUTINE
                                ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
                                ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
```

```
16326                                   ;STEXP
16327                                   ;               AC0
16328                                   ;               EXPONENT RESULT
16329                                   ;               FPS BEFORE EXECUTION
16330                                   ;               FPS AFTER EXECUTION
16331                                   ;
16332                                   ;NO TRAPS CAN OCCUR
16333                                   ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
16334                                   ;*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X*X
16335                                   ;
16336 104256  012602                   SXPSUB: MOV     (SP)+,R2                ; RETURN ADDRESS TO USE AS POINTER
16337 104260  012737  104360  000244           MOV     #50$,@#FPVEC            ;REDIRECT TRAP VECTOR
16338 104266  012701  003142                   MOV     #RECDST,R1              ;POINT TO RESULT AREA
16339 104272  012700  000200                   MOV     #200,R0                 ;SET FPS TO DOUBLE
16340 104276  170100                           LDFPS   R0                      ;
16341 104300  010204                           MOV     R2,R4                   ;POINT TO AC0 DATA
16342 104302  172414                           LDD     (R4),AC0                        ;LOAD AC0
16343 104304  016200  000012                   MOV     12(R2),R0               ;GET TEST FPS
16344 104310  170100                           LDFPS   R0                      ;LOAD TEST FPS
16345                                   ;
16346 104312  175011                   40$:    STEXP   AC0,(R1)                ;*TEST INSTRUCTION(ACCORDING TO MODE)
16347                                   ;
16348                                   ;VERIFY STATUS
16349                                   ;
16350 104314  170203                   2$:     STFPS   R3                      ;SAVE FPS
16351 104316  016200  000014                   MOV     14(R2),R0                       ;GET EXPECTED STATUS
16352 104322  020003                           CMP     R0,R3                   ;VERIFY STATUS
16353 104324  001403                           BEQ     3$                      ;BRANCH IF GOOD
16354 104326  004737  140132                   CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
16355 104332  104003                           ERROR   +3              ;FPP ERROR
16356                                                                  ;BAD FPS
16357 104334  016204  000010           3$:     MOV     10(R2),R4                       ;POINT TO EXPECTED EXPONENT
16358 104340  020437  003142                   CMP     R4,@#RECDST                     ;VERIFY EXPONENT
16359 104344  001403                           BEQ     5$                      ;BRANCH IF GOOD
16360 104346  004737  140132                   CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
16361 104352  104003                           ERROR   +3              ;FPP ERROR
16362                                                                  ;BAD AC0
16363 104354  000162  000016           5$:     JMP     16(R2)                  ;RETURN FROM TEST
16364                                   ;
16365                                   ;INSTRUCTION TRAPPED
16366                                   ;
16367 104360  012600                   50$:    MOV     (SP)+,R0                ;SAVE PC
16368 104362  012605                           MOV     (SP)+,R5                ;SAVE OLD PS
16369 104364  004737  140132                   CALL    @#DETFPA                ;DETERMINE FLOATING POINT FAULT. $$$
16370 104370  104003                           ERROR   +3              ;FPP ERROR
16371                                                                  ;WILD TRAP DURING STEXP
16372 104372  000167  177756                   JMP     5$                      ;REJOIN CODE
16373                                   ;
16374 104376                           HOP22:
16377
16378                                   .ENABL AMA
16379                                   .SBTTL  TEST  - CHECK REGISTER ACCESS
16380                                   ;CHECK REGISTER ACCESS - THIS TEST WILL VERIFY EACH OF THE THREE
16381                                   ;CACHE MEMORY SYSTEM REGISTERS CAN BE ACCESSED WITHOUT A TRAP TO
16382                                   ;LOCATION 4(NON-EXISTANT ADDRESS TRAP) OCCURRING. THE REGISTERS TO
16383                                   ;BE TESTED ARE:
16384                                   ;       CACHE CONTROL           17777746
```

```
16385                                   ;       MEMORY SYSTEM ERROR     17777744
16386                                   ;       HIT/MISS                17777752
16387                                   ;EACH REGISTER WILL BE ACCESSED WITH A WRITE CYCLE.
16388                                   ;
16389                                   ;BGNTST
16390                                   ;SAVE CONTENT OF LOCATION 4
16391                                   ;LET LOCATION 4 POINT TO ERROR ROUTINE
16392                                   ;INITIALIZE R TO TOP OF ADDRESS TABLE
16393                                   ;DO UNTIL FOR ALL REGISTERS
16394                                   ;.      TEST REGISTER UNDER TEST
16395                                   ;.      IF TIMEOUT DID HAPPEN
16396                                   ;.              ERROR
16397                                   ;.      ENDIF
16398                                   ;ENDDO
16399                                   ;RESTORE CONTENTS OF LOCATION 4
16400                                   ;EXIT   TST
16401                                   ;
16402                                   ;ADDRESS TABLE: 17777746
16403                                   ;               17777744
16404                                   ;               17777752
16405                                   ;
16406                                   ;ENDTST
16407                                   ;ERROR ROUTINE: SET TIMEOUT FLAG
16408                                   ;               RETURN
16409
16410                                   ;;*****************************************************************
     104376  000004                     TST2:   SCOPE
16411 104400  000240                             NOP
16412 104402  005737  003032                     TST     CCHPAS                  ;have done enough inclusive passes?
16413 104406  001002                             BNE     99$                     ; not yet
16414
16415 104410  000240                             NOP                             ; debug aid
16416 104412  000453                             BR      TST3            ;;GO TO NEXT TEST
16417 104414  000240                     99$:    NOP
16418 104416  012737  137542  000004             MOV     #TOUT,  @#4
16419 104424  013737  000004  003012             MOV     @#4,    SLOCOO          ;SAVE CONTENTS OF VECTOR 4
16420 104432  012737  104536  000004             MOV     #ERROUT,@#4             ;LET VECTOR 4 POINT TO ERROR ROUTINE
16421 104440  005001                             CLR     R1                      ;CLEAR TIMEOUT FLAG
16422 104442  005737  177746             1$:     TST     CCR                     ;READ REGISTER UNDER TEST
16423 104446  005701                             TST     R1                      ;TIMEOUT?
16424 104450  001404                             BEQ     2$
16425 104452  012737  177746  001122             MOV     #CCR,$BDADR             ;STORE LOCATION
16426 104460  104130                             ERROR   +130                    ;TIMEOUT ACCESING CCR
16427 104462  005001                     2$:     CLR     R1                      ;CLEAR TIMOUT FLAG
16428 104464  005737  177744                     TST     MSER                    ;READ MSER
16429 104470  005701                             TST     R1                      ;TIMEOUT ?
16430 104472  001404                             BEQ     3$
16431 104474  012737  177744  001122             MOV     #MSER,$BDADR            ;STORE LOCATION
16432 104502  104130                             ERROR   +130                    ;TIMEOUT ACCESSING MSER
16433 104504  005001                     3$:     CLR     R1                      ;CLEAR TIMEOUT FLAG
16434 104506  005737  177752                     TST     HITMIS                  ;ACCESS HIT/MISS
16435 104512  005701                             TST     R1                      ;TIMEOUT?
16436 104514  001404                             BEQ     4$
16437 104516  012737  177752  001122             MOV     #HITMIS,$BDADR          ;STORE LOCATION
16438 104524  104130                             ERROR   +130                    ;TIMEOUT ACCESSING HIT/MISS
16439 104526  013737  003012  000004     4$:     MOV     SLOCOO, @#4             ;RESTORE VECTOR 4
16440 104534  000402                             BR      TST3            ;;GO TO NEXT TEST
```

I7

TEST - CHECK REGISTER ACCESS

```
16441
16442 104536                        ERROUT:
16443 104536  005201                        INC     R1                      ;FLAG TIMEOUT
16444 104540  000002                        RTI
16445
```

J7

TEST - CCR REGISTER BIT TEST

```
                                .SBTTL  TEST  - CCR REGISTER BIT TEST
                                ;CCR REGISTER BIT TEST - THIS TEST WILL VERIFY THAT EACH READ/WRITE BIT OF
                                ;THE CACHE CONTROL REGISTER CAN BE SET AND CLEARED INDIVIDUALLY AND THAT
                                ;BITS 15   11 AND BIT 8 ARE ALWAYS READ AS ZEROS.
                                ;
                                ;BGNTST
                                ;INITIALIZE GOOD DATA TO #1
                                ;CLEAR CCR
                                ;DO UNTIL ALL BITS TESTED
                                ;.      WRITE GOOD DATA TO CCR
                                ;.      READ CCR
                                ;.      IF CCR NOT EQUAL TO GOOD DATA THEN
                                ;.      .       IF CCR EQUAL TO ZERO THEN
                                ;.      .       .       IF GOOD DATA NOT EQUAL TO BIT 15-11 OR BIT 8 THEN
                                ;.      .       .       .       ERROR IN READ/WRITE BITS OF CCR
                                ;.      .       .       ENDIF
                                ;.      ENDIF
                                ;.      UPDATE TO NEXT BIT
                                ;ENDDO
                                ;ENDTST

                                ;;*****************************************************************
    104542  000004              TST3:   SCOPE
16469 104544  000240                    NOP
16470 104546  005737  003032            TST     CCHPAS                  ;have done enough inclusive passes?
16471 104552  001002                    BNE     99$                     ; not yet
16472 104554  000240                    NOP                             ; debug aid
16473 104556  000431                    BR      TST4            ;;GO TO NEXT TEST
16474 104560  000240              99$:  NOP
16475 104562  012701  000001            MOV     #1,     R1              ;INITIALIZE GOOD DATA TO #1
16476 104566  005037  177746            CLR     CCR                     ;CLEAR CCR
16477 104572  042737  001000  177520    BIC     #1000,BCSR              ;DISABLE HALT ON BREAK
16478 104600  010137  177746      1$:   MOV     R1,     CCR             ;WRITE GOOD DATA TO CCR
16479 104604  023701  177746            CMP     CCR,    R1              ;IF CCR NOT EQUAL GOOD DATA
16480 104610  001407                    BEQ     3$                      ;THEN
16481 104612  005737  177746            TST     CCR                     ;IF CCR EQUAL TO ZERO
16482 104616  001003                    BNE     2$                      ;THEN
16483 104620  032701  174400            BIT     #174400,R1              ;IF GOOD DATA NE TO BIT 15 11 OR BIT 8
16484 104624  001001                    BNE     3$                      ;THEN
16485 104626  104004              2$:   ERROR   .4                      ;ERROR IN READ/WRITE BITS OF CCR
16486 104630  006101              3$:   ROL     R1                      ;UPDATE TO NEXT BIT
16487 104632  103362                    BCC     1$                      ;DO UNTIL ALL BITS TESTED
16488 104634  052737  001000  177520    BIS     #1000,BCSR              ;ENABLE HALT ON BREAK
16489
16490
```

TEST - FORCE MISS TEST

```
16492                           .SBTTL  TEST  - FORCE MISS TEST
16493                           ;FORCE MISS TEST - THIS TEST WILL VERIFY THAT ALL REFERENCES MADE
16494                           ;WITH EITHER BIT<3> OR BIT<2> OF THE CCR SET CAUSE A CACHE MISS AND
16495                           ;LEAVE THE CACHE ENTRY UNCHANGED. FIRST WRITE A TEST ADDRESS WITH
16496                           ;BITS<3:2> CLEARED TO ALLOCATE CACHE AND SET A KNOWN DATA PATTERN
16497                           ;INTO THE CACHE. THEN SET BIT<2> AND REWRITE THE SAME ADDRESS WITH
16498                           ;NEW DATA. NEXT CLEAR BITS<3:2> AND READ THE TEST ADDRESS, THE DATA
16499                           ;SHOULD EQUAL TO PATTERN 1. FINALLY READ THE TEST ADDRESS WITH BIT<3>
16500                           ;SET, THE DATA SHOULD EQUAL PATTERN 2. A LAST WRITE MUST BE DONE WITH
16501                           ;BOTH FORCE BITS CLEARED BECAUSE THE CACHE AND MAIN MEMORY HAVE
16502                           ;DIFFERENT DATA.
16503                           ;
16504                           ;
16505                           ;BGNTST
16506                           ;WRITE TEST ADDRESS WITH PATTERN 1
16507                           ;SET CCR BITS<3:2> = 0,1
16508                           ;WRITE TEST ADDRESS WITH PATTERN 2
16509                           ;CLEAR CCR BIT<3>
16510                           ;READ TEST ADDRESS
16511                           ;SAVE HIT/MISS REGISTER DATA
16512                           ;COMPARE RECEIVED DATA TO PATTERN 1
16513                           ;IF DATA NOT EQUAL THEN
16514                           ;.      IF DATA EQUAL TO PATTERN 2 THEN
16515                           ;.      .       IF HIT THEN
16516                           ;.      .       .       ERROR FORCE MISS WRITES TO CACHE
16517                           ;.      .       ELSE
16518                           ;.      .       .       IF PARITY ERROR INDICATORS EQUAL 0 THEN
16519                           ;.      .       .       .       ERROR FROCE MISS INVALIDATES CACHE
16520                           ;.      .       .       ELSE
16521                           ;.      .       .       .       IF ALL PARITY INDICATORS SET THEN
16522                           ;.      .       .       .       .       ERROR PARITY ARORT AFTER FORCE MISS
16523                           ;.      .       .       .       ENDIF
16524                           ;.      .       .       .       IF TAG PARITY ERROR THEN
16525                           ;.      .       .       .       .       ERROR  TAG PARITY ERROR AFTER FORCE MISS
16526                           ;.      .       .       .       ELSE
16527                           ;.      .       .       .       .       IF B0 AND  B1 ERROR THEN
16528                           ;.      .       .       .       .       .       ERROR DATA PARITY ERROR AFTER FORCE MISS
16529                           ;.      .       .       .       .       ENDIF
16530                           ;.      .       .       .       .       IF B0 PARITY ERROR THEN
16531                           ;.      .       .       .       .       .       ERROR LOW BYTE PARITY ERROR
16532                           ;.      .       .       .       .       ENDIF
16533                           ;.      .       .       .       .       IF B1 PARITY ERROR THEN
16534                           ;.      .       .       .       .       .       ERROR HIGH BYTE PARITY ERROR
16535                           ;.      .       .       .       .       ENDIF
16536                           ;.      .       .       .       ENDIF
16537                           ;.      .       .       ENDIF
16538                           ;.      .       ENDIF
16539                           ;.      ELSE
16540                           ;.      .       ERROR IN DATA PATH
16541                           ;.      ENDIF
16542                           ;ENDIF
16543                           ;SET CCR<3:2> = 1,0
16544                           ;READ TEST ADDRESS
16545                           ;SAVE HIT/MISS REGISTER DATA
16546                           ;COMPARE RECEIVED DATA TO PATTERN 2
16547                           ;IF DATA NOT EQUAL THEN
16548                           ;.      IF RECIEVED DATA EQUAL TO PATTERN 1 THEN
```

TEST - FORCE MISS TEST

```
16549                                   ;.      .       IF HIT THEN
16550                                   ;.      .               ERROR FORCE MISS READS FROM CACHE
16551                                   ;.      .       ELSE
16552                                   ;.      .               ERROR FORCE MISS READS FROM CACHE AND MISS
16553                                   ;.      .       ENDIF
16554                                   ;.      ELSE
16555                                   ;.              ERROR IN DATA PATH
16556                                   ;.      ENDIF
16557                                   ;ENDIF
16558                                   ;CLEAR CCR<3:2>
16559                                   ;WRITE TEST ADDRESS
16560                                   ;ENDTST
16561                                   ;
16562                                   ;;*****************************************************************
      104642  000004                  TST4:   SCOPE
16563 104644  000240                          NOP
16564 104646  005737  003032                  TST     CCHPAS                  ;have done enough inclusive passes?
16565 104652  001002                          BNE     99$                     ; not yet
16566 104654  000240                          NOP                             ; debug aid
16567 104656  000563                          BR      TST5            ;;GO TO NEXT TEST
16568 104660  000240                  99$:    NOP
16569 104662  013737  000114  003012        MOV     @#114,  SLOC00          ;SAVE CONTENTS OF VECTOR 114
16570 104670  012737  105214  000114        MOV     #FMPARR,@#114           ;SETUP PARITY VECTOR TO POINT TO HANDLER
16571 104676  005003                        CLR     R3                      ;CLEAR MSER SAVE LOCATION
16572 104700  005037  003162                CLR     @#TSTLOC                ;WRITE TEST LOCATION WITH PATTERN 1
16573 104704  012737  000004  177746        MOV     #BIT02, CCR             ;SET CCR BITS<3:2> = 0,1
16574 104712  012737  177777  003162        MOV     #177777,@#TSTLOC        ;WRITE TEST LOCATION WITH PATTERN 2
16575 104720  012737  000200  177746        MOV     #200,   CCR             ;CLEAR CCR BIT 3 AND SET PARITY ABORTS
16576 104726  012737  177777  001124        MOV     #177777,$GDDAT          ;SAVE DATA IN MEMORY
16577 104734  042737  001000  177520        BIC     #1000,BCSR              ;DISABLE HALT ON BREAK
16578 104742  013701  003162                MOV     @#TSTLOC,R1             ;READ TEST ADDRESS
16579 104746  013702  177752                MOV     @#HITMIS,R2             ;SAVE HIT/MISS DATA
16580 104752  052737  001000  177520        BIS     #1000,BCSR              ;ENABLE HALT ON BREAK
16581 104760  005701                        TST     R1                      ;COMPARE RECEIVED DATA TO PATTERN 1
16582 104762  001451                        BEQ     1$                      ;IF DATAS NOT EQUAL THEN
16583 104764  022701  177777                CMP     #177777,R1              ;IF DATA EQUAL TO PATTERN 2
16584 104770  001045                        BNE     106$                    ;THEN
16585 104772  032702  000002                BIT     #BIT01, R2              ;IF HIT
16586 104776  001402                        BEQ     100$                    ;THEN
16587 105000  104005                        ERROR   .5                      ;FORCE MISS WRITES TO CACHE
16588 105002  000441                        BR      1$                      ;ELSE
16589 105004  032703  100340          100$:   BIT     #100340,R3              ;IF PARITY INDICATORS EQUAL 0
16590 105010  001002                        BNE     101$                    ;THEN
16591 105012  104006                        ERROR   .6                      ;FORCE MISS WRITE INVALIDATES CACHE
16592 105014  000434                        BR      1$                      ;ELSE
16593 105016  022703  100340          101$:   CMP     #100340,R3              ;IF ALL PARITY INDICATORS SET
16594 105022  001002                        BNE     102$                    ;THEN
16595 105024  104007                        ERROR   .7                      ;PARITY ABORT AFTER FORCE MISS
16596 105026  000427                        BR      1$                      ;ELSE
16597 105030  032703  000040          102$:   BIT     #BIT05, R3              ;IF TAG PARITY ERROR
16598 105034  001402                        BEQ     103$                    ;THEN
16599 105036  104010                        ERROR   .10                     ;TAG PARITY ERROR AFTER FORCE MISS
16600 105040  000422                        BR      1$                      ;ELSE
16601 105042  010304                  103$:   MOV     R3,     R4              ;COPY MSER INTO REG 4
16602 105044  042704  177477                BIC     #177477,R4              ;MASK FOR B0 AND B1 DATA
16603 105050  022704  000300                CMP     #300,   R4              ;IF B0 AND B1 DATA
16604 105054  001002                        BNE     104$                    ;THEN
```

TEST - FORCE MISS TEST

```
16605 105056  104011                          ERROR   ·11                     ;DATA PARITY ERROR AFTER FORCE MISS
16606 105060  000412                          BR      1$                      ;ELSE
16607 105062  032703  000100          104$:   BIT     #100,   R3              ;IF B0 ERROR
16608 105066  001401                          BEQ     105$                    ;THEN
16609 105070  104012                          ERROR   ·12                     ;LOW BYTE PARITY ERROR AFTER FORCE MISS
16610 105072  032703  000200          105$:   BIT     #200,   R3              ;IF B1 ERROR
16611 105076  001403                          BEQ     1$                      ;THEN
16612 105100  104013                          ERROR   ·13                     ;HIGH BYTE PARITY ERROR AFTER FORCE MISS
16613 105102  000401                          BR      1$                      ;ENDIF
16614 105104  104014                  106$:   ERROR   ·14                     ;ERROR DATA PATH
16615 105106  005003                  1$:     CLR     R3                      ;CLEAR MSER SAVE REGISTER
16616 105110  052737  000010  177746          BIS     #BIT03, CCR             ;SET CCR BITS <3:2> = 1,0
16617 105116  042737  001000  177520          BIC     #1000,BCSR              ;DISABLE HALT ON BREAK
16618 105124  013701  003162                  MOV     @#TSTLOC,R1             ;READ TEST LOCATION
16619 105130  013702  177752                  MOV     @#HITMIS,R2             ;SAVE HIT/MISS REGISTER DATA
16620 105134  052737  001000  177520          BIS     #1000,BCSR              ;ENABLE HALT ON BREAK
16621 105142  020127  177777                  CMP     R1,     #177777         ;COMPARE RECEIVED DATA TO PATTERN 2
16622 105146  001412                          BEQ     2$                      ;IF DATAS NOT EQUAL THEN
16623 105150  005701                          TST     R1                      ;IF RECIEVED DATA EQUAL TO PATTERN 1
16624 105152  001007                          BNE     108$                    ;THEN
16625 105154  032702  000002                  BIT     #BIT01, R2              ;IF HIT
16626 105160  001402                          BEQ     107$                    ;THEN
16627 105162  104015                          ERROR   ·15                     ;FORCE MISS READS F     CACHE
16628 105164  000403                          BR      2$                      ;ELSE
16629 105166  104016                  107$:   ERROR   ·16                     ;FORCE MISS READS FROM CACHE AND MISS
16630 105170  000401                          BR      2$                      ;ENDIF
16631 105172  104014                  108$:   ERROR   ·14                     ;ERROR IN DATA PATH
16632 105174  013737  003012  000114  2$:     MOV     SLOC00, @#114           ;RESTORE VECTOR 114
16633 105202  005037  177746                  CLR     CCR                     ;CLEAR CCR BITS<3:2>
16634 105206  005037  003162                  CLR     @#TSTLOC                ;WRITE TEST ADDRESS
16635 105212  000405                          BR      TST5            ;;GO TO NEXT TEST
16636
16637
16638 105214  011637  001122          FMPARR: MOV     (SP),$BDADR             ;SAVE ADDRESS THAT CAUSED ABORT
16639 105220  013703  177744                  MOV     MSER,   R3              ;SAVE CONTENTS OF MEMORY SYSTEM ERROR
16640 105224  000002                          RTI                             ;REGISTER AND RETURN
16641
16642
16643                                 .SBTTL  TEST  - HIT/MISS REGISTER TEST PART 1
16644                                 ;HIT/MISS REGISTER TEST PART 1 - THIS TEST WILL VERIFY THAT THE HIT/MISS
16645                                 ;REGISTER CORRECTLY LOGS HITS AND MISSES.FIRST WRITE A TEST ADDRESS WITH
16646                                 ;BITS<3:2> CLEARED TO ALLOCATE CACHE AND SET A KNOWN DATA PATTERN INTO
16647                                 ;THE CACHE. THEN SET BIT<2> AND REWRITE THE SAME ADDRESS WITH NEW DATA.
16648                                 ;NEXT CLEAR CCR BITS<3:2> AND READ THE TEST ADDRESS, THE HIT/MISS REGISTER
16649                                 ;SHOULD LOGGED A HIT. FINALLY READ THE TEST ADDRESS PLUS 20000(8) THE
16650                                 ;HIT/MISS REGISTER SHOULD HAVE LOGGED A MISS.
16651                                 ;
16652                                 ;
16653                                 ;BGNTST
16654                                 ;WRITE TEST ADDRESS WITH PATTERN 1
16655                                 ;SET CCR BITS<3:2> = 0,1
16656                                 ;WRITE TEST ADDRESS WITH PATTERN 2
16657                                 ;CLEAR CCR BIT<3>
16658                                 ;READ TEST ADDRESS
16659                                 ;IF HIT/MISS REGISTER BIT 1 NOT SET THEN
16660                                 ;.      ERROR IN RECORDING HITS IN HIT/MISS
16661                                 ;ENDIF
```

TEST - HIT/MISS REGISTER TEST PART 1

```
16662                                   ;READ TEST ADDRESS + 20000(8)
16663                                   ;IF HIT/MISS REGISTER BIT 1 SET THEN
16664                                   ;.      ERROR IN RECORDING HITS IN HIT/MISS
16665                                   ;ENDIF
16666                                   ;ENDTST
16667                                   ;
16668                                   ;;*****************************************************************
     105226  000004                   TST5:   SCOPE
16669
16670 105230  000240                           NOP
16671 105232  005737  003032                   TST     CCHPAS                  ;have done enough inclusive passes?
16672 105236  001002                           BNE     99$                     ; not yet
16673 105240  000240                           NOP                             ; debug aid
16674 105242  000463                           BR      TST6            ;;GO TO NEXT TEST
16675 105244  000240                   99$:    NOP
16676 105246  013737  000114  003012           MOV     @#114,  SLOC00          ;SAVE CONTENTS OF VECTOR 114
16677 105254  012737  105404  000114           MOV     #HMPARR,@#114           ;SETUP PARITY VECTOR TO POINT TO HANDLER
16678 105262  005037  003162                   CLR     @#TSTLOC                ;WRITE TEST LOCATION WITH PATTERN 1
16679 105266  012737  000004  177746           MOV     #BIT02, CCR             ;SET CCR BITS<3:2> = 0,1
16680 105274  012737  177777  003162           MOV     #177777,@#TSTLOC        ;WRITE TEST LOCATION WITH PATTERN 2
16681 105302  012737  000200  177746           MOV     #200,   CCR             ;CLEAR CCR BIT 3 AND SET PARITY ABORTS
16682 105310  042737  001000  177520           BIC     #1000,BCSR              ;DISABLE HALT ON BREAK
16683 105316  013701  003162                   MOV     @#TSTLOC,R1             ;READ TEST ADDRESS
16684 105322  013737  177752  114150           MOV     HITMIS,RECDAT           ;STORE REGISTER
16685 105330  032737  000004  114150           BIT     #BIT02,RECDAT           ;IF HIT/MISS REGISTER BIT 1 NOT SET
16686 105336  001001                           BNE     1$                      ;THEN
16687 105340  104045                           ERROR   +45                     ;ERROR IN RECORDING HITS IN HIT/MISS
16688 105342  013701  023162           1$:     MOV     @#TSTLOC+8192.,R1       ;READ TEST LOCATION + 20000(8)
16689 105346  013737  177752  114150           MOV     HITMIS,RECDAT           ;STORE REGISTER
16690 105354  032737  000004  114150           BIT     #BIT02,RECDAT           ;IF HIT/MISS REGISTER BIT 1 SET
16691 105362  001401                           BEQ     2$                      ;THEN
16692 105364  104045                           ERROR   +45                     ;ERROR IN RECORDING HITS IN HIT/MISS
16693 105366  013737  003012  000114   2$:     MOV     SLOC00, @#114           ;RESTORE VECTOR 114
16694 105374  052737  001000  177520           BIS     #1000,BCSR              ;ENABLE HALT ON BREAK
16695 105402  000403                           BR      TST6            ;;GO TO NEXT TEST
16696
16697 105404  013703  177744           HMPARR: MOV     MSER,   R3              ;SAVE CONTENTS OF MEMORY SYSTEM ERROR
16698 105410  000002                           RTI                             ;REGISTER AND RETURN
16699
```

```
16701                                   .SBTTL  TEST  - HIT/MISS REGISTER TEST
16702                                   ;HIT/MISS REGISTER TEST   THIS TEST WILL VERIFY THAT THE HIT/MISS
16703                                   ;REGISTER CORRECTLY LOGS CACHE HITS AND MISSES. IT WILL ALSO VERIFY
16704                                   ;THAT EACH BIT OF THE REGISTER IS UNIQUE. THIS WILL BE DONE BY
16705                                   ;FLOATING A ZERO THROUGH A FIELD OF ONES. THE ROTATING WILL BE DONE
16706                                   ;BY EXECUTING A TST @#HM (WHERE HM IS THE ADDRESS OF THE HIT/MISS
16707                                   ;REGISTER) AT SUCCESSIVE POSITIONS IN A SET OF EIGHT NOPS. THE READ
16708                                   ;OF THE I/O PAGE ADDRESS OF THE HIT/MISS REGISTER SHOULD CAUSE A
16709                                   ;MISS TO BE RECORDED.
16710                                   ;
16711                                   ;BGNTST
16712                                   ;INITIALIZE LOOP INDICATOR
16713                                   ;INITIALIZE EXPECTED DATA
16714                                   ;DO UNTIL LOOP INDICATOR = SEVEN
16715                                   ;.      PUT BACKGROUND DATA INTO EXECUTION BUFFER
16716                                   ;.      PUT TST INSTRUCTION INTO TEST LOCATION
16717                                   ;.      JUMP TO EXECUTE BUFFER
16718                                   ;.      IF EXPECTED DATA NE RECEIVED DATA THEN
16719                                   ;.              ERROR IN RECORDING HITS IN HIT/MISS
16720                                   ;.      ENDIF
16721                                   ;.      INCREMENT LOOP INDICATOR
16722                                   ;.      UPDATE EXPECTED DATA
16723                                   ;ENDDO
16724                                   ;EXIT   TST
16725                                   ;
16726                                   ;BACKGROUND DATA:NOP
16727                                   ;               NOP
16728                                   ;               NOP
16729                                   ;               NOP
16730                                   ;               NOP
16731                                   ;               NOP
16732                                   ;               NOP
16733                                   ;               NOP
16734                                   ;               MOV     @#HM,R2
16735                                   ;               RTS     PC
16736                                   ;ENDTST
16737                                   ;;*****************************************************************
     105412  000004                     TST6:   SCOPE
16738 105414  000240                             NOP
16739 105416  005737  003032                     TST     CCHPAS                  ;have done enough inclusive passes?
16740 105422  001002                             BNE     99$                     ; not yet
16741 105424  000240                             NOP                             ; debug aid
16742 105426  000511                             BR      TST7            ;;GO TO NEXT TEST
16743 105430  000240                     99$:    NOP
16744 105432  005037  003154                     CLR     LOOPIN                  ;INITIALIZE LOOP INDICATOR
16745 105436  013737  023204  001162             MOV     @#TSTLOC+20022,$TMP1    ;SAVE FOR LATER
16746 105444  012703  003164                     MOV     #TSTLOC+2,R3            ;GET ADDRESS OF EXECUTION BUFFER
16747 105450  012704  105634                     MOV     #EXPTBL,R4              ;GET ADDRESS OF EXPECTED DATA TABLE
16748 105454  012705  177752                     MOV     #HITMIS,R5              ;PUT ADDRESS OF HIT/MISS REGISTER IN GPR
16749 105460  042737  001000  177520             BIC     #1000,BCSR              ;DISABLE HALT ON BREAK
16750 105466  023727  003154  000007  1$:       CMP     LOOPIN, #7              ;DO UNTIL LOOP INDICATOR EQUALS 7
16751 105474  001440                             BEQ     ENDHRT                  ;THEN EXIT
16752 105476  012702  000013                     MOV     #13,    R2
16753 105502  012700  105606                     MOV     #BACDAT,R0              ;PUT BACKGROUND DATA INTO
16754 105506  012701  003162                     MOV     #TSTLOC,R1              ;EXECUTION BUFFER
16755 105512  012021                     2$:     MOV     (R0)+,  (R1)+
16756 105514  077202                             SOB     R2,     2$              ;LOOP FOR TEN WORDS
```

C8

SEQ 0300

TEST HIT/MISS REGISTER TEST

```
16757 105516  023727  003154  000006          CMP     LOOPIN, #6              ;IS THIS LAST LOOP
16758 105524  001004                          BNE     3$                      ;IF YES THEN
16759 105526  013737  001162  023204          MOV     $TMP1,@#TSTLOC+20022    ;INVALIDATE FETCH OF "RECDAT"
16760 105534  000404                          BR      4$                      ;ELSE
16761 105536  012723  005737          3$:     MOV     #5737,  (R3)+           ;MOVE "TST" INSTRUCTION TO BUFFER
16762 105542  012713  177752                  MOV     #HITMIS,(R3)            ;MOVE HIT.MISS REG. ADD. TO BUFFER
16763 105546  004737  003162          4$:     JSR     PC,     TSTLOC          ;GO EXECUTE TEST CODE
16764 105552  021437  114150                  CMP     (R4),   RECDAT          ;IF EXPECTED DATA NOT EQUAL TO
16765 105556  001403                          BEQ     5$                      ;RECEIVED DATA THEN
16766 105560  011437  001124                  MOV     (R4),$GDDAT             ;SAVE EXPECTED PATTERN
16767 105564  104017                          ERROR   +17                     ;ERROR IN RECORDING HITS IN HIT/MISS
16768 105566  005724                  5$:     TST     (R4)+                   ;INCREMENT POINTER TO EXPECTED PATTERN
16769 105570  005237  003154                  INC     LOOPIN                  ;INCREMENT LOOP COUNTER
16770 105574  000734                          BR      1$
16771 105576                          ENDHRT:
16772 105576  052737  001000  177520          BIS     #1000,BCSR              ;ENABLE HALT ON BREAK
16773 105604  000422                          BR      TST7            ;;GO TO NEXT TEST
16774
16775 105606  000240                  BACDAT: .WORD   NOP
16776 105610  000240                          .WORD   NOP
16777 105612  000240                          .WORD   NOP
16778 105614  000240                          .WORD   NOP
16779 105616  000240                          .WORD   NOP
16780 105620  000240                          .WORD   NOP
16781 105622  000240                          .WORD   NOP
16782 105624  000240                          .WORD   NOP
16783 105626  011537  114150                  MOV     (R5),RECDAT             ;SAVE CONTENTS OF HIT/MISS REGISTER
16784 105632  000207                          RTS     PC
16785
16786 105634  000077                  EXPTBL: .WORD   77
16787 105636  000037                          .WORD   37
16788 105640  000057                          .WORD   57
16789 105642  000067                          .WORD   67
16790 105644  000073                          .WORD   73
16791 105646  000075                          .WORD   75
16792 105650  000076                          .WORD   76
16793
```

```
16795                                   .SBTTL  TEST  - BYTE ALLOCATION TEST
16796                                   ;BYTE ALLOCATION TEST   THIS TEST WILL VERIFY THAT THE CACHE SYSTEM CORRECTLY
16797                                   ;HANDLES BYTE ACCESSES. THE TEST WILL FIRST CHECK THAT A WRITE ACCESS WHICH
16798                                   ;IS A MISS DOES NOT UPDATE THE CACHE. THIS WILL BE DONE BY FIRST ASSURRING
16799                                   ;A MISS AT A TEST LOCATION. THEN A WRITE BYTE ACCESS WILL BE DONE. FINALLY
16800                                   ;THE LOCATION WILL BE READ. THE WRITE BYTE SHOULD NOT HAVE ALLOCATED THE
16801                                   ;ADDRESS. NEXT THE TEST WILL VERIFY THAT IF A WRITE BYTE ACCESS IS A HIT
16802                                   ;THAT THE CACHE LOCATION IS UPDATED. THE TEST WILL FIRST WRITE A TEST LOCATION
16803                                   ;WITH A PATTERN TO ALLOCATE THE ADDRESS. THEN A SECOND PATTERN WILL BE WRITTEN
16804                                   ;TO THE HIGH BYTE OF THE TEST LOCATION. THE TEST LOCATION WILL THEN BE READ.
16805                                   ;IF THE HIGH BYTE OF THE TEST LOCATION IS EQUAL TO THE SECOND PATTERN THEN THE
16806                                   ;LOCATION WAS UPDATED. THE SECOND TEST WILL BE REPEATED TO TEST LOW BYTE
16807                                   ;ACCESSES.
16808                                   ;
16809                                   ;BGNTST
16810                                   ;SAVE VECTOR 114
16811                                   ;LET VECTOR 114 POINT TO BYTE PARITY ROUTINE
16812                                   ;SET PARITY ABORT BIT IN CACHE CONTROL REGISTER
16813                                   ;READ TEST LOCATION + 4K
16814                                   ;WRITE LOW BYTE TO TEST ADDRESS
16815                                   ;READ LOW BYTE OF TEST ADDRESS
16816                                   ;IF HIT/MISS REGISTER BIT 1 SET THEN
16817                                   ;.      ERROR WRITE BYTE ALLOCATES THE CACHE
16818                                   ;ENDIF
16819                                   ;WRITE PATTERN 1 TO TEST LOCATION
16820                                   ;WRITE BYTE PATTERN 2 TO TEST LOCATION+1
16821                                   ;IF BIT1 NOTSETIN HIT/MISS REGISTER THEN
16822                                   ;.      IF B0 PARITY ERROR THEN
16823                                   ;.      .       ERROR LOW BYTE PARITY ERROR ON WRITE BYTE HIT
16824                                   ;.      ELSE
16825                                   ;.      .       IF B1 PARITY ERROR THEN
16826                                   ;.      .       .       ERROR HIGH BYTE PARITY ERROR ON WRITE BYTE HIT
16827                                   ;.      .       ELSE
16828                                   ;.      .       .       WRITE BYTE HIT DOES NOT RECORD A HIT
16829                                   ;.      .       ENDIF
16830                                   ;.      ENDIF
16831                                   ;ELSE
16832                                   ;.      READ TEST LOCATION
16833                                   ;.      IF RECIEVED DATA NOT EQUAL TO EXPECTED DATA THEN
16834                                   ;.      .       IF BYTE DATA REVERSED THEN
16835                                   ;.      .       .       ERROR BYTES REVERSED ON WRITE CYCLES
16836                                   ;.      .       ELSE
16837                                   ;.      .       .       IF HIGH BYTE DATA ZERO THEN
16838                                   ;.      .       .       .       ERROR IN WRITING TO HIGH BYTE
16839                                   ;.      .       .       ELSE
16840                                   ;.      .       .       .       ERROR IN WRITING TO HIGH BYTE
16841                                   ;.      .       .       ENDIF
16842                                   ;.      .       ENDIF
16843                                   ;.      ENDIF
16844                                   ;ENDIF
16845                                   ;WRITE PATTERN 1 TO TEST LOCATION
16846                                   ;WRITE BYTE PATTERN 2 TO TEST LOCATION LOW BYTE
16847                                   ;IF BIT1 NOTSETIN HIT/MISS REGISTER THEN
16848                                   ;.      IF B0 PARITY ERROR THEN
16849                                   ;.      .       ERROR LOW BYTE PARITY
16850                                   ;.      ELSE
16851                                   ;.      .       IF B1 PARITY ERROR THEN
```

```
16852                                   ;.      .       .       ERROR HIGH BYTE PRITY
16853                                   ;.      .       ELSE
16854                                   ;.      .       .       WRITE BYTE HIT DOES NOT RECORD A HIT
16855                                   ;.      .       ENDIF
16856                                   ;.      ENDIF
16857                                   ;ELSE
16858                                   ;.      READ TEST LOCATION
16859                                   ;.      IF RECIEVED DATA NOT EQUAL TO EXPECTED DATA THEN
16860                                   ;.      .       IF BYTE DATA REVERSED THEN
16861                                   ;.      .       .       ERROR BYTES REVERSED
16862                                   ;.      .       ELSE
16863                                   ;.      .       .       IF LOW BYTE DATA ZERO THEN
16864                                   ;.      .       .       .       ERROR IN DATA PATH
16865                                   ;.      .       .       ELSE
16866                                   ;.      .       .       .       ERROR IN DATA PATH
16867                                   ;.      .       .       ENDIF
16868                                   ;.      .       ENDIF
16869                                   ;.      ENDIF
16870                                   ;ENDIF
16871                                   ;CLEAR CACHE CONTROL REGISTER
16872                                   ;RESTORE VECTOR 114
16873                                   ;EXIT   TST
16874                                   ;
16875                                   ;BYTE PARITY ROUTINE:   SAVE MSER
16876                                   ;                       RETURN
16877                                   ;ENDTST
16878
16879                                   ;;*****************************************************************
    105652  000004                      TST7:   SCOPE
16880 105654  000240                              NOP
16881 105656  005737  003032                      TST     CCHPAS                  ;have done enough inclusive passes?
16882 105662  001002                              BNE     99$                     ; not yet
16883 105664  000240                              NOP                             ; debug aid
16884 105666  000517                              BR      TST10           ;;GO TO NEXT TEST
16885 105670  000240                      99$:    NOP
16886 105672  013737  000114  003012              MOV     @#114,  SLOC00          ;SAVE VECTOR 114
16887 105700  012737  106114  000114              MOV     #BYPARR,@#114           ;LET VECTOR 114 POINT TO ABORT ROUTINE
16888 105706  005003                              CLR     R3                      ;CLEAR MSER SAVE REGISTER
16889 105710  012737  000200  177746              MOV     #BIT07, CCR             ;SET PARITY ABORT BIT IN CCR
16890 105716  005737  023162                      TST     TSTLOC+8192.            ;READ TEST LOCATION + 4K TO CAUSE MISS
16891 105722  042737  001000  177520              BIC     #1000,BCSR              ;DISABLE HALT ON BREAK
16892 105730  105037  003162                      CLRB    TSTLOC                  ;WRITE LOW BYTE OF TEST LOCATION
16893 105734  105737  003162                      TSTB    TSTLOC                  ;READ LOW BYTE OF TEST LOCATION
16894 105740  013737  177752  114150              MOV     HITMIS,RECDAT           ;STORE REGISTER
16895 105746  032737  000004  114150              BIT     #BIT02,RECDAT           ;IF BIT1 OF HIT/MISS REGISTER SET
16896 105754  001401                              BEQ     1$                      ;THEN
16897 105756  104020                              ERROR   +20                     ;WRITE BYTE ALLOCATES CACHE
16898 105760  005037  003162              1$:     CLR     TSTLOC                  ;WRITE PATTERN 1 TO TEST LOCATION
16899 105764  152737  000377  003163              BISB    #377,   @#TSTLOC+1      ;WRITE PATTERN 2 TO HIGH BYTE
16900 105772  013737  177752  114150              MOV     HITMIS,RECDAT           ;STORE REGISTER
16901 106000  032737  000004  114150              BIT     #BIT02,RECDAT           ;IF BIT1 NOTSETIN HIT/MISS REGISTER
16902 106006  001002                              BNE     2$                      ;THEN
16903 106010  104021                      101$:   ERROR   +21                     ;WRITE BYTE HIT DOES NOT RECORD HIT
16904 106012  000405                              BR      3$                      ;ELSE
16905 106014  022737  177400  003162      2$:     CMP     #177400,@#TSTLOC        ;IF TEST LOCATION NOT EQUAL TO
16906 106022  001401                              BEQ     3$                      ;PATTERN 2 THEN
16907 106024  104022                              ERROR   +22                     ;BYTES REVERSED ON WRITE CYCLES
```

F8

TEST - BYTE ALLOCATION TEST

```
16908 106026  005037  003162          3$:     CLR     TSTLOC                  ;WRITE PATTERN 1 TO TEST LOCATION
16909 106032  152737  000377  003162          BISB    #377,   @#TSTLOC        ;WRITE PATTERN 2 TO LOW BYTE
16910 106040  013737  177752  114150          MOV     HITMIS,RECDAT           ;STORE REGISTER
16911 106046  032737  000004  114150          BIT     #BIT02,RECDAT           ;IF BIT1 NOTSETIN HIT/MISS REGISTER
16912 106054  001001                          BNE     4$                      ;THEN
16913 106056  104012                          ERROR   ·12                     ;LOW BYTE PARITY ERROR ON WRITE BYTE HIT
16914 106060  022737  000377  003162  4$:     CMP     #377,   @#TSTLOC        ;IF TEST LOCATION NOT EQUAL TO
16915 106066  001401                          BEQ     5$                      ;PATTERN 2 THEN
16916 106070  104022                          ERROR   ·22                     ;BYTES REVERSED ON WRITE CYCLES
16917 106072  005037  177746          5$:     CLR     CCR                     ;CLEAR CCR BEFORE EXIT
16918 106076  052737  001000  177520          BIS     #1000,BCSR              ;ENABLE HALT ON BREAK
16919 106104  013737  003012  000114          MOV     SLOC00, @#114           ;RESTORE VECTOR 114
16920 106112  000405                          BR      TST10           ;;GO TO NEXT TEST
16921
16922
16923 106114  011637  001122          BYPARR: MOV     (SP),   $BDADR          ;SAVE ADDRESS THAT CAUSE ABORT
16924 106120  013703  177744                  MOV     MSER,   R3              ;SAVE CONTENTS OF MSER
16925 106124  000002                          RTI                             ;RETURN
16926
```

TEST - PDR BIT15 (BYPASS) TEST

```
                                        .SBTTL  TEST    PDR BIT15 (BYPASS) TEST
                                        ;PDR BIT15 (BYPASS) TEST   THIS TEST WILL VERIFY THAT WHEN BIT<15> IS SET
                                        ;IN A PDR AND AN ACCESS IS MADE TO A LOCATION MAPPED BY THE SELECTED PDR
                                        ;THAT WOULD NORMALLY CAUSE A CACHE ACCESS, THE CACHE IS BYPASSED (I.E. THE
                                        ;CACHE LOCATION IS INVALIDATED AND A MAIN MEMORY IS READ. THIS WILL BE DONE
                                        ;BY FIRST WRITING A TEST LOCATION WITH A KNOWN PATTERN TO ALLOCATE THE ADDRESS.
                                        ;THEN THE LOCATION WILL BE WRITTEN AGAIN WITH THE MMU ENABLED AND BIT <15> SET
                                        ;IN THE CONTROLING PDR WITH A NEW PATTERN. THE LOCATION WILL THEN BE READ
                                        ;AND A MISS SHOULD BE LOGGED IN THE HIT/MISS REGISTER. THIS READ WILL ALSO
                                        ;BRING VALID DATA INTO CACHE FROM MEMORY. THE MMU WILL AGAIN BE ENABLED
                                        ;WITH BIT <15> SET AND A READ CYCLE DONE. THIS SHOULD INVALIDATE THE CACHE.
                                        ;NEXT THE MMU WILL BE DISABLED AND THE LOCATION READ FOR A THIRD TIME. THIS
                                        ;WILL BE DONE WITH BIT<15> STILL SET. A MISS SHOULD ALSO BE LOGGED. LASTLY
                                        ;BIT<15> WILL BE CLEARED AND THE MMU ENABLED AND THE LOCATION AGAIN READ.
                                        ;THIS TIME A CACHE HIT SHOULD BE LOGGED.
                                        ;
                                        ;
                                        ;BGNTST
                                        ;SETUP MMU REGISTERS
                                        ;WRITE TEST LOCATION WITH PATTERN 1
                                        ;SET BIT<15> IN PDR FOR TEST LOCATION
                                        ;ENABLE MMU
                                        ;WRITE TEST LOCATION WITH PATTERN 2
                                        ;DISABLE MMU AND CLEAR BIT<15> IN PDR
                                        ;READ TEST LOCATION
                                        ;IF HIT/MISS REGISTER BIT<1> SET THEN
                                        ;.      ERROR CONDITIONAL BYPASS DOESN'T INVALIDATE CACHE
                                        ;ENDIF
                                        ;SET BIT<15> IN PDR FOR TEST LOCATION
                                        ;ENABLE MMU
                                        ;READ TEST LOCATION (SHOULD INVALIDATE CACHE)
                                        ;DISABLE MMU
                                        ;READ TEST LOCATION
                                        ;IF HIT/MISS REGISTER BIT<1> SET THEN
                                        ;.      ERROR CONDITIONAL BYPASS DOESN'T INVALIDATE CACHE
                                        ;ENDIF
                                        ;CLEAR BIT<15> IN PDR
                                        ;TURN ON MMU
                                        ;READ TEST LOCATION
                                        ;IF HIT/MISS REGISTER BIT<1> NOT SET THEN
                                        ;.      ERROR IN RECORDING HITS IN HIT/MISS
                                        ;ENDIF
                                        ;TURN OFF MMU
                                        ;ENDTST
                                        ;
                                        ;;*****************************************************************
 106126  000004                         TST10:  SCOPE
16974 106130  000240                            NOP
16975 106132  005737  003032                    TST     CCHPAS                  ;have done enough inclusive passes?
16976 106136  001002                            BNE     99$                     ; not yet
16977 106140  000240                            NOP                             ; debug aid
16978 106142  000512                            BR      TST11           ;;GO TO NEXT TEST
16979 106144  000240                    99$:    NOP
16980 106146  004737  136574                    JSR     PC,     INITMM          ;SETUP MEMORY MANAGEMEMT REGISTERS
16981 106152  005037  003162                    CLR     @#TSTLOC                ;WRITE TEST LOCATION WITH PATTERN 1
16982 106156  012737  177777  001124            MOV     #177777,$GDDAT          ;SAVE DATA IN MEMORY
16983 106164  052737  100000  172300            BIS     #BIT15, KIPDR0          ;SET BIT15 IN PDR FOR TEST LOCATION
```

H8

SEQ 0305

TEST - PDR BIT15 (BYPASS) TEST

```
16984 106172  005237  177572                        INC     SR0             ;TURN ON MEMORY MANAGEMENT
16985 106176  012737  177777  003162                MOV     #177777,@#TSTLOC ;WRITE TEST LOCATION WITH PATTERN 2
16986 106204  005037  177572                        CLR     SR0             ;TURN OFF MEMORY MANAGEMENT
16987 106210  042737  100000  172300                BIC     #BIT15, KIPDR0  ;CLEAR BIT15 IN PDR FOR TEST LOCATION
16988 106216  042737  001000  177520                BIC     #1000,BCSR      ;DISABLE HALT ON BREAK
16989 106224  013701  003162                        MOV     @#TSTLOC,R1     ;READ TEST LOCATION
16990 106230  013737  177752  114150                MOV     HITMIS,RECDAT   ;SAVE HIT/MISS REGISTER
16991 106236  032737  000004  114150                BIT     #BIT02,RECDAT   ;IF HIT/MISS REGISTER BIT 1 SET
16992 106244  001401                                BEQ     2$              ;THEN
16993 106246  104023                                ERROR   +23             ;CONDITIONAL BYPASS DOESN'T INVALIDATE CACHE
16994 106250  052737  100000  172300  2$:           BIS     #BIT15, KIPDR0  ;SET BIT15 IN PDR FOR TEST LOCATION
16995 106256  005237  177572                        INC     SR0             ;TURN ON MEMORY MANAGEMENT
16996 106262  005737  003162                        TST     TSTLOC          ;READ TEST LOCATION
16997 106266  042737  100000  172300                BIC     #BIT15, KIPDR0  ;CLEAR BIT 15 IN PDR
16998 106274  005037  177572                        CLR     SR0             ;TURN OFF MMU
16999 106300  013701  003162                        MOV     @#TSTLOC,R1     ;READ TEST LOCATION
17000 106304  013737  177752  114150                MOV     HITMIS,RECDAT   ;STORE HIT/MISS TO REGISTER 2
17001 106312  032737  000004  114150                BIT     #BIT02,RECDAT   ;IF HIT/MISS REGISTER BIT 1 SET
17002 106320  001401                                BEQ     3$              ;THEN
17003 106322  104023                                ERROR   +23             ;CONDITIONAL BYPASS DOESN'T INVALIDATE 0 CACHE
17004 106324  005237  177572          3$:           INC     SR0             ;TURN ON MEMORY MANAGEMENT
17005 106330  005737  003162                        TST     @#TSTLOC        ;READ TEST LOCATION ONE MORE TIME
17006 106334  013737  177752  114150                MOV     HITMIS,RECDAT   ;STORE HIT/MISS TO REGISTER 2
17007 106342  032737  000004  114150                BIT     #BIT02,RECDAT   ;IF HIT/MISS REGISTER BIT 1 NOT SET
17008 106350  001001                                BNE     4$              ;THEN
17009 106352  104045                                ERROR   +45             ;ERROR IN RECORDING HITS IN HIT/MISS
17010 106354  042737  000001  177572  4$:           BIC     #BIT00, SR0     ;TURN OFF MMU
17011 106362  052737  001000  177520                BIS     #1000,BCSR      ;ENABLE HALT ON BREAK
17012
```

```
17014                                   .SBTTL  TEST    FLUSH CACHE TEST
17015                                   ;FLUSH CACHE TEST   THIS TEST WILL VERIFY THAT WHEN CCR BIT<8> IS
17016                                   ;SET, THE ENTIRE CACHE IS INVALIDATED. FIRST 8K B TES OF ADDRESSES
17017                                   ;WILL BE READ TO ALLOCATE CACHE. THEN THE CACHE WILL BE FLUSHED.
17018                                   ;THE SAME SET OF ADDRESS WILL THEN BE READ AND THE HIT/MISS REGISTER
17019                                   ;CHECKED AFTER EACH READ FOR A MISS TO BE LOGGED.
17020                                   ;
17021                                   ;BGNTST
17022                                   ;GET FIRST ADDRESS OF 8KB BUFFER
17023                                   ;INITIALIZE LOOP COUNTER TO 4KWORD COUNT
17024                                   ;DO UNTIL LOOP COUNTER = 0
17025                                   ;.      READ @ADDRESS+
17026                                   ;ENDDO
17027                                   ;GET FIRST ADDRESS OF 8KB BUFFER
17028                                   ;INITIALIZE LOOP COUNTER TO 2KWORD COUNT
17029                                   ;INITIALIZE MISS COUNT TO 4KWORD COUNT
17030                                   ;FLUSH CACHE
17031                                   ;DO UNTIL LOOP COUNTER = 0
17032                                   ;.      READ @ADDRESS+
17033                                   ;.      ICREMENT ADDRESS TO READ EVERY 2WORDS
17034                                   ;.      IF HIT/MISS REGISTER BIT<1> SET THEN
17035                                   ;.      .       DECREMENT MISS COUNT
17036                                   ;.      ENDIF
17037                                   ;ENDDO
17038                                   ;GET FIRST ADDRESS OF 8KB BUFFER
17039                                   ;INITIALIZE LOOP COUNTER TO 4KWORD COUNT
17040                                   ;DO UNTIL LOOP COUNTER = 0
17041                                   ;.      READ @ADDRESS+
17042                                   ;ENDDO
17043                                   ;GET LAST ADDRESS OF 8KB BUFFER
17044                                   ;INITIALIZE LOOP COUNTER TO 2KWORD COUNT
17045                                   ;FLUSH CACHE
17046                                   ;DO UNTIL LOOP COUNTER = 0
17047                                   ;.      READ @ADDRESS-
17048                                   ;.      DECREMENT ADDRESS TO READ EVERY 2WORDS
17049                                   ;.      IF HIT/MISS REGISTER BIT<1> SET THEN
17050                                   ;.      .       DECREMENT MISS COUNT
17051                                   ;.      ENDIF
17052                                   ;ENDDO
17053                                   ;IF MISS COUNT NOT EQUAL TO 4096. THEN
17054                                   ;.      ERROR HITS RECORDED AFTER FLUSHING CACHE
17055                                   ;ENDIF
17056                                   ;ENDTST
17057                                   ;
17058                                   ;;*****************************************************************
    106370  000004                      TST11:  SCOPE
17059 106372  000240                          NOP
17060 106374  005737  003032                  TST     CCHPAS                  ;have done enough inclusive passes?
17061 106400  001002                          BNE     99$                     ; not yet
17062 106402  000240                          NOP                             ; debug aid
17063 106404  000517                          BR      TST12           ;;GO TO NEXT TEST
17064 106406  000240                  99$:    NOP
17065 106410  004737  136574                  JSR     PC,     INITMM          ;INITIALIZE MMU
17066 106414  012737  001600  172354          MOV     #1600,KIPAR6            ;LET PAR6 POINT TO LOW 28K
17067 106422  012701  140000                  MOV     #140000,R1              ;GET ADDRESS OF 8K BYTE BUFFER
17068 106426  005237  177572                  INC     SR0                     ;ENABLE MMU
17069 106432  012702  010000                  MOV     #4096., R2              ;INIT LOOP COUNTER TO 4K WORD COUNT
```

TEST - FLUSH CACHE TEST

```
17070 106436  005721                  1$:   TST   (R1)+              ;READ ADDRESS TO ALLOCATE CACHE
17071 106440  077202                        SOB   R2,     1$          ;DO UNTIL ALL LOCATIONS ALLOCATED
17072 106442  012701  140000              MOV   #140000,R1          ;GET ADDRESS OF 8K BYTE BUFFER
17073 106446  012702  004000              MOV   #2048., R2          ;INIT LOOP COUNTER TO 2K WORD COUNT
17074 106452  012703  010000              MOV   #4096., R3          ;INITIALIZE MISS COUNT TO 4K
17075 106456  012737  000400  177746      MOV   #400,   CCR         ;FLUSH CACHE
17076 106464  042737  001000  177520      BIC   #1000,BCSR          ;DISABLE HALT ON BREAK
17077 106472  005721                  2$:   TST   (R1)+              ;READ ADDRESS
17078 106474  013737  177752  114150      MOV   @#HITMIS,RECDAT     ;STORE REGISTER
17079 106502  032737  000004  114150      BIT   #BIT02,RECDAT       ;IF HIT/MISS REGISTER NOT EQUAL
17080 106510  001401                      BEQ   3$                  ;TO ZERO THEN
17081 106512  005303                      DEC   R3                  ;DECREMENT MISS COUNT
17082 106514  062701  000002          3$:   ADD   #2,R1               ;DO IN 2 WORD INCREMENTS (PMI MEMORY)
17083 106520  077214                      SOB   R2,     2$          ;LOOP UNTIL ENTIRE CACHE CHECKED
17084 106522  052737  001000  177520      BIS   #1000,BCSR          ;ENABLE HALT ON BREAK
17085 106530  012702  010000              MOV   #4096.,R2           ;DO FOR 4K
17086 106534  012701  140000              MOV   #140000,R1          ;STARTING WITH 0
17087 106540  005721                  4$:   TST   (R1)+              ;ALLOCATE IN CACHE
17088 106542  077202                      SOB   R2,4$               ;DO FOR 4K
17089 106544  012702  004000              MOV   #2048.,R2           ;DO FOR THE 2ND  ALF
17090 106550  012701  160000              MOV   #160000,R1          ;START WITH LAST LOCATION
17091 106554  012737  000400  177746      MOV   #400,CCR            ;FLUSH CACHE
17092 106562  042737  001000  177520      BIC   #1000,BCSR          ;DISABLE HALT ON BREAK
17093 106570  005741                  5$:   TST   -(R1)               ;READ ADDRESS
17094 106572  013737  177752  114150      MOV   @#HITMIS,RECDAT     ;STORE REGISTER
17095 106600  032737  000004  114150      BIT   #BIT02,RECDAT       ;IF HIT/MISS REGISTER NOT EQUAL
17096 106606  001401                      BEQ   6$                  ;TO ZERO THEN
17097 106610  005303                      DEC   R3                  ;DECREMENT MISS COUNT
17098 106612  162701  000002          6$:   SUB   #2,R1               ;DO IN 2 WORD DECREMENTS
17099 106616  077214                      SOB   R2,5$               ;DO FOR 2K WORDS
17100 106620  052737  001000  177520      BIS   #1000,BCSR          ;ENABLE HALT ON BREAK
17101 106626  005037  177572              CLR   SR0                 ;TURN OFF MMU
17102 106632  022703  010000              CMP   #4096., R3          ;IF MISS COUNT NOT EQUAL TO 4K WORDS
17103 106636  001401                      BEQ   8$                  ;THEN
17104 106640  104024                      ERROR +24                 ;HITS RECORDED AFTER FLUSHING CACHE
17105 106642                          8$:
      106642  000400                      BR    TST12           ;;GO TO NEXT TEST
17106
```

```
17108                                   .SBTTL  TEST    UNCONDITIONAL BYPASS TEST
17109                                   ;UNCONDITIONAL BYPASS TEST   THIS TEST WILL VERIFY THAT WHEN CCR
17110                                   ;BIT<9> IS SET, A MEMORY REFERENCE IS FORCED TO MAIN MEMORY. THIS
17111                                   ;WILL ALSO VERIFY THAT READ AND WRITE HIT INVALIDATE THE CACHE
17112                                   ;LOCATIONS WHEN BYPASS IS SET.
17113                                   ;
17114                                   ;
17115                                   ;BGNTST
17116                                   ;
17117                                   ;READ TEST LOCATIONS TO SETUP POSSIBILITY OF HIT
17118                                   ;SET CCR BIT<9>
17119                                   ;
17120                                   ;READ FIRST TEST LOCATION
17121                                   ;IF HIT/MISS REGISTER BIT<1> NOT SET THEN
17122                                   ;.      ERROR IN RECORDING HITS IN HIT/MISS
17123                                   ;ENDIF
17124                                   ;
17125                                   ;WRITE SECOND TEST LOCATION
17126                                   ;IF HI./MISS REGISTER BIT<1> NOT SET THEN
17127                                   ;.      ERROR IN RECORDING HITS IN HIT/MISS
17128                                   ;ENDIF
17129                                   ;CLEAR CCR BIT<9>
17130                                   ;
17131                                   ;READ FIRST LOCATION
17132                                   ;IF HIT/MISS REGISTER BIT<1> SET THEN
17133                                   ;.      ERROR BYPASS DOESN'T INVALIDATE CACHE
17134                                   ;ENDIF
17135                                   ;
17136                                   ;READ SECOND LOCATION
17137                                   ;IF HIT/MISS REGISTER BIT<1> SET THEN
17138                                   ;.      ERROR BYPASS DOESN'T INVALIDATE CACHE
17139                                   ;ENDIF
17140                                   ;
17141                                   ;ENDTST
17142                                   ;
17143                                   ;
17144
17145                                   ;;*****************************************************************
    106644  000004                      TST12:  SCOPE
17146 106646  000240                              NOP
17147 106650  005737  003032                      TST     CCHPAS                  ;have done enough inclusive passes?
17148 106654  001002                              BNE     99$                     ; not yet
17149 106656  000240                              NOP                             ; debug aid
17150 106660  000471                              BR      TST13           ;;GO TO NEXT TEST
17151 106662  000240                      99$:    NOP
17152
17153 106664  005737  003162                      TST     @#TSTLOC                ;ALLOCATE FIRST TEST LOCATION
17154 106670  005737  003164                      TST     @#TSTLOC+2              ;ALLOCATE SECOND TEST LOCATION
17155 106674  052737  001000  177746              BIS     #BIT09, CCR             ;SET CCR BIT 9 (CACHE BYPASS)
17156 106702  042737  001000  177520  1$:         BIC     #1000,BCSR              ;DISABLE HALT ON BREAK
17157 106710  005737  003162                      TST     @#TSTLOC                ;READ FIRST TEST LOCATION
17158 106714  013737  177752  114150              MOV     HITMIS,RECDAT           ;STORE HIT/MISS TO REGISTER 2
17159 106722  032737  000004  114150              BIT     #BIT02,RECDAT           ;IF HIT/MISS REG. BIT 1 NOT SET
17160 106730  001001                              BNE     2$                      ;THEN
17161 106732  104045                              ERROR   +45                     ;ERROR IN RECORDING HITS IN HIT/MISS WITH CCR<9>=1
17162 106734  005037  003164              2$:     CLR     @#TSTLOC+2              ;WRITE SECOND LOCATION
17163 106740  013737  177752  114150              MOV     HITMIS,RECDAT           ;STORE HIT/MISS TO REGISTER 2
```

L8

SEQ 0309

TEST - UNCONDITIONAL BYPASS TEST

```
17164 106746 032737 000004 114150         BIT     #BIT02,RECDAT           ;IF HIT/MISS REG. BIT 1 NOT SET
17165 106754 001001                       BNE     3$                      ;THEN
17166 106756 104045                       ERROR   .45                     ;ERROR IN RECORDING HITS IN HIT/MISS WITH CCR<4> 1
17167 106760 005737 003162         3$:    TST     @#TSTLOC                ;READ FIRST TEST LOCATION
17168 106764 013737 177752 114150         MOV     HITMIS,RECDAT           ;STORE HIT/MISS TO REGISTER 2
17169 106772 032737 000004 114150         BIT     #BIT02,RECDAT           ;IF HIT/MISS REG. BIT 1 SET
17170 107000 001401                       BEQ     4$                      ;THEN
17171 107002 104025                       ERROR   .25                     ;BYPASS DOESN'T INVALIDATE CACHE
17172 107004 005037 003164         4$:    CLR     @#TSTLOC+2              ;WRITE SECOND LOCATION
17173 107010 013737 177752 114150         MOV     HITMIS,RECDAT           ;STORE HIT/MISS TO REGISTER 2
17174 107016 032737 000004 114150         BIT     #BIT02,RECDAT           ;IF HIT/MISS REG. BIT 1 SET
17175 107024 001401                       BEQ     5$                      ;THEN
17176 107026 104025                       ERROR   .25                     ;BYPASS DOESN'T INVALIDATE CACHE
17177 107030 042737 001000 177746 5$:     BIC     #BIT09,CCR              ;RESTORE CCR
17178 107036 052737 001000 177520         BIS     #1000,BCSR              ;ENABLE HALT ON BREAK
17179
```

```
17181                                   .SBTTL  TEST    WRITE WRONG DATA PARITY TEST
17182                                   ;WRITE WRONG DATA PARITY TEST - THIS TEST WILL VERIFY THAT WHEN CCR
17183                                   ;BIT<6> = 1 AND CCR BITS<7,0> = 0,1, A READ MISS OCCURS AFTER A
17184                                   ;WRITE. THE WRITE WITH CCR BIT<6> = 1 TO A LOCATION WILL CAUSE A
17185                                   ;CACHE UPDATE AND WRONG PARITY TO BE WRITTEN SO WHEN THE LOCATION
17186                                   ;IS READ INSTEAD OF A HIT BEING RECORDED THE CACHE PARITY ERROR
17187                                   ;WILL CAUSE A CACHE MISS. THIS TEST WILL BE DONE A BYTE AT A TIME.
17188                                   ;
17189                                   ;BGNTST
17190                                   ;SAVE CONTENTS OF VECTOR 114
17191                                   ;LET VECTOR 114 POINT TO ABORT ROUTINE
17192                                   ;CLEAR MSER
17193                                   ;IF MSER NOT CLEAR THEN
17194                                   ;.      ERROR MSER DOESN'T CLEAR ON WRITE REFERENCE
17195                                   ;ENDIF
17196                                   ;WRITE TEST LOCATION
17197                                   ;SET BITS<6,0> IN CCR
17198                                   ;WRITE TEST LOCATION LOW BYTE
17199                                   ;CLEAR CCR BIT<6> (WRITE WRONG DATA PARITY)
17200                                   ;INITIALIZE ERROR INDICATORS
17201                                   ;READ TEST LOCATION LOW BYTE
17202                                   ;IF BIT<1> SET IN HIT/MISS REGISTER THEN
17203                                   ;.      PARITY ERROR DOESN'T CAUSE MISS
17204                                   ;ELSE
17205                                   ;.      IF MSER BITS <7:5> ZERO THEN
17206                                   ;.      .       PARITY ERROR DOESN'T SET MSER PROPERLY
17207                                   ;.      ELSE
17208                                   ;.      .       SET ERROR IN LOW BYTE INDICATOR
17209                                   ;.      ENDIF
17210                                   ;.      ENDIF
17211                                   ;ENDIF
17212                                   ;CLEAR MSER
17213                                   ;IF MSER NOT CLEAR THEN
17214                                   ;.      .       ERROR MSER DOESN'T CLEAR ON WRITE REFERENCE
17215                                   ;ENDIF
17216                                   ;SET CCR BIT<6>
17217                                   ;WRITE TEST LOCATION HIGH BYTE
17218                                   ;CLEAR CCR BIT<6>
17219                                   ;READ TEST LOCATION HIGH BYTE
17220                                   ;IF BIT<1> SET IN HIT/MISS REGISTER THEN
17221                                   ;.      IF MSER BITS<7:5> NOT SET THEN
17222                                   ;.      .       PARITY ERROR DOESN'T CAUSE A MISS
17223                                   ;.      ELSE
17224                                   ;.      .       SET ERROR IN HIGH BYTE INDICATOR
17225                                   ;.      ENDIF
17226                                   ;ELSE
17227                                   ;.      IF MSER BITS <7:5> ZERO THEN
17228                                   ;.      .       PARITY ERROR DOESN'T SET MSER
17229                                   ;.      ENDIF
17230                                   ;ENDIF
17231                                   ;RESTORE CONTENTS OF VECTOR 114
17232                                   ;IF ERROR INDICATORS SET THEN
17233                                   ;.      IF ERROR IN BOTH BYTES THEN
17234                                   ;.      .       PARITY ERROR IGNORED
17235                                   ;.      ELSE
17236                                   ;.      .       IF ERROR IN LOW BYTE THEN
17237                                   ;.      .       .       LOW BYTE PARITY ERROR IGNORED
```

```
                                ;.      .       ELSE
                                ;.      .               HIGH BYTE PARITY ERROR IGNORED
                                ;.      .       ENDIF
                                ;.      ENDIF
                                ;ENDIF
                                ;EXIT   TST
                                ;
                                ;DATA PARITY ABORT ROUTINE:
                                ;                               ILLEGAL PARITY INTERRUPT
                                ;                               RETURN
                                ;
                                ;ENDTST
                                ;;*****************************************************************
     107044  000004             TST13:  SCOPE
17251 107046  000240                     NOP
17252 107050  005737  003032            TST     CCHPAS                  ;have done enough inclusive passes?
17253 107054  001002                    BNE     99$                     ; not yet
17254 107056  000240                    NOP                             ; debug aid
17255 107060  000537                    BR      TST14           ;;GO TO NEXT TEST
17256 107062  000240             99$:    NOP
17257 107064  013737  000114  003012    MOV     @#114,  SLOC00          ;SAVE CONTENTS OF VECTOR 114
17258 107072  012737  107350  000114    MOV     #DAPABO,@#114           ;LET VECTOR POINT TO ABORT ROUTINE
17259 107100  005037  177744            CLR     MSER                    ;CLEAR MSER
17260 107104  005737  177744            TST     MSER                    ;IF MSER NOT CLEAR
17261 107110  001401                    BEQ     1$                      ;THEN
17262 107112  104026                    ERROR   +26                     ;MSER DOES NOT CLEAR ON WRITE REFERENCE
17263 107114  005037  003162     1$:    CLR     @#TSTLOC                ;WRITE TEST LOCATION TO ALLOCATE CACHE
17264 107120  042737  001000  177520    BIC     #1000,BCSR              ;DISABLE HALT ON BREAK
17265 107126  012737  000101  177746    MOV     #101,   CCR             ;SET BITS<6,0> IN CCR
17266 107134  112737  000377  003162    MOVB    #377,   TSTLOC          ;WRITE LOW BYTE WITH BAD PARITY
17267 107142  042737  000100  177746    BIC     #BIT06, CCR             ;CLEAR WRITE WRONG DATA PARITY BIT
17268 107150  005002                    CLR     R2                      ;CLEAR ERROR INDICATORS
17269 107152  105737  003162            TSTB    @#TSTLOC                ;READ LOW BYTE OF TEST LOCATION
17270 107156  013703  177752            MOV     HITMIS, R3              ;SAVE HIT/MISS
17271 107162  032703  000004            BIT     #BIT02, R3              ;IF BIT 1 SET IN HIT/MISS REGISTER
17272 107166  001402                    BEQ     2$                      ;THEN
17273 107170  104027                    ERROR   +27                     ;PARITY ERROR DON'T CAUSE A MISS
17274 107172  000405                    BR      3$                      ;ELSE
17275 107174  032737  000340  177744  2$:    BIT     #340,   MSER            ;IF MSER BIT<7:5> NOT ZERO
17276 107202  001001                    BNE     3$                      ;THEN
17277 107204  104030                    ERROR   +30                     ;PARITY ERROR DON'T SET MSER WITH CCR<7>=0
17278 107206  005037  177744     3$:    CLR     MSER                    ;CLEAR MSER
17279 107212  005737  177744            TST     MSER                    ;IF MSER NOT CLEAR
17280 107216  001401                    BEQ     4$                      ;THEN
17281 107220  104026                    ERROR   +26                     ;MSER DOES NOT CLEAR ON WRITE REFERENCE
17282 107222  052737  000100  177746  4$:    BIS     #BIT06, CCR             ;SET CCR BIT 6 (WRITE WRONG PARITY)
17283 107230  112737  000377  003163    MOVB    #377,   @#TSTLOC+1      ;WRITE HIGH BYTE OF TEST LOCATION
17284 107236  042737  000100  177746    BIC     #BIT06, CCR             ;CLEAR WRITE WRONG PARITY BIT
17285 107244  105737  003163            TSTB    @#TSTLOC+1              ;READ HIGH BYTE OF TEST LOCATION
17286 107250  013737  177752  114150    MOV     HITMIS,RECDAT           ;SAVE HIT/MISS
17287 107256  032737  000004  114150    BIT     #BIT02,RECDAT           ;IF BIT 1 SET IN HIT/MISS REGISTER
17288 107264  001402                    BEQ     5$                      ;THEN
17289 107266  104027                    ERROR   +27                     ;PARITY ERROR DON'T CAUSE A MISS
17290 107270  000405                    BR      6$                      ;ELSE
17291 107272  032737  000340  177744  5$:    BIT     #340,   MSER            ;IF BITS <7:5> ZERO
17292 107300  001001                    BNE     6$                      ;THEN
17293 107302  104030                    ERROR   +30                     ;PARITY ERROR DON'T SET MSER WITH CCR<7>=0
```

TEST - WRITE WRONG DATA PARITY TEST

```
17294 107304  005037  177746          6$:     CLR     CCR                     ;CLEAR CCR BEFORE EXIT
17295 107310  013737  003012  000114          MOV     SLOC00, @#114           ;RESTORE CONTENTS OF VECTOR 114
17296 107316  032702  000003                  BIT     #3,     R2              ;IF ERROR INDICATORS SET
17297 107322  001401                          BEQ     7$                      ;THEN
17298 107324  104031                          ERROR   +31                     ;PARITY ERROR IGNORED
17299 107326  005037  177746          7$:     CLR     CCR                     ;CLEAR CCR
17300 107332  013737  003012  000114          MOV     SLOC00,@#114            ;RESTORE PARITY TRAP
17301 107340  052737  001000  177520          BIS     #1000,BCSR              ;ENABLE HALT ON BREAK
17302 107346  000404                          BR      TST14           ;;GO TO NEXT TEST
17303
17304
17305 107350  011637  001122          DAPABO: MOV     (SP),   $BDADR          ;SAVE ADDRESS THAT CAUSED ABORT
17306 107354  104007                          ERROR   +7                      ;ILLEGAL PARITY INTERRUPT
17307 107356  000002                          RTI
17308
```

C9

TEST - WRITE WRONG TAG PARITY

```
17310                                   .SBTTL  TEST    WRIT  WRONG TAG PARITY
17311                                   ;WRITE WRONG TAG PARITY   THIS TEST WILL VERIFY THAT A READ MISS
17312                                   ;OCCURS AFTER A WRITE WILL CCR<10> = 1 AND CCR<7,0> = 0,1. THE WRITE
17313                                   ;TO THE LOCATION WILL CAUSE A CACHE UPDATE BUT THE TAG WILL BE
17314                                   ;WRITTEN WITH THE WRONG PARITY. WHEN THE LOCATION IS READ INSTEAD
17315                                   ;OF A CACHE HIT OCCURRING THE PARITY ERROR SHOULD CAUSE A CACHE
17316                                   ;MISS.
17317                                   ;
17318                                   ;BGNTST
17319                                   ;SAVE CONTENTS OF VECTOR 114
17320                                   ;LET VECTOR 114 POINT TO ABORT ROUTINE
17321                                   ;CLEAR MSER
17322                                   ;SET WRITE WRONG TAG PARITY BIT<10>
17323                                   ;WRITE TEST LOCATION
17324                                   ;CLEAR CCR BIT<10> AND SET ABORT DISABLE BIT<0>
17325                                   ;READ TEST LOCATION
17326                                   ;IF BIT<1> SET IN HIT/MISS REGISTER THEN
17327                                   ;.      IF MSER BITS <7:5> SET THEN
17328                                   ;.              PARITY ERROR DOESN'T CAUSE MISS
17329                                   ;.      ELSE
17330                                   ;.              PARITY ERROR DOESN'T SET MSER WITH CCR<7>=0
17331                                   ;.      ENDIF
17332                                   ;ENDIF
17333                                   ;SET WRITE WRONG TAG PARITY BIT<10>
17334                                   ;WRITE TO A BYTE OF A TEST LOCATION
17335                                   ;SAVE HIT/MISS REGISTER
17336                                   ;CLEAR WRITE WRONG TAG PARITY BIT
17337                                   ;IF BIT<1> NOT SET IN HIT/MISS REGISTER THEN
17338                                   ;.      ERROR
17339                                   ;ENDIF
17340                                   ;RESTORE VECTOR 114
17341                                   ;EXIT   TST
17342                                   ;
17343                                   ;TAG PARITY ABORT ROUTINE:
17344                                   ;                               ILLEGAL PARITY INTERRUPT
17345                                   ;                               RETURN
17346                                   ;ENDTST
17347                                   ;;*****************************************************************
    107360  000004                    TST14:  SCOPE
17348 107362  000240                            NOP
17349 107364  005737  003032                    TST     CCHPAS                  ;have done enough inclusive passes?
17350 107370  001002                            BNE     99$                     ; not yet
17351 107372  000240                            NOP                             ; debug aid
17352 107374  000453                            BR      TST15           ;;GO TO NEXT TEST
17353 107376  000240                    99$:    NOP
17354 107400  013737  000114  003012            MOV     @#114,  SLOC00          ;SAVE CONTENTS OF VECTOR 114
17355 107406  012737  107514  000114            MOV     #TAPAB0,@#114           ;LET VECTOR POINT TO ABORT ROUTINE
17356 107414  042737  001000  177520            BIC     #1000,BCSR              ;DISABLE HALT ON BREAK
17357 107422  005037  177744                    CLR     MSER                    ;CLEAR MSER
17358 107426  012737  002000  177746            MOV     #BIT10, CCR             ;SET WRITE WRONG TAG PARITY
17359 107434  005037  003162                    CLR     @#TSTLOC                ;WRITE LOCATION WITH BAD TAG PARITY
17360 107440  012737  000001  177746            MOV     #BIT00, CCR             ;CLEAR BIT 10 AND SET BIT 0
17361 107446  005737  003162                    TST     @#TSTLOC                ;READ TEST LOCATION
17362 107452  013737  177752  114150            MOV     HITMIS,RECDAT           ;SAVE HIT/MISS REGISTER
17363 107460  032737  000004  114150            BIT     #BIT02,RECDAT           ;IF BIT 1 SET IN HIT/MISS REGISTER
17364 107466  001401                            BEQ     2$                      ;THEN
17365 107470  104027                            ERROR   .27                     ;PARITY ERROR DON'T CAUSE A MISS
```

D9

TEST - WRITE WRONG TAG PARITY

```
17366 107472  013737  003012  000114  2$:     MOV     SLOC00, @#114           ;RESTORE CONTENTS OF VECTOR 114
17367 107500  005037  177744                  CLR     MSER                    ;CLEAR ERRORS
17368 107504  052737  001000  177520          BIS     #1000,BCSR              ;ENABLE HALT ON BREAK
17369 107512  000404                          BR      TST15           ;;GO TO NEXT TEST
17370
17371
17372 107514  011637  001122          TAPABO: MOV     (SP),$BDADR             ;SAVE ADDRESS THAT CAUSE ABORT
17373 107520  104007                          ERROR   .7                      ;ILLEGAL PARITY INTERRUPT
17374 107522  000002                          RTI
17375
```

TEST - PARITY ABORT TEST

```
17377                                   .SBTTL  TEST    PARITY ABORT TEST
17378                                   ;PARITY ABORT TEST   THIS TEST WILL VERIFY THAT WHEN CCR<7,0> =
17379                                   ;1,0, AN ABORT OCCURS ON THE EXECUTION OF AN INSTRUCTION THAT HAS
17380                                   ;BEEN WRITTEN WITH WRONG PARITY.
17381                                   ;
17382                                   ;
17383                                   ;BGNTST
17384                                   ;SAVE VECTOR 114
17385                                   ;SAVE VECTOR 4
17386                                   ;LET VECTOR 114 POINT TO ABORT ROUTINE
17387                                   ;LET VECTOR 4 POINT TO ERROR ROUTINE
17388                                   ;CLEAR EXPECTING ABORT FLAG
17389                                   ;SET CCR<10> WRITE WRONG TAG PARITY BIT
17390                                   ;WRITE TEST ADDRESS
17391                                   ;CLEAR CCR<10>
17392                                   ;SET CCR TO #200 ENABLE PARITY ABORTS
17393                                   ;SET EXPECTING ABORT FLAG
17394                                   ;READ TEST ADDRESS
17395                                   ;IF ABORT FLAG NE 0 THEN
17396                                   ;.      ERROR IN PARITY ABORT LOGIC
17397                                   ;ENDIF
17398                                   ;RESTORE VECTOR 114
17399                                   ;RESTORE VECTOR 4
17400                                   ;EXIT   TST
17401                                   ;
17402                                   ;ABORT ROUTINE: IF EXPECTING ABORT FLAG NOT SET THEN
17403                                   ;                       ERROR NO ABORT SHOULD HAVE OCCURRED
17404                                   ;               ELSE
17405                                   ;                       CLEAR (EXPECTING) ABORT FLAG
17406                                   ;               ENDIF
17407                                   ;               IF MSER NOT EQUAL TO 100040 THEN
17408                                   ;                       PARITY ABORT LOGIC DOESN'T SET MSER PROPERLY
17409                                   ;               ENDIF
17410                                   ;               IF PC = UPDATED PC THEN
17411                                   ;                       ILLEGAL PARITY ABORT
17412                                   ;               ENDIF
17413                                   ;               RETURN
17414                                   ;
17415                                   ;ENDTST
17416                                   ;;*****************************************************************
      107524  000004                  TST15:  SCOPE
17417 107526  000240                          NOP
17418 107530  005737  003032                  TST     CCHPAS                  ;have done enough inclusive passes?
17419 107534  001002                          BNE     99$                     ; not yet
17420 107536  000240                          NOP                             ; debug aid
17421 107540  000504                          BR      TST16           ;;GO TO NEXT TEST
17422 107542  000240                  99$:    NOP
17423
17424 107544  013737  000114  003012          MOV     @#114,  SLOC00          ;SAVE VECTOR 114
17425 107552  013737  000004  003014          MOV     @#4,    SLOC01          ;SAVE VECTOR 4
17426 107560  012737  107676  000114          MOV     #ABORTR,@#114           ;LET VECTOR 114 POINT TO ABORT ROUTINE
17427                                   ;
17428                                   ; TO AVOID CONFUSION ALLOCATE IN CACHE FOLLOWING INSTRUCIONS
17429                                   ;
17430 107566  012704  107566                  MOV     #.,R4                   ;START WITH CURRENT
17431 107572  005724                  1$:     TST     (R4)+                   ;READ A WORD
17432 107574  022704  107676                  CMP     #ABORTR,R4              ;GOT TO ABORT ROUTINE?
```

F9

TEST - PARITY ABORT TEST

```
17433 107600  001374                        BNE     1$                      ;IF NOT, KEEP ON ALLOCATING
17434 107602  005000                        CLR     R0                      ;CLEAR EXPECTING ABORT FLAG
17435 107604  042737  001000  177520        BIC     #1000,BCSR              ;DISABLE HALT ON BREAK
17436 107612  012737  002000  177746        MOV     #BIT10, CCR             ;SET WRITE WRONG PARITY BIT
17437 107620  005037  003162                CLR     @#TSTLOC                ;WRITE TEST LOCATION WITH BAD PARITY
17438 107624  012737  000200  177746        MOV     #BIT07, CCR             ;ENABLE ABORTS, CLEAR WWTP BIT
17439 107632  005100                        COM     R0                      ;SET EXPECTING ABORT FLAG
17440 107634  005737  003162        ABORTI: TST     @#TSTLOC                ;READ TEST LOCATION (SHOULD CAUSE ABORT)
17441 107640  005700                        TST     R0                      ;IF ABORT FLAG NOT EQUAL ZERO
17442 107642  001401                        BEQ     1$                      ;THEN
17443 107644  104034                        ERROR   +34                     ;PARITY ABORT LOGIC DOESN T WORK
17444 107646  052737  001000  177520  1$:   BIS     #1000,BCSR              ;ENABLE HALT ON BREAK
17445 107654  013737  003012  000114        MOV     SLOC00, @#114           ;RESTORE VECTOR 114
17446 107662  013737  003014  000004        MOV     SLOC01, @#4             ;RESTORE VECTORE 4
17447 107670  005037  177744                CLR     MSER
17448 107674  000426                        BR      TST16           ;;GO TO NEXT TEST
17449
17450
17451 107676  013703  177744        ABORTR: MOV     MSER,R3                 ;SAVE MSER
17452 107702  005700                        TST     R0                      ;IF EXPECTING ABORT FLAG NOT SET
17453 107704  001004                        BNE     1$                      ;THEN
17454 107706  011637  001122                MOV     (SP),   $BDADR          ;SAVE ABORT ADDRESS
17455 107712  104007                        ERROR   +7                      ;ILLEGAL PARITY INTERRUPT
17456 107714  000401                        BR      2$                      ;ELSE
17457 107716  005000                  1$:   CLR     R0                      ;CLEAR (EXPECTING) ABORT FLAG
17458 107720  022737  100040  177744  2$:   CMP     #100040,MSER            ;IF MSER NOT EQUAL TO 100040
17459 107726  001404                        BEQ     3$                      ;THEN
17460 107730  012737  100040  001124        MOV     #100040,$GDDAT          ;SAVE PROPER MSER SETTING
17461 107736  104035                        ERROR   +35                     ;PARITY ABORT DON'T SET MSER PROPERLY
17462 107740  021627  107640          3$:   CMP     (SP),   #ABORTI+4       ;IF PC EQUAL TO UPDATE PC
17463 107744  001401                        BEQ     4$                      ;THEN
17464 107746  104007                        ERROR   +7                      ;ILLEGAL PARITY INTERRUPT
17465 107750  000002                  4$:   RTI                             ;RETURN
17466
```

TEST - PARITY INTERRUPT TEST

```
17468                                   .SBTTL  TEST  - PARITY INTERRUPT TEST
17469                                   ;PARITY INTERRUPT TEST   THIS TEST WILL VERIFY THAT WHEN CCR<7,0> =
17470                                   ;0,0, A PARITY INTERRRUPT OCCURS AFTER EXECUTION OF AN INSTRUCTION
17471                                   ;THAT HAS BEEN WRITTEN WITH WRONG PARITY.
17472                                   ;
17473                                   ;BGNTST
17474                                   ;SAVE CONTENTS OF 114
17475                                   ;SETUP VECTOR 114 TO POINT TO INTERRUPT ROUTINE
17476                                   ;CLEAR EXPECTING INTERRUPT FLAG
17477                                   ;WRITE TEST ADDRESS WITH BAD PARITY
17478                                   ;SET EXPECTING INTERRUPT FLAG
17479                                   ;READ TEST ADDRESS
17480                                   ;IF INTERRUPT FLAG NE 0 THEN
17481                                   ;.      PARITY INTERRUPT LOGIC DOESN'T WORK
17482                                   ;ENDIF
17483                                   ;RESTORE CONTENTS OF VECTOR 114
17484                                   ;EXIT   TST
17485                                   ;
17486                                   ;INTERRUPT ROUTINE:     IF EXPECTING INTERRUPT FLAG NE 1 THEN
17487                                   ;                       .       ERROR NO INTERRUPT SHOULD HAVE OCCURRED
17488                                   ;                       ELSE
17489                                   ;                       .       CLEAR (EXPECTING) INTERRUPT FLAG
17490                                   ;                       ENDIF
17491                                   ;                       IF SAVED PC NE TO UPDATED PC THEN
17492                                   ;                       .       IF PC = TEST INSTRUCTION PC THEN
17493                                   ;                       .       .       ERROR INSTRUCTION ABORTED
17494                                   ;                       .       ELSE
17495                                   ;                       .       .       ILLEGAL PARITY ABORT
17496                                   ;                       .       ENDIF
17497                                   ;                       ENDIF
17498                                   ;                       IF MSER NE #340 THEN
17499                                   ;                       .       PARITY INTERRUPT DOESN'T SET MSER PROPERLY
17500                                   ;                       ENDIF
17501                                   ;                       RETURN
17502                                   ;
17503                                   ;ENDTST
17504                                   ;;*****************************************************************
     107752  000004                   TST16:  SCOPE
17505 107754  000240                           NOP
17506 107756  005737  003032                   TST     CCHPAS                  ;have done enough inclusive passes?
17507 107762  001002                           BNE     99$                     ; not yet
17508 107764  000240                           NOP                             ; debug aid
17509 107766  000473                           BR      TST17           ;;GO TO NEXT TEST
17510 107770  000240                   99$:    NOP
17511
17512 107772  013737  000114  003012           MOV     @#114,  SLOC00          ;SAVE CONTENTS OF VECTOR 114
17513 110000  012737  110106  000114           MOV     #INTERR,@#114           ;LET VECTOR POINT TO INTERRUPT ROUTINE
17514                                   ;
17515                                   ; TO AVOID CONFUSION ALLOCATE IN CACHE FOLLOWING INSTRUCIONS
17516                                   ;
17517 110006  012704  110006           1$:     MOV     #.,R4                   ;START WITH CURRENT
17518 110012  005724                           TST     (R4)+                   ;READ A WORD
17519 110014  022704  110106                   CMP     #INTERR,R4              ;GOT TO INTERRUPT ROUTINE?
17520 110020  001374                           BNE     1$                      ;IF NOT, KEEP ON ALLOCATING
17521 110022  005001                           CLR     R1                      ;CLEAR EXPECTING INTERRUPT FLAG
17522 110024  042737  001000  177520           BIC     #1000,BCSR              ;DISABLE HALT ON BREAK
17523 110032  052737  000100  177746           BIS     #BIT06, CCR             ;SET WRITE WRONG DATA PARITY
```

H9

TEST - PARITY INTERRUPT TEST

```
17524 110040  005037  003162                 CLR     @#TSTLOC                ;WRITE LOCATION WITH BAD DATA PARITY
17525 110044  005037  177746                 CLR     CCR                     ;CLEAR WRITE WRONG DATA PARITY
17526 110050  005101                         COM     R1                      ;SET EXPECTING INTERRUPT FLAG
17527 110052  005737  003162         INTRPC: TST     @#TSTLOC                ;READ TEST LOCATION
17528 110056  005701                         TST     R1                      ;IF INTERRUPT FLAG NOT EQUAL ZERO
17529 110060  001401                         BEQ     1$                      ;THEN
17530 110062  104036                         ERROR   +36                     ;PARITY INTERRUPT LOGIC DOESN'T WORK
17531 110064  052737  001000  177520  1$:    BIS     #1000,BCSR              ;ENABLE HALT ON BREAK
17532 110072  013737  003012  000114         MOV     SLOC00, @#114           ;RESTORE VECTOR 114
17533 110100  005037  177744                 CLR     MSER                    ;CLEAR MSER
17534 110104  000424                         BR      TST17           ;;GO TO NEXT TEST
17535
17536
17537 110106  005701         INTERR: TST     R1                      ;IF EXPECTING INTERRUPT FLAG NOT SET
17538 110110  001004                         BNE     1$                      ;THEN
17539 110112  011637  001122                 MOV     (SP),   $BDADR          ;SAVE INTERRUPT ADDRESS
17540 110116  104007                         ERROR   +7                      ;ILLEGAL PARITY INTERRUPT
17541 110120  000401                         BR      2$                      ;ELSE
17542 110122  005001                 1$:     CLR     R1                      ;CLEAR (EXPECTING) INTERRUPT FLAG
17543 110124  021627  110056         2$:     CMP     (SP),   #INTRPC+4       ;IF SAVED PC NOT EQUAL TO UPDATED PC
17544 110130  001401                         BEQ     4$                      ;THEN
17545 110132  104007                         ERROR   +7                      ;ILLEGAL PARITY INTERRUPT
17546 110134  022737  000340  177744  4$:    CMP     #340,   MSER            ;IF MSER NOT EQUAL TO EXPECTED VALUE
17547 110142  001404                         BEQ     5$                      ;THEN
17548 110144  012737  000340  001124         MOV     #340,   $GDDAT          ;SAVE PROPER MSER
17549 110152  104035                         ERROR   +35                     ;PARITY INTERRUPT DON'T SET MSER PROPERLY
17550 110154  000002                 5$:     RTI                             ;RETURN
17551
```

```
17553                                   .SBTTL  TEST    - MISCELLANEOUS PARITY TEST
17554                                   ;MISCELLANEOUS PARITY TEST - THIS TEST CHECKS THAT BYPASS CYCLES WITH
17555                                   ;PARITY ERRORS CAUSE CACHE HIT RESPONSE AND THAT FORCE MISS CYCLES
17556                                   ;IGNORE PARITY ERRORS.
17557                                   ;
17558                                   ;BGNTST
17559                                   ;WRITE A LOCATION WITH BAD PARITY
17560                                   ;SET BYPASS IN CCR
17561                                   ;READ THE LOCATION BACK
17562                                   ;IF NO HIT OR MSER NOT SET THEN
17563                                   ;.      ERROR
17564                                   ;ENDIF
17565                                   ;WRITE A LOCATION WITH BAD PARITY AND SET BYPASS
17566                                   ;IF MSER SET WHILE READING IT BACK
17567                                   ;.      ERROR
17568                                   ;ENDIF
17569                                   ;ENDTST
17570                                   ;;*****************************************************************
    110156  000004                      TST17:  SCOPE
17571 110160  000240                          NOP
17572 110162  005737  003032                   TST     CCHPAS                  ;have done enough inclus've passes?
17573 110166  001002                          BNE     99$                     ; not yet
17574 110170  000240                          NOP                             ; debug aid
17575 110172  000503                          BR      TST20           ;;GO TO NEXT TEST
17576 110174  000240                  99$:    NOP
17577
17578 110176  012703  110176                  MOV     #.,R3                           ;START WITH CURRENT INSTRUCTION
17579 110202  005723                  10$:    TST     (R3)+                           ;READ A WORD
17580 110204  022703  110354                  CMP     #4$,R3                          ;LAST WORD?
17581 110210  001374                          BNE     10$                             ;IF NOT, CONTINUE
17582                                   ;
17583                                   ; CHECK BYPASS AND BAD PARITY
17584                                   ;
17585 110212  013737  000114  003012          MOV     @#114,  SLOC00                  ;SAVE PARITY VECTOR
17586 110220  012737  110264  000114          MOV     #1$,@#114                       ;POINT NEW VECTOR
17587 110226  042737  001000  177520          BIC     #1000,BCSR                      ;DISABLE HALT ON BREAK
17588 110234  052737  002101  177746          BIS     #BIT10!BIT06!BIT00,CCR          ;DATA AND TAG PAR., NO INT.
17589 110242  005037  003162                  CLR     @#TSTLOC                        ;WRITE CYCLE
17590 110246  012737  001000  177746          MOV     #BIT09,CCR                      ;SET BYPASS
17591 110254  005737  003162                  TST     @#TSTLOC                        ;BYPASS WITH WRONG PARITY
17592 110260  104045                          ERROR   .45                             ;ERROR
17593 110262  000407                          BR      2$                              ;
17594 110264  062706  000004          1$:     ADD     #4,SP                           ;ADJUST STACK
17595 110270  032737  000340  177744          BIT     #340,MSER                       ;MSER OK?
17596 110276  001001                          BNE     2$                              ;IF YES, BRANCH
17597 110300  104042                          ERROR   .42                             ;BYPASS WRONG
17598                                   ;
17599                                   ; CHECK FORCE MISS AND BAD PARITY
17600                                   ;
17601 110302  005037  177744          2$:     CLR     MSER                            ;CLEAR MSER
17602 110306  005037  177746                  CLR     CCR                             ;CLEAR CCR
17603 110312  012737  110346  000114          MOV     #3$,@#114                       ;POINT NEW VECTOR
17604 110320  052737  002101  177746          BIS     #BIT10!BIT06!BIT00,CCR          ;DATA AND TAG PAR., NO INTER
17605 110326  005037  003162                  CLR     @#TSTLOC                        ;FORCE MISS WITH PARITY
17606 110332  012737  000014  177746          MOV     #14,CCR                         ;FORCE MISS
17607 110340  005737  003162                  TST     @#TSTLOC                        ;ALLOCATE CACHE
17608 110344  000403                          BR      4$
```

J9

SEQ 0320

TEST MISCELLANEOUS PARITY TEST

```
 17609 110346  104042                  3$:     ERROR   .42
 17610 110350  062706  000004                  ADD     #4,SP                   ;ADJUST STACK
 17611 110354  005037  177746            4$:   CLR     CCR                     ;CLEAR CCR
 17612 110360  052737  001000  177520          BIS     #1000,BCSR              ;ENABLE HALT ON BREAK
 17613 110366  013737  003012  000114          MOV     SLOC00,@#114            ;RESTORE PARITY VECTOR
 17614 110374  012737  000400  177746          MOV     #BIT08,CCR              ;FLUSH CACHE
 17615
```

```
                        .SBTTL  TEST    MEMORY SYSTEM ERROR REGISTER TEST
                        ;MEMORY SYSTEM ERROR REGISTER TEST   THIS TEST WILL VERIFY THE
                        ;FUNCTIONALITY OF BITS <15> AND <7:5> OF THE MEMORY SYSTEM ERROR
                        ;REGISTER. THIS TEST WILL USE THE WRITE WRONG PARITY BITS (BITS<10>
                        ;AND <6> OF THE CCR) TO WRITE BAD PARITY INTO A LOCATION. THE
                        ;LOCATION WILL THEN BE READ WITH CACHE TRAPS ENABLED AND THE MSER
                        ;WILL BE CHECKED AFTER THE ABORT FOR THE CORRECT BIT(S) BEING SET.
                        ;THIS WILL BE DONE FOR ALL COMBINATIONS OF BITS<10> AND <6> FOR WORD
                        ;ACCESSES THEN REPEATED FOR ALL COMBINATIONS OF BITS<10> AND <6> FOR
                        ;BYTE ACCESSES. THE TEST WILL THEN BE REPEATED A THIRD TIME FOR BYTE
                        ;ACCESSES AND ABORTS DISABLED. THE MSER SHOULD CONTAIN BITS <7:5>
                        ;SET TO 1 S AND BIT <15> A ZERO FOR ALL COMBINATIONS.
                        ;
                        ;BGNTST
                        ;SAVE CONTENTS OF VECTOR 114
                        ;SETUP VECTOR TO POINT TO ABORT ROUTINE
                        ;INITIALIZE LOOP COUNTER
                        ;DO UNTIL ALL WORD COMBINATIONS CHECKED
                        ;.      INITIALIZE MSER
                        ;.      SETUP CCR FROM CCR TABLE
                        ;.      READ TEST LOCATION TO ALLOCATE LOCATION WITH DESIRED PARITY
                        ;.      CLEAR <10,6> FROM CCR
                        ;.      SETUP CCR FOR ABORT
                        ;.      READ TEST LOCATION                      ;THIS COULD CAUSE TRAP
                        ;.      IF EXPECTED DATA NE RECEIVED DATA THEN
                        ;.              ERROR IN SETTING MSER
                        ;.      ENDIF
                        ;.      UPDATE LOOP COUNTER
                        ;ENDDO
                        ;INITIALIZE LOOP COUNTER
                        ;DO UNTIL ALL BYTE COMBINATIONS CHECKED
                        ;.      INITIALIZE MSER
                        ;.      SETUP CCR FROM CCR TABLE
                        ;.      READ TEST LOCATION TO ALLOCATE LOCATION WITH DESIRED PARITY
                        ;.      CLEAR <10,6> FROM CCR
                        ;.      SETUP CCR FOR ABORT
                        ;.      READ LOW BYTE TEST LOCATION
                        ;.      GET EXPECTED BYTE DATA FROM TABLE
                        ;.      IF EXPECTED DATA NE RECEIVED DATA THEN
                        ;.              ERROR IN SETTING MSER
                        ;.      ENDIF
                        ;.      INITIALIZE MSER
                        ;.      READ HIGH BYTE TEST LOCATION
                        ;.      GET EXPECTED BYTE DATA FROM TABLE
                        ;.      IF EXPECTED DATA NE RECEIVED DATA THEN
                        ;.              ERROR IN SETTING MSER
                        ;.      ENDIF
                        ;.      INCREMENT LOOP COUNTER
                        ;ENDDO
                        ;INITIALIZE LOOP COUNTER
                        ;DO UNTIL ALL BYTE COMBINATIONS CHECKED
                        ;.      INITIALIZE MSER
                        ;.      SETUP CCR FROM CCR TABLE
                        ;       READ TEST LOCATION TO ALLOCATE LOCATION WITH DESIRED PARITY
                        ;.      CLEAR <10,6> FROM CCR
                        ;.      SETUP CCR FOR NO ABORTS
                        ;.      READ LOW BYTE TEST LOCATION
```

```
17674                                   ;.      IF RECEIVED DATA NE TO #340 THEN
17675                                   ;.              ERROR IN SETTING MSER
17676                                   ;.      ENDIF
17677                                   ;.      INITIALIZE MSER
17678                                   ;.      READ HIGH BYTE TEST LOCATION
17679                                   ;.      IF RECEIVED DATA NE #340 THEN
17680                                   ;.              ERROR IN SETTING MSER
17681                                   ;.      ENDIF
17682                                   ;.      INCREMENT LOOP COUNTER
17683                                   ;ENDDO
17684                                   ;EXIT   TST
17685                                   ;
17686                                   ;CCR TABLE:     0
17687                                   ;               100
17688                                   ;               2000
17689                                   ;               2100
17690                                   ;EXPECTED WORD DATA:    0
17691                                   ;                       100300
17692                                   ;                       100040
17693                                   ;                       10034C
17694                                   ;EXPECTED BYTE DATA:    0
17695                                   ;                       0
17696                                   ;                       100100
17697                                   ;                       100200
17698                                   ;                       100040
17699                                   ;                       100040
17700                                   ;                       100140
17701                                   ;                       100240
17702                                   ;ABORT ROUTINE: RTI
17703                                   ;
17704                                   ;ENDTST
17705                                   ;;*****************************************************************
      110402  000004                    TST20:  SCOPE
17706 110404  000240                            NOP
17707 110406  005737  003032                    TST     CCMPAS                  ;have done enough inclusive passes?
17708 110412  001003                            BNE     99$                     ; not yet
17709 110414  000240                            NOP                             ; debug aid
17710 110416  000137  111104                    JMP     10$                     ; yes skip this
17711 110422  000240                    99$:    NOP
17712
17713 110424  013737  000114  003012            MOV     @#114,  SLOC00          ;SAVE CONTENTS OF VECTOR 114
17714 110432  012737  111146  000114            MOV     #ABROUT,@#114           ;SETUP VECTOR TO POINT TO ABORT ROUTINE
17715 110440  012704  000004                    MOV     #4,     R4              ;INITIALIZE LOOP COUNTER
17716 110444  012700  111106                    MOV     #CCRTBL,R0              ;GET ADDRESS OF CCR TABLE
17717 110450  012701  111116                    MOV     #EXPWDT,R1              ;GET ADDRESS OF EXPECTED DATA TABLE
17718 110454  042737  001000  177520            BIC     #1000,BCSR              ;DISABLE HALT ON BREAK
17719 110462  005037  177744            1$:     CLR     MSER                    ;INITIALIZE MSER DATA
17720 110466  012037  177746                    MOV     (R0)+,  CCR             ;SETUP CCR FROM CCR TABLE
17721 110472  005037  003162                    CLR     @#TSTLOC                ;ALLOCATE CACHE LOC. WITH DESIRED PARITY
17722 110476  012737  000200  177746            MOV     #BIT07, CCR             ;SETUP CCR TO ABORT POSSIBLE BAD PARITY
17723 110504  005737  003162                    TST     @#TSTLOC                ;READ TEST LOCATION
17724 110510  021137  177744                    CMP     (R1),   MSER            ;IF RECEIVED DATA NOT EQUAL TO EXPECTED
17725 110514  001403                            BEQ     2$                      ;DATA THEN
17726 110516  011137  001124                    MOV     (R1),   $GDDAT          ;SAVE PROPER MSER SETTING
17727 110522  104035                            ERROR   #35                     ;MSER NOT SET PROPERLY
17728 110524  005721                    2$:     TST     (R1)+                   ;INCREMENT POINTER THRU MSER TABLE
17729 110526  005737  023162                    TST     @#TSTLOC+8192.          ;TO INSURE MISS ON THE NEXT LOOP
```

```
17730 110532 077425                        SOB     R4,     1$              ;LOOP UNTIL ALL COMBINATIONS CHECKED
17731 110534 052737 001000 177520          BIS     #1000,BCSR              ;ENABLE HALT ON BREAK
17732
17733                              ;CHECK BYTE OPERATIONS NOW
17734 110542 012704 000004                 MOV     #4,     R4              ;INITIALIZE LOOP COUNTER
17735 110546 012700 111106                 MOV     #CCRTBL,R0              ;GET ADDRESS OF CCR TABLE
17736 110552 012701 111126                 MOV     #EXPBDT,R1              ;GET ADDRESS OF EXPECTED BYTE DATA TABLE
17737 110556 042737 001000 177520          BIC     #1000,BCSR              ;DISABLE HALT ON BREAK
17738 110564 005037 177744           3$:   CLR     MSER                    ;INITIALIZE MSER
17739 110570 005737 003162                 TST     @#TSTLOC                ;ALLOCATE TO HAVE WRITE BYTE HIT
17740 110574 011037 177746                 MOV     (R0),   CCR             ;SETUP CCR FROM TABLE
17741 110600 105037 003162                 CLRB    @#TSTLOC                ;ALLOCATE CACHE LOC. WITH DESIRED PARITY
17742 110604 012737 000200 177746          MOV     #BIT07, CCR             ;SETUP CCR TO ABORT POSSIBLE BAD PARITY
17743 110612 105737 003162                 TSTB    @#TSTLOC                ;READ LOW BYTE OF TEST LOCATION
17744 110616 021137 177744                 CMP     (R1),   MSER            ;IF RECEIVED DATA NOT EQUAL TO EXPECTED
17745 110622 001403                        BEQ     4$                      ;THEN
17746 110624 011137 001124                 MOV     (R1),   $GDDAT          ;SAVE PROPER MSER SETTING
17747 110630 104035                        ERROR   ;35                     ;MSER NOT SET PROPERLY
17748 110632 005721                  4$:   TST     (R1)+                   ;INCREMENT POINTER THRU MSER TABLE
17749 110634 005037 177744                 CLR     MSER                    ;INITIALIZE MSER
17750 110640 005737 003162                 TST     @#TSTLOC                ;ALLOCATE TO HAVE WRITE BYTE HIT
17751 110644 012037 177746                 MOV     (R0)+,  CCR             ;SETUP CCR FROM TABLE
17752 110650 105037 003163                 CLRB    @#TSTLOC+1              ;ALLOCATE CACHE LOC. WITH DESIRED PARITY
17753 110654 012737 000200 177746          MOV     #BIT07, CCR             ;SETUP CCR TO ABORT POSSIBLE BAD PARITY
17754 110662 105737 003163                 TSTB    @#TSTLOC+1              ;READ HIGH BYTE OF TEST LOCATION
17755 110666 021137 177744                 CMP     (R1),   MSER            ;IF RECEIVED DATA NOT EQUAL TO EXPECTED
17756 110672 001403                        BEQ     5$                      ;THEN
17757 110674 011137 001124                 MOV     (R1),   $GDDAT          ;SAVE PROPER MSER SETTING
17758 110700 104035                        ERROR   ;35                     ;MSER NOT SET PROPERLY
17759 110702 005721                  5$:   TST     (R1)+                   ;INCREMENT POINTER THRU MSER TABLE
17760 110704 077451                        SOB     R4,     3$              ;DO UNTIL ALL BYTE COMBINATIONS CHECKED
17761 110706 052737 001000 177520          BIS     #1000,BCSR              ;ENABLE HALT ON BREAK
17762                              ;REPEAT WITHOUT ABORT
17763 110714 012704 000003                 MOV     #3,     R4              ;INITIALIZE LOOP COUNTER
17764 110720 012700 111110                 MOV     #CCRTBL+2,R0            ;GET ADDRESS OF CCR TABLE
17765 110724 042737 001000 177520          BIC     #1000,BCSR              ;DISABLE HALT ON BREAK
17766 110732 005037 177744           6$:   CLR     MSER                    ;INITIALIZE MSER
17767 110736 005737 003162                 TST     @#TSTLOC                ;ALLOCATE CACHE LOC.
17768 110742 011037 177746                 MOV     (R0),   CCR             ;SETUP CCR FROM TABLE
17769 110746 105037 003162                 CLRB    @#TSTLOC                ;ALLOCATE CACHE LOC. WITH DESIRED PARITY
17770 110752 012737 000001 177746          MOV     #BIT00, CCR             ;SETUP CCR TO NOT ABORT
17771 110760 105737 003162                 TSTB    @#TSTLOC                ;READ LOW BYTE OF TEST LOCATION
17772 110764 022737 000340 177744          CMP     #340,   MSER            ;IF RECEIVED DATA NOT EQUAL TO EXPECTED
17773 110772 001404                        BEQ     7$                      ;THEN
17774 110774 012737 000340 001124          MOV     #340,   $GDDAT          ;SAVE PROPER MSER SETTING
17775 111002 104035                        ERROR   ;35                     ;MSER NOT SET PROPERLY
17776 111004 005037 177744           7$:   CLR     MSER                    ;INITIALIZE MSER
17777 111010 005737 003162                 TST     @#TSTLOC                ;ALLOCATE CACHE LOC.
17778 111014 012037 177746                 MOV     (R0)+,  CCR             ;SETUP CCR FROM TABLE
17779 111020 105037 003163                 CLRB    @#TSTLOC+1              ;ALLOCATE CACHE LOC. WITH DESIRED PARITY
17780 111024 012737 000001 177746          MOV     #BIT00, CCR             ;SETUP CCR TO NOT ABORT
17781 111032 105737 003163                 TSTB    @#TSTLOC+1              ;READ HIGH BYTE OF TEST LOCATION
17782 111036 022737 000340 177744          CMP     #340,   MSER            ;IF RECEIVED DATA NOT EQUAL TO EXPECTED
17783 111044 001404                        BEQ     8$                      ;THEN
17784 111046 012737 000340 001124          MOV     #340,   $GDDAT          ;SAVE PROPER MSER SETTING
17785 111054 104035                        ERROR   ;35                     ;MSER NOT SET PROPERLY
17786 111056 077453                  3$:   SOB     R4,     6$              ;DO UNTIL ALL BYTE COMBINATIONS CHECKED
```

N9

TEST MEMORY SYSTEM ERROR REGISTER TEST

```
17787 111060  005037  177744                CLR     MSER                    ;CLEAR ERROR REGISTER
17788 111064  052737  001000  177520        BIS     #1000,BCSR              ;ENABLE HALT ON BREAK
17789 111072  013737  003012  000114        MOV     SLOC00, @#114           ;RESTORE VECTOR 114
17790 111100  005037  177746                CLR     CCR                     ;CLEAR ALL BIT IN CCR
17791 111104                        10$:
      111104  000421                        BR      TST21           ;;GO TO NEXT TEST
17792
17793
17794 111106  000000                CCRTBL: .WORD   0
17795 111110  000100                        .WORD   100
17796 111112  002000                        .WORD   2000
17797 111114  002100                        .WORD   2100
17798
17799 111116  000000                EXPWDT: .WORD   0
17800 111120  100300                        .WORD   100300
17801 111122  100040                        .WORD   100040
17802 111124  100340                        .WORD   100340
17803
17804 111126  000000                EXPBDT: .WORD   0
17805 111130  000000                        .WORD   0
17806 111132  100100                        .WORD   100100
17807 111134  100200                        .WORD   100200
17808 111136  100040                        .WORD   100040
17809 111140  100040                        .WORD   100040
17810 111142  100140                        .WORD   100140
17811 111144  100240                        .WORD   100240
17812
17813 111146  000002                ABROUT: RTI
17814
17815
```

```
17817                                   .SBTTL  TEST    CHECK PARITY ABORTS BLOCKED BY NON EXISTENT MEMORY ABORT
17818                                   ;CHECK PARITY ABORTS BLOCKED BY NON EXISTENT MEMORY ABORT   THIS TEST WILL
17819                                   ;VERIFY THAT IF A PARITY ERROR OCCURS ON THE SAME ADDRESS REFERENCE AS A
17820                                   ;NON-EXISTENT MEMORY ERROR THAT THE CACHE DATA PATH GATE ARRAY BLOCKS THE
17821                                   ;PARITY ERROR TO THE J 11 CHIP SET. THIS WILL BE DONE BY USING THE DIAGNOSTIC
17822                                   ;BIT TO CAUSE A PARITY ERROR IN A CACHE REFERENCE THAT DOES NOT HAVE A
17823                                   ;CORRESPONDING ADDRESS IN MAIN MEMORY. THE ADDRESS WILL THEN BE READ CAUSING
17824                                   ;BOTH A CACHE PARITY ERROR AND A NON EXISTENT MEMORY ERROR. TO AVOID HAVING
17825                                   ;TO SIZE THE ENTIRE MEMORY TO FIND A NON-EXISTENT MEMORY ADDRESS THIS TEST
17826                                   ;WILL USE THE LARGEST NON I/O ADDRESS (17757776). THE TEST WILL FIRST READ
17827                                   ;THIS ADDRESS AND IF A NXM TRAP OCCURS THE TEST WILL BE DONE  IF THE ACCESS
17828                                   ;TO THIS ADDRESS DOES NOT TRAP THEN THE TEST WILL BE SKIPPED.
17829                                   ;
17830                                   ;BGNTST
17831                                   ;SAVE CONTENTS OF VECTOR 4
17832                                   ;SAVE CONTENTS OF VECTOR 114
17833                                   ;LET VECTOR 4 POINT TO CONTINUE TESTING (A:)
17834                                   ;LET PAR6 = #177400
17835                                   ;ACCESS ADDRESS 157776 (PHYSICAL 17757776)
17836                                   ;IF NO TRAP THEN
17837                                   ;.      GOTO ENDTST
17838                                   ;ENDIF
17839                                   ;A:
17840                                   ;SET DIAGNOSTIC AND WRITE WRONG PARITY BITS IN CCR
17841                                   ;WRITE ADDRESS 157776
17842                                   ;CLEAR CCR
17843                                   ;SET PARITY ERROR ABORT BIT IN CCR
17844                                   ;LET VECTOR 4 POINT TO CONTINUE TESTING (B:)
17845                                   ;LET VECTOR 114 POINT TO NXM-PARITY ERROR ROUTINE
17846                                   ;READ ADDRESS 157776
17847                                   ;IF NO TRAP THEN
17848                                   ;.      ERROR IN ABORT LOGIC
17849                                   ;ENDIF
17850                                   ;B:
17851                                   ;CLEAR CCR
17852                                   ;RESTORE CONTENTS OF VECTOR 114
17853                                   ;RESTORE CONTENTS OF VECTOR 4
17854                                   ;EXIT   TST
17855                                   ;
17856                                   ;
17857                                   ;NXM-PARITY ERROR ROUTINE:      RESET STACK AFTER TRAP
17858                                   ;                                       PARITY ABORT NOT BLOCKED BY NXM
17859                                   ;                               GOTO B:
17860                                   ;ENDTST
17861                                   ;;*****************************************************************
    111150  000004                      TST21:  SCOPE
17862 111152  000240                              NOP
17863 111154  005737  003032                      TST     CCHPAS                  ;have done enough inclus've passes?
17864 111160  001002                              BNE     99$                     ; not yet
17865 111162  000240                              NOP                             ; debug aid
17866 111164  000475                              BR      TST22           ;;GO TO NEXT TEST
17867 111166  000240                      99$:    NOP
17868
17869 111170  013737  000004  003012              MOV     @#4,    SLOC00          ;SAVE CONTENTS OF VECTOR 4
17870 111176  013737  000114  003014              MOV     @#114,  SLOC01          ;SAVE CONTENTS OF VECTOR 114
17871 111204  012737  111232  000004              MOV     #1$,    @#4             ;LET VECTOR 4 POINT TO CONTINUE TESTING
17872 111212  012737  177400  172354              MOV     #177400,KIPAR6          ;LET PAR6 = OFFSET TO HIGHEST MEMORY
```

```
TEST  - CHECK PARITY ABORTS BLOCKED BY NON EXISTENT MEMORY ABO

 17873 111220  005237  177572            INC     SR0                     ;TURN ON MMU
 17874 111224  005737  157776            TST     @#157776                ;ACCESS ADDRESS 17757776
 17875 111230  000431                    BR      ABOEXT                  ;IF NO TRAP SKIP TEST
 17876 111232  062706  000004      1$:   ADD     #4,     SP              ;RESET STACK AFTER TRAP
 17877 111236  042737  001000  177520    BIC     #1000,BCSR              ;DISABLE HALT ON BREAK
 17878 111244  012737  000102  177746    MOV     #102,   CCR             ;SET DIAG. AND WRITE WRONG PARITY BITS
 17879 111252  005037  157776            CLR     @#157776                ;WRITE TO ADDRESS 17757776
 17880 111256  012737  000200  177746    MOV     #200,   CCR             ;CLEAR CCR AND SET PARITY ABORT BIT
 17881 111264  012737  111310  000004    MOV     #2$,    @#4             ;LET VECTOR 4 POINT TO CONTINUE TESTING
 17882 111272  012737  111350  000114    MOV     #NXMPAR,@#114           ;LET VECTOR 114 POINT TO ERROR ROUTINE
 17883 111300  005737  157776            TST     @#157776                ;READ ADDRESS 17757776 (SHOULD TRAP)
 17884 111304  104037                    ERROR   +37                     ;NXM AND PARITY ABORT DIN'T HAPPEN
 17885 111306  000402                    BR      ABOEXT                  ;GO EXIT TEST
 17886 111310  062706  000004      2$:   ADD     #4,     SP              ;RESET STACK AFTER TRAP
 17887 111314  005037  177746    ABOEXT: CLR     CCR                     ;CLEAR CCR FOR EXIT
 17888 111320  005037  177572            CLR     SR0                     ;DISABLE MMU
 17889 111324  052737  001000  177520    BIS     #1000,BCSR              ;ENABLE HALT ON BREAK
 17890 111332  013737  003012  000004    MOV     SLOC00, @#4             ;RESTORE VECTOR 4
 17891 111340  013737  003014  000114    MOV     SLOC01, @#114           ;RESTORE VECTOR 114
 17892 111346  000404                    BR      TST22           ;;GO TO NEXT TEST
 17893
 17894
 17895
 17896 111350  062706  000004    NXMPAR: ADD     #4,     SP              ;RESET STACK AFTER TRAP
 17897 111354  104040                    ERROR   +40                     ;PARITY ABORT NOT BLOCKED BY NXM TRAP
 17898 111356  000002                    RTI
 17899
```

TEST - MULTIPROCCESSING INSTRUCTION TESTS

```
17901                                   .SBTTL  TEST    MULTIPROCCESSING INSTRUCTION TESTS
17902                                   ;MULTIPROCCESSING INSTRUCTION TESTS - THIS TEST WILL VERIFY THAT THE MULTI-
17903                                   ;PROCCESSING INSTRUCTIONS DO A BYPASS OF THE CACHE. THIS TEST WILL NOT VERIFY
17904                                   ;THE REST OF THE FUNCTIONALITY OF THESE INSTRUCTIONS BECAUSE THAT WILL ALREADY
17905                                   ;HAVE BEEN CHECKED IN THE BASE INSTRUCTION TESTS. THE TEST WILL FIRST ALLOCATE
17906                                   ;AN ADDRESS IN CACHE THEN A TSTSET INSTRUCTION WILL BE DONE AT THAT ADDRESS.
17907                                   ;A HIT SHOULD BE RECORDED ON THE ACCESS. NEXT, THE ADDRESS WILL BE READ AND
17908                                   ;A MISS SHOULD BE RECORDED BECAUSE THE FORCED BYPASS ON THE TSTSET INSTRUCTION
17909                                   ;SHOULD HAVE INVALIDATED THE CACHE ENTRY. THE SAME SEQUENCE WILL THEN BE
17910                                   ;REPEATED FOR THE WRTLCK INSTRUCTION.
17911                                   ;
17912                                   ;BGNTST
17913                                   ;READ TEST LOCATION TO ALLOCATE CACHE
17914                                   ;DO TSTSET INSTRUCTION
17915                                   ;IF HIT/MISS REGISTER BIT 3 NOT SET OR BIT 2 SET THEN
17916                                   ;.      ERROR IN MULTI-PROCESSOR HOOKS
17917                                   ;ENDIF
17918                                   ;READ TEST LOCATION (ALSO ALLOCATES CACHE FOR WRTLCK)
17919                                   ;DO WRTLCK INSTRUCTION
17920                                   ;IF HIT/MISS REGISTER BIT 3 NOT SET OR BIT 2 SET THEN
17921                                   ;.      ERROR IN MULTI PROCESSOR HOOKS
17922                                   ;ENDIF
17923                                   ;READ TEST LOCATION
17924                                   ;DO ASRB INSTRUCTION
17925                                   ;IF HIT/MISS REGISTER BIT 3 NOT SET OR BIT 2 SET THEN
17926                                   ;.      ERROR IN MULTI-PROCESSOR HOOKS
17927                                   ;ENDIF
17928                                   ;ENDTST
17929                                   ;NOTE: THE CODE IS POSITION DEPENDENT
17930
17931                                   ;;*****************************************************************
    111360  000004                      TST22:  SCOPE
17932 111362  000240                          NOP
17933 111364  005737  003032                   TST     CCHPAS                  ;have done enough inclusive passes?
17934 111370  001002                          BNE     99$                     ; not yet
17935 111372  000240                          NOP                             ; debug aid
17936 111374  000501                          BR      TST23           ;;GO TO NEXT TEST
17937 111376  000240                  99$:    NOP
17938
17939 111400  042737  001000  177520          BIC     #1000,BCSR              ;DISABLE HALT ON BREAK
17940 111406  005737  003162                   TST     TSTLOC                  ;READ TEST LOCATION TO ALLOCATE CACHE
17941                                  ;       TSTSET  TSTLOC                  ;DO TSTSET INSTRUCTION
17942 111412  007237                  7$:     .WORD   7237                    ;THESE NEXT TWO LOCATIONS ARE THE TSTSET
17943 111414  003162                          .WORD   TSTLOC                  ;INSTR. BECAUSE THE ASSEMBLER WAS NOT READY
17944 111416  013737  177752  114150          MOV     HITMIS,RECDAT           ;STORE REGISTER
17945 111424  032737  000010  114150          BIT     #BIT3,RECDAT            ;IF HIT/MISS REGISTER BIT 3 NOT SET
17946 111432  001001                          BNE     1$                      ;THEN
17947 111434  104045                          ERROR   .45                     ;ERROR IN RECORDING HITS IN HIT/MISS
17948 111436  032737  000004  114150  1$:     BIT     #BIT2,RECDAT            ;IF HIT/MISS REGISTER BIT 2 SET
17949 111444  001404                          BEQ     2$                      ;THEN
17950 111446  013737  111412  001126          MOV     @#7$,   $BDDAT          ;SAVE OPCODE
17951 111454  104041                          ERROR   .41                     ;MULTI-PROCESSOR HOOK INSTRUCTION DOESN TT CAUSE MISS
17952                                  ;2$:    WRTLCK  TSTLOC                  ;DO WRTLCK INSTRUCTION
17953 111456  000240                  2$:     NOP                             ;PUT THE WRITE LOCK CODE ON THE RIGHT $$$
17954 111460  000240                          NOP                             ;BOUNDRY. $$$
17955 111462  007337                          .WORD   7337                    ;THESE NEXT TWO WORDS ARE THE WRTLCK
17956 111464  003162                          .WORD   TSTLOC                  ;INSTR. BECAUSE THE ASSEMBLER WAS NOT READY
```

E10

TEST - MULTIPROCCESSING INSTRUCTION TESTS

```
 17957 111466  013737  177752  114150          MOV     HITMIS,RECDAT           ;STORE REGISTER
 17958 111474  032737  000010  114150          BIT     #BIT3,RECDAT            ;IF HIT/MISS REGISTER BIT 3 NOT SET
 17959 111502  001001                          BNE     3$                      ;THEN
 17960 111504  104045                          ERROR   .45                     ;ERROR IN RECORDING HITS IN HIT/MISS
 17961 111506  032737  000004  114150  3$:     BIT     #BIT2,RECDAT            ;IF HIT/MISS REGISTER BIT 2 SET
 17962 111514  001404                          BEQ     4$                      ;THEN
 17963 111516  013737  111456  001126          MOV     @#2$,   $BDDAT          ;SAVE OPCODE
 17964 111524  104041                          ERROR   .41                     ;MULTI PROCESSOR HOOK INSTRUCTION DOESN TT CAUSE MISS
 17965 111526  106237  003162          4$:     ASRB    TSTLOC                  ;THE THIRD BYPASS INSTRUCTION
 17966 111532  013737  177752  114150          MOV     HITMIS,RECDAT           ;STORE REGISTER
 17967 111540  032737  000010  114150          BIT     #BIT3,RECDAT            ;IF HIT/MISS REGISTER BIT 3 NOT SET
 17968 111546  001001                          BNE     5$                      ;THEN
 17969 111550  104045                          ERROR   .45                     ;ERROR IN RECORDING HITS IN HIT/MISS
 17970 111552  052737  001000  177520  5$:     BIS     #1000,BCSR              ;ENABLE HALT ON BREAK
 17971 111560  032737  000004  114150          BIT     #BIT2,RECDAT            ;IF HIT/MISS REGISTER BIT 2 SET
 17972 111566  001404                          BEQ     TST23           ;;EXIT TEST
 17973 111570  013737  111526  001126          MOV     @#4$,   $BDDAT          ;SAVE OPCODE
 17974 111576  104041                          ERROR   .41                     ;MULTI PROCESSOR HOOK INSTRUCTION DOESN TT CAUSE MISS
 17975
 17976
 17977
```

F10


```
17979                                   .SBTTL  TEST  - DATA STORE RAM TESTS
17980                                   ;DATA STORE RAM TESTS   THERE ARE TWO TESTS FOR THE DATA STORE RAM.
17981                                   ;THE FIRST TEST WILL BE A NO DUAL ADDRESSING TEST AND THE SECOND
17982                                   ;TEST WILL BE A DATA RELIABILITY TEST. THE NO DUAL ADDRESSING TEST
17983                                   ;WILL FIRST WRITE EACH WORD OF THE CACHE RAM WITH ITS WORD ADDRESS
17984                                   ;AND VERIFY THE CONTENTS WITH A READ OF EACH ADDRESS. THEN EACH
17985                                   ;BYTE WILL BE WRITTEN WITH ITS ADDRESS AND CHECKED. THE DATA
17986                                   ;RELIABILITY TEST THAT WILL BE USED IS A TEST CALLED MOVING INVERSIONS.
17987                                   ;
17988                                   ;BGNTST 1
17989                                   ;SETUP MMU REGISTERS TO HAVE BYPASS ON KERNAL SPACE AND NO
17990                                   ;       BYPASS ON USER SPACE
17991                                   ;LET PS EQUAL KERNAL FOR CURRENT MODE AND USER FOR PREVIOUS
17992                                   ;       MODE
17993                                   ;ENABLE MMU
17994                                   ;GET FIRST ADDRESS OF 4K WORD DATA BUFFER
17995                                   ;CLEAR DATA TO BE WRITTEN
17996                                   ;SET DIAGNOSTIC BIT (BIT<1>) IN CCR
17997                                   ;DO UNTIL DATA TO BE WRITTEN EQUALS 20000(8)
17998                                   ;.      WRITE DATA TO ADDRESS
17999                                   ;.      ADD 2 TO ADDRESS
18000                                   ;.      ADD 2 TO DATA TO BE WRITTEN
18001                                   ;ENDDO
18002                                   ;GET FIRST ADDRESS OF 4K WORD DATA BUFFER
18003                                   ;CLEAR EXPECTED DATA
18004                                   ;DO UNTIL EXPECTED DATA EQUALS 20000(8)
18005                                   ;.      READ ADDRESS
18006                                   ;.      IF RECEIVED DATA NE EXPECTED DATA THEN
18007                                   ;.              GOTO DATA STORE PARITY ERROR ROUTINE
18008                                   ;.      ENDIF
18009                                   ;.      ADD 2 TO EXPECTED DATA
18010                                   ;.      ADD 2 TO ADDRESS
18011                                   ;ENDDO
18012                                   ;GET FIRST ADDRESS OF 8K BYTE DATA BUFFER
18013                                   ;CLEAR DATA TO BE WRITTEN
18014                                   ;DO UNTIL ALL BYTES CHECKED
18015                                   ;.      WRITE DATA TO ADDRESS
18016                                   ;.      ADD 1 TO ADDRESS
18017                                   ;.      ADD 1 TO DATA TO BE WRITTEN
18018                                   ;ENDDO
18019                                   ;GET FIRST ADDRESS OF 8K BYTE DATA BUFFER
18020                                   ;CLEAR EXPECTED DATA
18021                                   ;DO UNTIL EXPECTED DATA EQUALS 20000(8)
18022                                   ;.      READ ADDRESS
18023                                   ;.      IF RECEIVED DATA NE EXPECTED DATA THEN
18024                                   ;.              GOTO DATA STORE PARITY ERROR ROUTINE
18025                                   ;.      ENDIF
18026                                   ;.      ADD 1 TO EXPECTED DATA
18027                                   ;.      ADD 1 TO ADDRESS
18028                                   ;ENDDO
18029                                   ;ENDTST
18030                                   ;
18031                                   ;;*****************************************************************
     111600  000004                     TST23:  SCOPE
18032 111602  000240                             NOP
18033 '11604  005737  003032                     TST     CCHPAS                  ;have done enough inclusive passes?
18034 111610  001002                             BNE     99$                     ; not yet
```

G10

TEST - DATA STORE RAM TESTS

```
18035 111612  000240                          NOP                             ; debug a'd
18036 111614  000550                          BR      TST24           ;;GO TO NEXT TEST
18037 111616  000240                  99$:    NOP
18038
18039 111620  004737  136574                  JSR     PC,     INITMM          ;SETUP MMU REGISTERS
18040 111624  012737  002000  172354          MOV     #2000,  KIPAR6          ;LET PAR6 MAP TO ADDRESS 200000
18041 111632  005237  177572                  INC     SR0                     ;TURN ON MEMORY MANAGEMENT
18042 111636  012702  140000                  MOV     #140000,R2              ;GET FIRST ADDRESS OF 4K WORD BUFFER
18043 111642  005001                          CLR     R1                      ;INIT DATA TO BE WRITTEN
18044 111644  052737  000002  177746          BIS     #BIT01, CCR             ;SET DIAG BIT IN CASE NO MEMORY PRESENT
18045 111652  042737  001000  177520          BIC     #1000,BCSR              ;DISABLE HALT ON BREAK
18046 111660  020127  020000          1$:     CMP     R1,     #20000          ;DO UNTIL ALL ADDRESSES WRITTEN
18047 111664  001404                          BEQ     2$                      ;IF DONE GO CHECK DATA
18048 111666  010122                          MOV     R1,     (R2)+           ;WRITE DATA TO ADDRESS
18049 111670  062701  000002                  ADD     #2,     R1              ;UPDATE DATA BY 2 (WORD BOUNDARY)
18050 111674  000771                          BR      1$                      ;ENDDO
18051 111676  042737  000002  177746  2$:     BIC     #BIT01, CCR             ;CLEAR DIAGNOSTIC BIT
18052 111704  012702  140000                  MOV     #140000,R2              ;GET FIRST ADDRESS OF 4K WORD BUFFER
18053 111710  005001                          CLR     R1                      ;INIT DATA TO BE CHECKED
18054 111712  020127  020000          3$:     CMP     R1,     #20000          ;DO UNTIL ALL ADDRESSES CHECKED
18055 111716  001426                          BEQ     5$                      ;IF DONE GO DO BYTES
18056 111720  010137  001122                  MOV     R1,     $BDADR          ;STORE FOR ERRORS
18057 111724  052737  100000  001122          BIS     #BIT15, $BDADR          ;
18058 111732  012237  114150                  MOV     (R2)+,  RECDAT          ;READ TEST LOCATION
18059 111736  013737  177752  001160          MOV     HITMIS, $TMP0           ;STORE REGISTER
18060 111744  032737  000020  001160          BIT     #BIT04,$TMP0            ;HIT?
18061 111752  001001                          BNE     100$                    ;IF YES, BRANCH
18062 111754  104045                          ERROR   +45                     ;DATA RAM ERROR
18063 111756  020137  114150          100$:   CMP     R1,     RECDAT          ;IF RECIEVED DATA NOT EQUAL TO EXPECTED
18064 111762  001401                          BEQ     4$                      ;THEN
18065 111764  104043                          ERROR   +43                     ;DATA RAM ERROR
18066 111766  062701  000002          4$:     ADD     #2,     R1              ;UPDATE EXPECTED DATA BY 2
18067 111772  000747                          BR      3$                      ;ENDDO
18068 111774  012702  140000          5$:     MOV     #140000,R2              ;GET FIRST ADDRESS OF 4K WORD BUFFER
18069 112000  005001                          CLR     R1                      ;INIT DATA TO BE WRITTEN
18070 112002  052737  000002  177746          BIS     #BIT01, CCR             ;SET DIAG BIT IN CASE NO MEMORY PRESENT
18071 112010  020127  020000          6$:     CMP     R1,     #20000          ;DO UNTIL ALL ADDRESSES WRITTEN
18072 112014  001404                          BEQ     7$                      ;IF DONE GO CHECK DATA
18073 112016  110122                          MOVB    R1,     (R2)+           ;WRITE DATA TO ADDRESS
18074 112020  062701  000001                  ADD     #1,     R1              ;UPDATE DATA BY 1 (BYTE BOUNDARY)
18075 112024  000771                          BR      6$                      ;ENDDO
18076 112026  042737  000002  177746  7$:     BIC     #BIT01, CCR             ;CLEAR DIAGNOSTIC BIT
18077 112034  012702  140000                  MOV     #140000,R2              ;GET FIRST ADDRESS OF 4K WORD BUFFER
18078 112040  005001                          CLR     R1                      ;INIT DATA TO BE CHECKED
18079 112042  020127  020000          8$:     CMP     R1,     #20000          ;DO UNTIL ALL ADDRESSES CHECKED
18080 112046  001426                          BEQ     10$                     ;IF DONE EXIT TEST
18081 112050  010137  001122                  MOV     R1,     $BDADR          ;STORE FOR ERRORS
18082 112054  052737  100000  001122          BIS     #BIT15, $BDADR          ;
18083 112062  112237  114150                  MOVB    (R2)+,  RECDAT          ;READ TEST LOCATION
18084 112066  013737  177752  001160          MOV     HITMIS, $TMP0           ;STORE REGISTER
18085 112074  032737  000020  001160          BIT     #BIT04,$TMP0            ;HIT?
18086 112102  001001                          BNE     101$                    ;IF YES, BRANCH
18087 112104  104045                          ERROR   +45                     ;DATA RAM ERROR
18088 112106  120137  114150          101$:   CMPB    R1,     RECDAT          ;IF RECIEVED DATA NOT EQUAL TO EXPECTED
18089 112112  001401                          BEQ     9$                      ;THEN
18090 112114  104043                          ERROR   +43                     ;DATA RAM ERROR
18091 112116  062701  000001          9$:     ADD     #1,     R1              ;UPDATE EXPECTED DATA BY 1
```

TEST - DATA STORE RAM TESTS

```
 18092 112122  000747                          BR      8$                      ;ENDDO
 18093 112124  052737  001000  177520  10$:    BIS     #1000,BCSR              ;ENABLE HALT ON BREAK
 18094 112132  005037  177572                  CLR     SR0                     ;TURN OFF MMU
 18095
```

```
18097                           .SBTTL  TEST    TAG STORE RAM TESTS
18098                           ;TAG STORE RAM TESTS - THERE ARE TWO TESTS FOR THE TAG STORE RAMS. THE FIRST
18099                           ;TEST IS AN ADDRESSING TEST TO ENSURE NO DUAL ADDRESSING OF THE TAG STORE.
18100                           ;THE SECOND TEST IS A "MOVING INVERSIONS" TEST THAT CHECKS THE DATA RELIABILITY
18101                           ;OF THE RAM STORE.
18102                           ;
18103                           ;DUAL ADDRESSING TEST   THIS WILL BE DONE BY FIRST FLUSHING THE CACHE, THEN
18104                           ;THE FIRST 512(10) LOCATIONS (BLOCK 0) WILL BE WRITTEN WITH ADDRESSES THAT
18105                           ;WILL CAUSE A INCREMENTING PATTERN TO BE STORED IN THOSE LOCATIONS OF THE TAG
18106                           ;STORE RAM. NEXT THE ENTIRE 4K ADDRESS RANGE WILL BE READ. THE HIT/MISS
18107                           ;REGISTER SHOULD ONLY REPORT HITS ON THE ADDRESSES THAT WERE ALLOCATED. THIS
18108                           ;PROCESS WILL BE REPEATED FOR EACH BLOCK.
18109                           ;
18110                           ;INITIALIZE LOW ADDRESS
18111                           ;SETUP AND ENABLE MMU
18112                           ;INITIALIZE BLOCK COUNTER
18113                           ;DO UNTIL ALL BLOCKS TESTED (8 TIMES)
18114                           ;.      FLUSH CACHE AND SET DIAGNOSTIC BIT
18115                           ;.      LET CURRENT ADDRESS = LOW ADDRESS
18116                           ;.      INITIALIZE ALLOCATION COUNTER
18117                           ;.      DO UNTIL ENTIRE BLOCK ALLOCATED (1000 TIMES)
18118                           ;.      .       READ CURRENT ADDRESS
18119                           ;.      .       ELSE
18120                           ;.      .       WRITE CURRENT ADDRESS
18121                           ;.      .       ENDIF
18122                           ;.      .       UPDATE CURRENT ADDRESS (ALSO ADD 200 TO PAR)
18123                           ;.      .       FOR I/O PAGE WRITE THE SAME ADDRESS AS BEFORE
18124                           ;.      ENDDO
18125                           ;.      INITIALIZE GOOD ADDRESS
18126                           ;.      INITIALIZE CURRENT ADDRESS
18127                           ;.      INITIALIZE LOCATION COUNTER
18128                           ;.      SAVE CONTENTS OF VECTOR 114
18129                           ;.      LET VECTOR 114 POINT TO TAG STORE PARITY ABORT ROUTINE
18130                           ;.      DO UNTIL ALL LOCATIONS CHECKED (1000 TIMES)
18131                           ;.      .       INITIALIZE CHECK COUNTER
18132                           ;.      .       DO UNTIL ADDRESS IN EACH BLOCK CHECKED
18133                           ;.      .       .       READ CURRENT ADDRESS
18134                           ;.      .       .       IF HIT THEN
18135                           ;.      .       .       .       IF CURRENT ADDRESS NE GOOD ADDRESS THEN
18136                           ;.      .       .       .       .       ERROR
18137                           ;.      .       .       .       ENDIF
18138                           ;.      .       .       ELSE
18139                           ;.      .       .       .       IF CURRENT ADDRESS EQUAL GOOD ADDRESS THEN
18140                           ;.      .       .       .       .       ERROR
18141                           ;.      .       .       .       ENDIF
18142                           ;.      .       .       ENDIF
18143                           ;.      .       .       UPDATE GOOD ADDRESS (ADD #2)
18144                           ;.      .       .       UPDATE CURRENT ADDRESS
18145                           ;.      .       ENDDO
18146                           ;.      .       UPDATE LOW ADDRESS (ADD #2000)
18147                           ;.      ENDDO
18148                           ;ENDDO
18149                           ;DISABLE MMU
18150                           ;RESTORE VECTOR 114
18151                           ;ENDTST
18152
18153                           ;;*****************************************************************
```

```
                112136  000004                  TST24:  SCOPE
          18154 112140  000240                          NOP
          18155 112142  005737  003032                  TST     CCHPAS                  ;have done enough inclus've passes?
          18156 112146  001003                          BNE     99$                     ; not yet
          18157 112150  000240                          NOP                             ; debug aid
          18158 112152  000137  112710                  JMP     18$                     ; yes skip this
          18159 112156  000240                  99$:    NOP
          18160
          18161 112160  013737  000004  001160          MOV     @#4,$TMP0               ;STORE TIMEOUT VECTOR
          18162 112166  012737  112712  000004          MOV     #20$,@#4                ;POINT NEW
          18163 112174  012737  140000  003016          MOV     #140000,LOWADD          ;INITIALIZE LOW ADDRESS (USE PAR6)
          18164 112202  004737  136574                  JSR     PC,     INITMM          ;INITIALIZE MMU
          18165 112206  005237  177572                  INC     SR0                     ;ENABLE MMU
          18166 112212  012737  000020  172516          MOV     #BIT04,MMR3             ;ENABLE 22-BIT MAPPING
          18167 112220  012737  177770  003154          MOV     # 10,   LOOPIN          ;DO UNTIL ALL BLOCKS TESTED
          18168 112226  012701  000000          1$:     MOV     #0,R1                   ;DO IN 2 WORDS TO AVOID PMI MEMORY
          18169 112232  042737  001000  177520  9$:     BIC     #1000,BCSR              ;DISABLE HALT ON BREAK
          18170 112240  005037  172354                  CLR     KIPAR6                  ;SET UP PAR6 FOR THIS TEST
          18171 112244  012737  000402  177746          MOV     #402,   CCR             ;FLUSH CACHE AND SET DIAG BIT
          18172 112252  013737  003016  114170          MOV     LOWADD, CURADD          ;GET FIRST ADDRESS IN CURRENT BLOCK
          18173 112260  012737  177400  003152          MOV     #-400,  ALLCTR          ;DO UNTIL ALL ADDRESSES ALLOCATED
          18174 112266  022737  002000  172354  2$:     CMP     #2000,  KIPAR6          ;IF ADDRESS LESS THAN 32K
          18175 112274  002413                          BLT     3$                      ;THEN
          18176 112276  052737  100000  172314          BIS     #BIT15,KIPDR6           ;SET BYPASS
          18177 112304  017702  001660                  MOV     @CURADD,R2              ;STORE CURRENT DATA
          18178 112310  042737  100000  172314          BIC     #BIT15,KIPDR6           ;ALLOCATE NEXT ACCESS
          18179 112316  010277  001646                  MOV     R2,@CURADD              ;WRITE ALLOCATE
          18180 112322  000402                          BR      4$                      ;ELSE
          18181 112324  005077  001640          3$:     CLR     @CURADD                 ;WRITE CURRENT ADDRESS
          18182 112330  062737  000002  114170  4$:     ADD     #2,     CURADD          ;UPDATE CURRENT ADDRESS
          18183 112336  062737  000200  172354          ADD     #200,   KIPAR6
          18184 112344  022737  177600  172354          CMP     #177600,KIPAR6          ;REACHED I/O PAGE?
          18185 112352  001003                          BNE     10$                     ;BRANCH IF NOT
          18186 112354  162737  000200  172354          SUB     #200,KIPAR6             ;DON'T UPDATE PAR FOR I/O PAGE
          18187 112362  005237  003152          10$:    INC     ALLCTR                  ;IF ALL ADDRESSES ALLOCATED
          18188 112366  002737                          BLT     2$                      ;ENDDO
          18189 112370  052737  000004  177746          BIS     #BIT02, CCR             ;RUN WITH FORCE MISS
          18190 112376  013737  003016  003020          MOV     LOWADD, GOODAD          ;INITIALIZE GOOD ADDRESS
          18191 112404  060137  003020                  ADD     R1,     GOODAD          ;
          18192 112410  012737  140000  114170          MOV     #140000,CURADD          ;GET FIRST ADDRESS
          18193 112416  060137  114170                  ADD     R1,     CURADD          ;START WITH 1 OR 2 LOCATION
          18194 112422  005037  172354                  CLR     KIPAR6
          18195 112426  072127  000006                  ASH     #6,R1                   ;
          18196 112432  060137  172354                  ADD     R1,KIPAR6               ;MAKE SURE ON THE RIGHT BOUNDARY
          18197 112436  012737  177600  003152          MOV     #-200,  ALLCTR          ;DO UNTIL ALL LOCATIONS CHECKED
          18198 112444  012737  177770  114144  5$:     MOV     #-10,   DCOUNT          ;DO UNTIL ADDRESS IN EACH BLOCK CHECKED
          18199 112452  013737  114170  001122  6$:     MOV     CURADD, $BDADR          ;IN CASE OF ERRORS
          18200 112460  042737  160000  001122          BIC     #160000,$BDADR          ;CLEAR PAR BITS
          18201 112466  042737  000004  177746          BIC     #BIT02, CCR             ;CLEAR FORCE MISS
          18202 112474  005777  001470                  TST     @CURADD                 ;READ CURRENT ADDRESS
          18203 112500  013737  177752  114150          MOV     HITMIS, RECDAT          ;STORE REGISTER
          18204 112506  032737  000004  114150          BIT     #BIT2,  RECDAT          ;IF ACCESS WAS A HIT
          18205 112514  001406                          BEQ     7$                      ;THEN
          18206 112516  023737  114170  003020          CMP     CURADD, GOODAD          ;IF CURRENT ADDRESS NOT EQUAL TO GOOD
          18207 112524  001407                          BEQ     8$                      ;ADDRESS THEN
          18208 112526  104046                          ERROR   .46                     ;ERROR IN TAG STORE
          18209 112530  000405                          BR      8$                      ;
```

K10

TEST TAG STORE RAM TESTS

```
18210 112532  023737  114170  003020  7$:     CMP     CURADD, GOODAD          ;IF CURRENT ADDRESS EQUAL GOOD ADDRESS
18211 112540  001001                          BNE     8$                      ;THEN
18212 112542  104047                          ERROR   .47                     ;ERROR IN TAG STORE
18213 112544  062737  001000  114170  8$:     ADD     #1000,  CURADD          ;UPDATE TO NEXT BLOCK
18214 112552  052737  000004  177746          BIS     #BIT02, CCR             ;RUN WITH FORCE MISS
18215 112560  005237  114144                  INC     DCOUNT                  ;IF ADDRESS NOT CHECKED FOR EACH BLOCK
18216 112564  002732                          BLT     6$                      ;ENDDO
18217 112566  062737  000004  003020          ADD     #4,     GOODAD          ;UPDATE GOOD ADDRESS
18218 112574  162737  007774  114170          SUB     #7774,  CURADD          ;INCREMENT IN 2 WORDS
18219 112602  062737  000400  172354          ADD     #400,   KIPAR6
18220 112610  005237  003152                  INC     ALLCTR                  ;IF ALL ADDRESSES CHECKED
18221 112614  002713                          BLT     5$                      ;ENDDO
18222 112616  052737  001000  177520          BIS     #1000,BCSR              ;ENABLE HALT ON BREAK
18223 112624  062701  000002                  ADD     #2,     R1              ;PREPARE FOR THE 2ND PASS THRU
18224 112630  022701  000002                  CMP     #2,     R1              ;DONE TWICE?
18225 112634  001002                          BNE     30$                     ;IF SO, EXIT
18226 112636  000137  112232                  JMP     9$
18227 112642  062737  002000  003016  30$:     ADD     #2000,  LOWADD          ;UPDATE TO NEXT BLOCK TO BE ALLOCATED
18228 112650  005237  003154                  INC     LOOPIN                  ;IF ALL BLOCKS WERE TESTED
18229 112654  002002                          BGE     15$                     ;
18230 112656  000137  112226                  JMP     1$                      ;ENDDO
18231 112662  012737  000400  177746  15$:     MOV     #400,   CCR             ;CLEAR DIAGNOSTIC BIT
18232 112670  005037  177572                  CLR     SR0                     ;DISABLE MMU
18233 112674  013737  001160  000004          MOV     $TMP0,@#4               ;RESTORE TIMEOUT VECTOR
18234 112702  042737  000020  172516          BIC     #BIT04,MMR3             ;ENABLE 22 BIT MAPPING
18235 112710                          18$:
      112710  000401                          BR      TST25           ;;GO TO NEXT TEST
18236
18237 112712  000002                  20$:    RTI
18238
18239
```

```
18241                                   .SBTTL  TEST - STANDALONE MODE TEST
18242                                   ;THIS TEST VERIFIES THAT NMX CAN BE CREATED IN STANDALONE MODE.
18243                                   ;
18244                                   ;ALLOCATE INSTRUCTIONS IN CACHE
18245                                   ;GUARANTEE MISS ON TEST LOCATION
18246                                   ;IN STANALONE MODE ACCESS TEST LOCATION
18247                                   ;IF NO TIMEOUT THEN
18248                                   ;       ERROR
18249                                   ;ENDIF
18250                                   ;
18251                                   ;;*****************************************************************
     112714  000004                   TST25:  SCOPE
18252 112716  000240                           NOP
18253 112720  005737  003032                   TST     CCHPAS                  ;have done enough inclusive passes?
18254 112724  001002                           BNE     99$                     ; not yet
18255 112726  000240                           NOP                             ; debug aid
18256 112730  000452                           BR      TST26           ;;GO TO NEXT TEST
18257 112732  000240                   99$:    NOP
18258
18259 112734  012737  000400  177746           MOV     #400,CCR                ;FLUSH CACHE
18260 112742  012701  112742                   MOV     #.,R1                   ;START WITH CURRENT INSTRUCTION
18261 112746  005721                   1$:     TST     (R1)+                   ;READ A WORD
18262 112750  022701  113046                   CMP     #10$,R1                 ;DONE?
18263 112754  001374                           BNE     1$                      ;IF NOT, CONTINUE
18264 112756  005737  020000                   TST     @#20000                 ;TO GUARANTY MISS ON 0
18265 112762  013702  000004                   MOV     @#4,R2                  ;STORE TIMEOUT VECTOR
18266 112766  012737  113022  000004           MOV     #5$,@#4                 ;POINT NEW TO THE TEST
18267 112774  012737  000340  000006           MOV     #340,@#6                ;AT PRIORITY 7
18268 113002  042737  001000  177520           BIC     #BIT09,BCSR             ;DISABLE HALT ON BREAK
18269 113010  052737  000400  177520           BIS     #BIT08,BCSR             ;GO TO STANDALONE MODE
18270 113016  005737  000000                   TST     @#0                     ;MISS, SHOULD TIMEOUT
18271 113022  042737  000400  177520   5$:     BIC     #BIT08,BCSR             ;CLEAR STANDALONE BIT
18272 113030  052737  001000  177520           BIS     #BIT09,BCSR             ;ENABLE HALT ON BREAK
18273 113036  022716  113022                   CMP     #5$,(SP)                ;TIMEOUT?
18274 113042  001401                           BEQ     10$                     ;IF YES, BRANCH
18275 113044  104044                           ERROR   .44
18276 113046  012706  001100           10$:    MOV     #1100,SP                ;RESTORE STACK
18277 113052  010237  000004                   MOV     R2,@#4                  ;AND TIMEOUT VECTOR
```

```
18279                                   .SBTTL  TEST    MOVING INVERSIONS TEST FOR DATA RAMS
18280                                   ;MOVING INVERSIONS TEST FOR DATA RAMS   THE TEST IS STARTED AFTER LOADING THE
18281                                   ;RAM STORE WITH 0'S. EACH ADDRESS IS READ AND VERIFIED TO BE ALL 0'S. THEN A
18282                                   ;1 IS SUBSTITUTED IN A BIT POSITION AND THE NEW WORD IS WRITTEN. NEXT THE
18283                                   ;ADDRESS IS READ TO VERIFY THE NEW CONTENTS. THIS IS REPEATED FOR EACH BIT OF
18284                                   ;THE WORD LEAVING THE ARRAY FILED WITH 1'S. THE WHOLE PROCESS IS REPEATED
18285                                   ;PLUGGING IN 0 S TO THE 1'S AND REPEATED TWICE MORE ADDRESSING IN THE DOWNWARD
18286                                   ;DIRECTION. FINALLY EVERYTHING IS REPEATED FOR EACH BIT POSITION BEING THE
18287                                   ;LSB. TO SAVE TIME AND KNOWING THE LAYOUT OF THE RAM CHIPS INSTEAD OF DOING
18288                                   ;ONLY A SINGLE BIT AT A TIME EVERY FOURTH BIT WILL BE DONE CONCURRENTLY.
18289                                   ;THIS TEST RUNS IN STANDALONE MODE.
18290                                   ;
18291                                   ;BGNTST
18292                                   ;SETUP AND ENABLE MMU
18293                                   ;SETUP CCR TO ABORT PARITY ERRORS
18294                                   ;CLEAR CACHE
18295                                   ;DO IN STANDALONE MODE FOR EACH HALF SEPARATELY
18296                                   ;LET FWDSEQ = #1
18297                                   ;LET ADDLSB = #1
18298                                   ;DO UNTIL ADDLSB EQ #20000
18299                                   ;.      LET CURDAT = 0
18300                                   ;.      LET RITEDA = #1
18301                                   ;.      LET NEWDAT = #1
18302                                   ;.      IF FWDSEQ = #1 THEN
18303                                   ;.      .       LET FSTADD EQUAL FIRST ADDRESS OF 4K BYTE BUFFER
18304                                   ;.      .       LET LASTAD EQUAL LAST ADDRESS OF 4K BYTE BUFFER
18305                                   ;.      ELSE
18306                                   ;.      .       LET FSTADD EQUAL LAST ADDRESS OF 4K BYTE BUFFER
18307                                   ;.      .       LET LASTAD EQUAL FIRST ADDRESS OF 4K BYTE BUFFER
18308                                   ;.      ENDIF
18309                                   ;.      LET CURADD = FSTADD
18310                                   ;.      LET DCOUNT = #0
18311                                   ;.      SAVE CONTENTS OF VECTOR 114
18312                                   ;.      LET VECTOR 114 POINT TO DATA STORE PARITY ABORT ROUTINE
18313                                   ;.      DO UNTIL DCOUNT EQ #10
18314                                   ;.      .       LET RECDAT = @CURADD
18315                                   ;.      .       IF RECDAT NE CURDAT THEN
18316                                   ;.      .       .       LET R1 EQUAL CURRENT DATA
18317                                   ;.      .       .       GOTO DATA STORE PARITY ERROR ROUTINE
18318                                   ;.      .       ENDIF
18319                                   ;.      .       IF DCOUNT GT #3 THEN
18320                                   ;.      .       .       LET @CURADD = @CURADD CLEARBY RITEDA
18321                                   ;.      .       ELSE
18322                                   ;.      .       .       LET @CURADD = @CURADD SETBY RITEDA
18323                                   ;.      .       ENDIF
18324                                   ;.      .       LET RECDAT = @CURADD
18325                                   ;.      .       IF RECDAT NE NEWDAT THEN
18326                                   ;.      .       .       LET R1 EQUAL NEW DATA
18327                                   ;.      .       .       GOTO DATA STORE PARITY ERROR ROUTINE
18328                                   ;.      .       ENDIF
18329                                   ;.      .       IF CURADD EQ LASTAD THEN
18330                                   ;.      .       .       IF RITEDA = #210 THEN
18331                                   ;.      .       .       .       IF DCOUNT NE #7 THEN
18332                                   ;.      .       .       .       .       LET CURDAT = #377
18333                                   ;.      .       .       .       .       LET RITEDA = #1
18334                                   ;.      .       .       .       .       LET NEWDAT = #376
18335                                   ;.      .       .       .       ENDIF
```

TEST - MOVING INVERSIONS TEST FOR DATA RAMS

```
18336                                   ;.      .       .       ELSE
18337                                   ;.      .       .       .       LET CURDAT = NEWDAT
18338                                   ;.      .       .       .       ROTATE RITEDA
18339                                   ;.      .       .       .       IF DCOUNT GT #3 THEN
18340                                   ;.      .       .       .       .       LET NEWDAT = NEWDAT CLEAREDBY RITEDA
18341                                   ;.      .       .       .       ELSE
18342                                   ;.      .       .       .       .       LET NEWDAT = NEWDAT SETBY RITEDA
18343                                   ;.      .       .       .       ENDIF
18344                                   ;.      .       .       ENDIF
18345                                   ;.      .       .       INCREMENT DCOUNT
18346                                   ;.      .       .       LET CURADD = FSTADD
18347                                   ;.      .       ELSE
18348                                   ;.      .       .       IF FWDSEQ = #1 THEN
18349                                   ;.      .       .       .       LET CURADD = CURADD + ADDLSB
18350                                   ;.      .       .       .       IF CARRY THEN
18351                                   ;.      .       .       .       .       LET CURADD = CURADD + #1
18352                                   ;.      .       .       .       ENDIF
18353                                   ;.      .       .       ELSE
18354                                   ;.      .       .       .       LET CURADD = CURADD - ADDLSB
18355                                   ;.      .       .       .       IF CARRY THEN
18356                                   ;.      .       .       .       .       LET CURADD = LASTAD - ADDLSB
18357                                   ;.      .       .       .       .       LET CURADD = CURADD   #1
18358                                   ;.      .       .       .       ENDIF
18359                                   ;.      .       .       ENDIF
18360                                   ;.      .       ENDIF
18361                                   ;.      ENDDO
18362                                   ;.      IF FWDSEQ EQ #1 THEN
18363                                   ;.      .       LET FWDSEQ = #0
18364                                   ;.      ELSE
18365                                   ;.      .       ROTATE ADDLSB
18366                                   ;.      .       LET FWDSEQ = #1
18367                                   ;.      ENDIF
18368                                   ;ENDDO
18369                                   ;RESTORE VECTOR 114
18370                                   ;ENDTST
18371
18372                                   ;;*****************************************************************
    113056  000004                      TST26:  SCOPE
18373 113060  000240                          NOP
18374 113062  005737  003032                    TST     CCHPAS                  ;have done enough inclusive passes?
18375 113066  001003                          BNE     99$                     ; not yet
18376 113070  000240                          NOP                             ; debug aid
18377 113072  000137  114172                  JMP     ENDMOV                  ; yes skip this
18378 113076  000240                  99$:    NOP
18379
18380 113100  032777  000200  066032          BIT     #BIT07,@SWR             ;RUN THIS TEST?
18381 113106  001002                          BNE     100$                    ;IF SET, GO DO IT
18382 113110  000137  114172                  JMP     ENDMOV                  ;OTHERWISE, GO TO NEXT TEST
18383 113114  042737  001000  177520  100$:   BIC     #1000,BCSR              ;DISABLE HALT ON BREAK
18384 113122  004737  136574                  JSR     PC,     INITMM          ;SETUP MEMORY MANAGEMENT
18385 113126  012737  002000  172354          MOV     #2000,KIPAR6            ;START ON 32K BOUNDARY
18386 113134  005237  177572                  INC     SR0                     ;TURN ON MMU
18387 113140  052737  000002  177746          BIS     #2,     CCR             ;SET DIAG. BIT
18388 113146  013737  000004  001160          MOV     @#4,    $TMP0           ;SAVE 4
18389                                   ;
18390                                   ;STORE TEST IN THE FIRST 2K AND THEN IN THE SECOND 2K
18391                                   ;
```

TEST - MOVING INVERSIONS TEST FOR DATA RAMS

```
18392 113154  012703  140000                    MOV     #140000,R3              ;START FOR THE TEST
18393 113160  012704  150000                    MOV     #150000,R4              ;LOWER BOUNDARY TEST AREA
18394 113164  012705  157777                    MOV     #157777,R5              ;HIGH BOUNDARY TEST AREA
18395 113170  000406                            BR      2$
18396 113172  012703  150000            1$:     MOV     #150000,R3              ;START OF THE TEST
18397 113176  012704  140000                    MOV     #140000,R4              ;LOWER BOUNDARY TEST AREA
18398 113202  012705  147777                    MOV     #147777,R5              ;HIGH BOUNDARY
18399 113206  012702  113326            2$:     MOV     #STMOVI,R2              ;START WITH CURRENT
18400 113212  010300                            MOV     R3,R0                   ;MOVE TO UPPER 4K
18401 113214  012220                    3$:     MOV     (R2)+,(R0)+             ;WORD BY WORD
18402 113216  022702  114172                    CMP     #ENDMOV,R2              ;ALL DONE
18403 113222  001374                            BNE     3$                      ;
18404 113224  010400                            MOV     R4,R0                   ;CLEAR CACHE UNDER TEST
18405 113226  012701  004000                    MOV     #4000,  R1
18406 113232  005020                    4$:     CLR     (R0)+
18407 113234  077102                            SOB     R1,     4$
18408 113236  004713                            JSR     PC,(R3)                 ;GO DO THE ROUTINE
18409 113240  005702                            TST     R2                      ;ANY ERRORS?
18410 113242  001411                            BEQ     5$                      ;IF NOT, CONTINUE
18411 113244  010037  114150                    MOV     R0,RECDAT               ;DATA RECEIVED
18412 113250  010237  001122                    MOV     R2,$BDADR               ;ADDRESS RECIEVED
18413 113254  042737  140000  001122            BIC     #140000,$BDADR          ;STRIP OF PAR BITS
18414 113262  104043                            ERROR   +43                     ;
18415 113264  000403                            BR      6$                      ;EXIT
18416 113266  022703  150000            5$:     CMP     #150000,R3              ;DONE FOR BOTH HALVES?
18417 113272  001337                            BNE     1$                      ;IF NOT, DO AGAIN
18418 113274  052737  001000  177520    6$:     BIS     #BIT09,BCSR             ;ENABLE HALT ON BREAK
18419 113302  013737  001160  000004            MOV     $TMP0,@#4               ;RESTORE TIMEOUT VECTOR
18420 113310  012737  000400  177746            MOV     #400,CCR                ;INIT CCR FOR EXIT
18421 113316  005037  177572                    CLR     SR0                     ;TURN OFF MMU
18422 113322  000137  114172                    JMP     ENDMOV                  ;GO TO NEXT TEST
18423
18424                                   .DSABL AMA
18425 113326  052737  000400  177520  STMOVI:   BIS     #BIT08,@#BCSR           ;STANDALONE MODE
18426 113334  005002                            CLR     R2                      ;ERROR INDICATOR
18427 113336  012767  000001  000606            MOV     #1,     FWDSEQ          ;INIT UPWARD ADDRESSING INDICATOR
18428 113344  012767  000001  000602            MOV     #1,     ADDLSB          ;INIT LSB AND DO LOOP UNTIL SHIFTED OUT
18429 113352  042737  100000  172300            BIC     #100000,@#KIPDR0        ;NO BYPASS
18430 113360  012737  172360  000004            MOV     #KDPAR0,@#4             ;ALLOCATE TIMEOUT VECTOR
18431 113366  012737  000340  000006            MOV     #340,   @#6             ;AT PRIORITY 7
18432 113374  012737  000006  172360            MOV     #6,     @#KDPAR0        ;PUT RETURN
18433 113402  052737  100000  172300            BIS     #100000,@#KIPDR0        ;BYPASS
18434 113410  005067  000546          TSTLUP:   CLR     CURDAT                  ;INIT CURRENT DATA
18435 113414  012767  000021  000534            MOV     #21,    RITEDA          ;INIT DATA TO BE WRITTEN
18436 113422  012767  000021  000530            MOV     #21,    NEWDAT          ;INIT EXPECTED DATA
18437 113430  005767  000516                    TST     FWDSEQ                  ;IF ADDRESSING UPWARD
18438 113434  001405                            BEQ     1$                      ;THEN
18439 113436  010467  000522                    MOV     R4,FSTADD               ;LET FIRST ADDRESS EQUAL LOWEST VALUE
18440 113442  010567  000520                    MOV     R5,LSTADD               ;LET LAST ADDRESS EQUAL HIGHEST VALUE
18441 113446  000404                            BR      2$                      ;ELSE
18442 113450  010567  000510            1$:     MOV     R5,FSTADD               ;LET FIRST ADDRESS EQUAL HIGHEST VALUE
18443 113454  010467  000506                    MOV     R4,LSTADD               ;LET LAST ADDRESS EQUAL LOWEST VALUE
18444 113460  016767  000500  000502    2$:     MOV     FSTADD, CURADD          ;LET CURRENT ADDRESS EQUAL FIRST ADDRESS
18445 113466  005067  000452                    CLR     DCOUNT                  ;INIT LOOP COUNTER
18446 113472  022767  000010  000444  BGNTLP:   CMP     #10,    DCOUNT          ;DO LOOP 8 TIMES (4 TIMES TO WRITE 1 S
18447 113500  001567                            BEQ     ENDTLP                  ;AND 4 TIMES TO WRITE BACK 0'S)
18448                                   ;
```

TEST - MOVING INVERSIONS TEST FOR DATA RAMS

```
18449                                   ; DON'T REWRITE TIMEOUT VECTOR
18450                                   ;
18451 113502  016700  000462                    MOV     CURADD, R0              ;STORE CURRENT ADDRESS
18452 113506  042700  170000                    BIC     #170000,R0              ;LEAVE ONLY LOW 4K BITS
18453 113512  022700  000004                    CMP     #4,     R0              ;TIMOUT VECTOR?
18454 113516  001531                          BEQ     8$                      ;IF SO, DON'T REWRITE IT
18455 113520  022700  000005                    CMP     #5,     R0              ;
18456 113524  001526                          BEQ     8$                      ;
18457 113526  022700  000006                    CMP     #6,     R0              ;
18458 113532  001523                          BEQ     8$                      ;
18459 113534  022700  000007                    CMP     #7,     R0              ;
18460 113540  001520                          BEQ     8$                      ;
18461                                   ;
18462                                   ;CHECK PATTERNS NOW
18463                                   ;
18464 113542  016701  000414                    MOV     CURDAT, R1              ;MOVE GOOD DATA TO R1
18465 113546  117767  000416  000374            MOVB    @CURADD,RECDAT          ;FIRST READ LOCATION
18466 113554  126767  000370  000400            CMPB    RECDAT, CURDAT          ;IF RECIEVED DATA NOT EQUAL TO EXPECTED
18467 113562  001402                          BEQ     1$                      ;DATA THEN
18468 113564  000167  000334                    JMP     EXBAD                   ;EXIT TEST
18469 113570  016701  000364          1$:       MOV     NEWDAT, R1              ;MOVE GOOD DATA TO R1
18470 113574  022767  000003  000342            CMP     #3,     DCOUNT          ;IF LOOP COUNT GREATER THAN 3
18471 113602  002004                          BGE     2$                      ;THEN
18472 113604  146777  000346  000356            BICB    RITEDA, @CURADD         ;CLEAR TEST DATA BY WRITE DATA
18473 113612  000403                          BR      3$                      ;ELSE
18474 113614  156777  000336  000346  2$:       BISB    RITEDA, @CURADD         ;SET TEST DATA BY WRITE DATA
18475 113622  117767  000342  000320  3$:       MOVB    @CURADD,RECDAT          ;DO READ AFTER WRITE
18476 113630  126767  000314  000322            CMPB    RECDAT, NEWDAT          ;IF RECIEVED DATA NOT EQUAL TO EXPECTED
18477 113636  001402                          BEQ     4$                      ;DATA THEN
18478 113640  000167  000260                    JMP     EXBAD                   ;EXIT TEST
18479 113644  026767  000320  000314  4$:       CMP     CURADD, LSTADD          ;IF CURRENT ADDRESS EQUALS LAST ADDRESS
18480 113652  001053                          BNE     8$                      ;THEN
18481 113654  022767  000210  000274            CMP     #210,   RITEDA          ;AND IF WRITE PATTERN EQUALS LAST
18482 113662  001016                          BNE     5$                      ;PATTERN THEN
18483 113664  022767  000007  000252            CMP     #7,     DCOUNT          ;AND TEST IS NOT ON LAST LOOP
18484 113672  001435                          BEQ     7$                      ;THEN
18485 113674  012767  000377  000260            MOV     #377,   CURDAT          ;SET UP TO WRITE 0'S
18486 113702  012767  000021  000246            MOV     #21,    RITEDA
18487 113710  012767  000356  000242            MOV     #356,   NEWDAT
18488 113716  000423                          BR      7$                      ;ELSE
18489 113720  016767  000234  000234  5$:       MOV     NEWDAT, CURDAT          ;SET UP FOR NEXT DATA PATTERN
18490 113726  000241                          CLC
18491 113730  106167  000222                    ROLB    RITEDA
18492 113734  005567  000216                    ADC     RITEDA                  ;GET A PATTERN
18493 113740  022767  000003  000176            CMP     #3,     DCOUNT          ;AND IF DOWNWARD ADDRESSING
18494 113746  002004                          BGE     6$                      ;THEN
18495 113750  046767  000202  000202            BIC     RITEDA, NEWDAT          ;LET NEWDAT BE CLEARED BY WRITE DATA
18496 113756  000403                          BR      7$                      ;ELSE
18497 113760  056767  000172  000172  6$:       BIS     RITEDA, NEWDAT          ;LET NEWDAT BE SET BY WRITE DATA
18498 113766  005267  000152          7$:       INC     DCOUNT                  ;UPDATE LOOP COUNTER
18499 113772  016767  000166  000170            MOV     FSTADD  CURADD          ;REINIT FIRST ADDRESS FOR THIS PASS
18500 114000  000426                          BR      10$                     ;ELSE
18501 114002  005767  000144          8$:       TST     FWDSEQ                  ;IF ADDRESSING UPWARD
18502 114006  001412                          BEQ     9$                      ;THEN
18503 114010  066767  000140  000152            ADD     ADDLSB, CURADD          ;CALCULATE NEXT HIGHER ADDRESS
18504 114016  020567  000146                    CMP     R5,CURADD               ;IF CURRENT ADDRESS HAS BEEN INCREASED
18505 114022  002015                          BGE     10$                     ;ABOVE HIGHEST ADDRESS THEN
```

TEST - MOVING INVERSIONS TEST FOR DATA RAMS

```
18506 114024  162767  007777  000136          SUB     #7777,  CURADD          ;ROLL ADDRESS BACK
18507 114032  000411                          BR      10$                     ;ELSE
18508 114034  166767  000114  000126  9$:     SUB     ADDLSB, CURADD          ;CALCULATE NEXT LOWER ADDRESS
18509 114042  020467  000122                  CMP     R4,CURADD               ;IF CURRENT ADDRESS HAS BEEN DECREASED
18510 114046  003403                          BLE     10$                     ;BELOW LOWEST ADDRESS THEN
18511 114050  062767  007777  000112          ADD     #7777,  CURADD          ;ROLL ADDRESS BACK
18512 114056  000605                  10$:    BR      BGNTLP                  ;ENDDO
18513 114060  005767  000066          ENDTLP: TST     FWDSEQ                  ;IF ADDRESSING UPWARD FINISHED
18514 114064  001404                          BEQ     1$                      ;THEN
18515 114066  005067  000060                  CLR     FWDSEQ                  ;DO ADDRESSING DOWNWARD
18516 114072  000167  177312                  JMP     TSTLUP
18517 114076  012767  000001  000046  1$:     MOV     #1,     FWDSEQ          ;SET ADDRESSING UPWARD INDICATOR
18518 114104  006167  000044                  ROL     ADDLSB                  ;UPDATE LSB TO NEXT POSITION
18519 114110  022767  020000  000036  ENDLUP: CMP     #20000,ADDLSB           ;ALL DONE?
18520 114116  001406                          BEQ     EXITST                  ;ENDDO
18521 114120  000167  177264                  JMP     TSTLUP
18522 114124  016700  000020          EXBAD:  MOV     RECDAT,R0               ;STORE RECEIVED DATA
18523 114130  016702  000034                  MOV     CURADD,R2               ;STORE ADDRESS
18524 114134  042737  000400  177520  EXITST: BIC     #BIT08,@#BCSR           ;OUT OF STANDALONE
18525 114142  000207                          RTS     PC
18526                                 .ENABL AMA
18527
18528 114144  000000                  DCOUNT: .WORD   0
18529 114146  000000                  EXPDAT: .WORD   0               ;STORES EXPECTED (GOOD) DATA FOR COMPARISONS
18530 114150  000000                  RECDAT: .WORD   0               ;STORES RECIEVED DATA TO BE VERIFIED
18531 114152  000000                  FWDSEQ: .WORD   0               ;USED TO INDICATE DIRECTION OF ADDRESSING
18532 114154  000000                  ADDLSB: .WORD   0               ;STORES LEAST SIGNIFICANT BIT FOR RAM TESTS
18533 114156  000000                  RITEDA: .WORD   0               ;STORES WRITE DATA FOR RAM TESTS
18534 114160  000000                  NEWDAT: .WORD   0               ;DATA STORE FOR RAM TESTS
18535 114162  000000                  CURDAT: .WORD   0               ;DATA STORE FOR RAM TESTS
18536 114164  000000                  FSTADD: .WORD   0               ;STORES FIRST ADDRESS IN ADDRESSING SEQUENCE
18537 114166  000000                  LSTADD: .WORD   0               ;STORES LAST ADDRESS IN ADDRESSING SEQUENCE
18538 114170  000000                  CURADD: .WORD   0               ;STORES CURRENT ADDRESS FOR RAM TESTS
18539
18540 114172                          ENDMOV:
```

```
18542                                   .SBTTL  TEST  - MOVING INVERSIONS TEST FOR TAG STORE
18543                                   ;MOVING INVERSIONS TEST   THE TEST IS STARTED AFTER LOADING THE RAM STORE
18544                                   ;WITH 0'S. EACH ADDRESS IS READ AND VERIFIED TO BE ALL 0'S. THEN A 1 IS
18545                                   ;SUBSTITUTED IN A BIT POSITION AND THE NEW WORD IS WRITTEN. NEXT THE ADDRESS
18546                                   ;IS READ TO VERIFY THE NEW CONTENTS. THIS IS REPEATED FOR EACH BIT OF THE
18547                                   ;WORD LEAVING THE ARRAY FILED WITH 1'S. THE WHOLE PROCESS IS REPEATED PLUGGING
18548                                   ;IN 0'S TO THE 1'S AND REPEATED TWICE MORE ADDRESS IN THE DOWARD DIRECTION.
18549                                   ;FINALLY EVERYTHING IS REPEATED FOR EACH BIT POSITION BEING THE LSB. TO SAVE
18550                                   ;TIME AND KNOWING THE LAYOUT OF THE RAM CHIPS INSTEAD OF DOING ONLY A SINGLE
18551                                   ;BIT AT A TIME EVERY THIRD BIT WILL BE DONE CONCURRENTLY. ALSO NOTE THAT
18552                                   ;SINCE THE TAG STORE CANNOT BE DIRECTLY ACCESSED THE ENTIRE PATTERN MUST BE
18553                                   ;DONE BY DOING MEMORY CYCLES TO THE CORRECT BUS ADDRESSES. TO DO THIS THE
18554                                   ;TEST WILL BE DONE WITH THE DIAGNOSTIC BIT IN THE CCR (BIT 1) SET TO A 1.
18555                                   ;THIS TEST RUNS IN STANDALONE MODE.
18556                                   ;
18557                                   ;
18558                                   ;BGNTST
18559                                   ;SETUP AND ENABLE MMU
18560                                   ;SETUP CCR TO ABORT PARITY ERRORS
18561                                   ;DO IN STANDALONE MODE FOR EACH HALF SEPARATELY
18562                                   ;LET FWDSEQ = #1
18563                                   ;LET ADDLSB = #1
18564                                   ;DO UNTIL ADDLSB EQ #20000
18565                                   ;.      LET NEWDAT = 22200
18566                                   ;.      LET CURDAT = 0
18567                                   ;.      IF FWDSEQ = #1 THEN
18568                                   ;.      .       LET FSTADD EQUAL FIRST ADDRESS OF 4K BYTE BUFFER
18569                                   ;.      .       LET LASTAD EQUAL LAST ADDRESS OF 4K BYTE BUFFER
18570                                   ;.      ELSE
18571                                   ;.      .       LET FSTADD EQUAL LAST ADDRESS OF 4K BYTE BUFFER
18572                                   ;.      .       LET LASTAD EQUAL FIRST ADDRESS OF 4K BYTE BUFFER
18573                                   ;.      ENDIF
18574                                   ;.      LET DCOUNT = #0
18575                                   ;.      DO UNTIL DCOUNT EQ #10
18576                                   ;.      .       READ CURADD USING CURDAT AS PAR
18577                                   ;.      .       IF MISS THEN
18578                                   ;.      .       .       ERROR
18579                                   ;.      .       ENDIF
18580                                   ;.      .       WRITE CURADD USING NEWDAT AS PAR
18581                                   ;.      .       ENDIF
18582                                   ;.      .       IF HIT THEN
18583                                   ;.      .       .       ERROR
18584                                   ;.      .       ENDIF
18585                                   ;.      .       READ CURADD USING NEWDAT AS PAR
18586                                   ;.      .       IF MISS THEN
18587                                   ;.      .       .       ERROR
18588                                   ;.      .       ENDIF
18589                                   ;.      .       IF CURADD EQ LASTAD THEN
18590                                   ;.      .       .       LET CURDAT = NEWDAT
18591                                   ;.      .       .       IF DCOUNT < #1 THEN
18592                                   ;.      .       .       .       LET NEWDAT = NEWDAT SETBY RITEDA ROTATED LEFT
18593                                   ;.      .       .       ENDIF
18594                                   ;.      .       .       IF DCOUNT > #1 THEN
18595                                   ;.      .       .       .       LET NEWDAT = NEWDAT CLR BY RITEDA ROTATED LEFT
18596                                   ;.      .       .       ELSE
18597                                   ;.      .       .       .       LET NEWDAT = #155400 (NO ALL 1'S)
18598                                   ;.      .       .       .       LET RITEDA = #22200
```

TEST - MOVING INVERSIONS TEST FOR TAG STORE

```
18599                                   ;.      .       .       ENDIF
18600                                   ;.      .       .       INCREMENT DCOUNT
18601                                   ;.      .       ELSE
18602                                   ;.      .       .       IF FWDSEQ = #1 THEN
18603                                   ;.      .       .       .       LET CURADD = CURADD + ADDLSB
18604                                   ;.      .       .       .       IF CURADD GE HIGH ADDRESS THEN
18605                                   ;.      .       .       .       .       LET CURADD = CURADD + #7776
18606                                   ;.      .       .       .       ENDIF
18607                                   ;.      .       .       ELSE
18608                                   ;.      .       .       .       LET CURADD = CURADD   ADDLSB
18609                                   ;.      .       .       .       IF CURADD LE LOW ADDRESS THEN
18610                                   ;.      .       .       .       .       LET CURADD - CURADD   #7776
18611                                   ;.      .       .       .       ENDIF
18612                                   ;.      .       .       ENDIF
18613                                   ;.      .       ENDIF
18614                                   ;.      ENDDO
18615                                   ;.      IF FWDSEQ EQ #1 THEN
18616                                   ;.      .       LET FWDSEQ = #0
18617                                   ;.      ELSE
18618                                   ;.      .       ROTATE ADDLSB
18619                                   ;.      .       LET FWDSEQ = #1
18620                                   ;.      ENDIF
18621                                   ;ENDDO
18622                                   ;RESTORE PARITY ABORT VECTOR
18623                                   ;ENDTST
18624
18625                                   ;;*****************************************************************
     114172  000004                    TST27:  SCOPE
18626 114174  000240                            NOP
18627 114176  005737  003032                    TST     CCHPAS                  ;have done enough inclusive passes?
18628 114202  001003                            BNE     99$                     ; not yet
18629 114204  000240                            NOP                             ; debug aid
18630 114206  000137  115314                    JMP     ENDTAG                  ; yes skip th's
18631 114212  000240                    99$:    NOP
18632
18633 114214  032777  000100  064716            BIT     #BIT06,@SWR             ;RUN THIS TEST?
18634 114222  001002                            BNE     100$                    ;IF SET, GO DO IT
18635 114224  000137  115314                    JMP     ENDTAG                  ;OTHERWISE, GO TO NEXT TEST
18636 114230  042737  001000  177520  100$:    BIC     #1000,BCSR              ;DISABLE HALT ON BREAK
18637 114236  004737  136574                    JSR     PC,     INITMM          ;SETUP MEMORY MANAGEMENT
18638 114242  005237  177572                    INC     SR0                     ;TURN ON MMU
18639 114246  012737  000020  172516            MOV     #BIT04,MMR3             ;ENABLE 22 BIT MAPPING
18640 114254  052737  000002  177746            BIS     #2,     CCR             ;SET DIAG. BIT
18641 114262  013737  000004  001160            MOV     @#4,    $TMP0           ;STORE TIMEOUT VECTOR
18642                                   ;
18643                                   ;STORE TEST IN THE FIRST 2K AND THEN IN THE SECOND 2K
18644                                   ;
18645 114270  012703  140000                    MOV     #140000,R3              ;START FOR THE TEST
18646 114274  012704  130000                    MOV     #130000,R4              ;LOWER BOUNDARY TEST AREA
18647 114300  012705  137776                    MOV     #137776,R5              ;HIGH BOUNDARY TEST AREA
18648 114304  000406                            BR      2$
18649 114306  012703  150000            1$:     MOV     #150000,R3              ;START OF THE TEST
18650 114312  012704  120000                    MOV     #120000,R4              ;LOWER BOUNDARY TEST AREA
18651 114316  012705  127776                    MOV     #127776,R5              ;HIGH BOUNDARY
18652 114322  012737  002000  172354    2$:     MOV     #2000,KIPAR6            ;ABOVE 32K
18653 114330  012702  114134                    MOV     #EXITST,R2              ;START WITH CURRENT
18654 114334  010300                            MOV     R3,R0                   ;MOVE TO UPPER 4K
```

TEST - MOVING INVERSIONS TEST FOR TAG STORE

```
18655 114336  012220                    3$:     MOV     (R2)+,(R0)+             ;WORD BY WORD
18656 114340  022702  115314                    CMP     #ENDTAG,R2              ;ALL DONE?
18657 114344  001374                            BNE     3$                      ;
18658 114346  012700  114450                    MOV     #STMOVT,R0              ;ADDRESS OF ROUTINE
18659 114352  162700  114134                    SUB     #EXITST,R0              ;PROPER OFSET
18660 114356  050003                            BIS     R0,R3                   ;
18661 114360  004713                            JSR     PC,(R3)                 ;GO DO THE ROUTINE
18662 114362  005700                            TST     R0                      ;ANY ERRORS?
18663 114364  001407                            BEQ     4$                      ;CONTINUE
18664 114366  010037  001122                    MOV     R0,$BDADR               ;STORE FAILED ADDRESS
18665 114372  042737  160000  001122            BIC     #160000,$BDADR          ;CLEAR PAR
18666 114400  104050                            ERROR   +50
18667 114402  000403                            BR      5$                      ;EXIT TEST
18668 114404  022703  150000          4$:       CMP     #150000,R3              ;DONE FOR BOTH HALVES?
18669 114410  103736                            BLO     1$                      ;IF NOT, DO AGAIN
18670 114412  013737  001160  000004  5$:       MOV     $TMP0,  @#4             ;RESTORE TIMEOUT VECTOR
18671 114420  012737  000400  177746            MOV     #400,   CCR             ;INIT CCR FOR EXIT
18672 114426  005037  177572                    CLR     SR0                     ;TURN OFF MMU
18673 114432  005037  172516                    CLR     MMR3                    ;
18674 114436  052737  001000  177520            BIS     #1000,BCSR              ;ENABLE HALT ON BREAK
18675 114444  000137  115314                    JMP     ENDTAG                  ;EXIT TEST
18676
18677                                   .DSABL AMA
18678 114450  052737  000400  177520  STMOVT:   BIS     #BIT08,@#BCSR           ;STANDALONE MODE
18679 114456  042737  100000  172312            BIC     #BIT15, @#KIPDR5        ;NO BYPASS
18680 114464  042737  100000  172300            BIC     #100000,@#KIPDR0        ;NO BYPASS
18681 114472  012737  172360  000004            MOV     #KDPAR0,@#4             ;ALLOCATE TIMEOUT VECTOR
18682 114500  012737  000340  000006            MOV     #340,   @#6             ;AT 7
18683 114506  012737  000006  172360            MOV     #6,     @#KDPAR0        ;PUT RETURN
18684 114514  052737  100000  172300            BIS     #100000,@#KIPDR0        ;BYPASS
18685 114522  005037  172352                    CLR     @#KIPAR5                ;
18686 114526  010400                            MOV     R4,R0                   ;CLEAR CACHE UNDER TEST
18687 114530  012701  004000                    MOV     #4000,  R1              ;
18688 114534  005020                    4$:     CLR     (R0)+                   ;
18689 114536  077102                            SOB     R1,     4$              ;
18690 114540  005000                            CLR     R0                      ;CLEAR ERROR FLAG
18691 114542  012767  000001  177402            MOV     #1,     FWDSEQ          ;SET UPWARD ADDRESSING INDICATOR
18692 114550  012767  000002  177376            MOV     #2,     ADDLSB          ;INITIALIZE LEAST SIGNIFIGANT BIT
18693 114556  005067  177400          TSLOOP:   CLR     CURDAT                  ;OLD DATA
18694 114562  012767  022200  177370            MOV     #22200,NEWDAT           ;SET ADDRESS BITS
18695 114570  012767  022200  177360            MOV     #22200,RITEDA           ;SET ADDRESS BITS
18696 114576  005767  177350                    TST     FWDSEQ                  ;IF ADDRESSING UPWARD
18697 114602  001405                            BEQ     1$                      ;THEN
18698 114604  010467  177354                    MOV     R4,     FSTADD          ;FIRST ADDRESS WILL BE LOWEST ADDRESS
18699 114610  010567  177352                    MOV     R5,     LSTADD          ;AND LAST ADDRESS WILL BE HIGHEST
18700 114614  000404                            BR      2$                      ;ELSE
18701 114616  010567  177342          1$:       MOV     R5,     FSTADD          ;FIRST ADDRESS WILL BE HIGHEST ADDRESS
18702 114622  010467  177340                    MOV     R4,     LSTADD          ;AND LAST ADDRESS WILL BE LOWEST
18703 114626  005067  177312          2$:       CLR     DCOUNT                  ;INITIALIZE LOOP COUNTER
18704 114632  016767  177326  177330            MOV     FSTADD, CURADD
18705                                   ;
18706                                   ; DON'T REWRITE TIMEOUT VECTOR
18707                                   ;
18708 114640  016700  177324          3$:       MOV     CURADD, R0              ;STORE CURRENT ADDRESS
18709 114644  042700  170000                    BIC     #170000,R0              ;LEAVE ONLY LOW 4K BITS
18710 114650  022700  000004                    CMP     #4,     R0              ;TIMOUT VECTOR?
18711 114654  001526                            BEQ     16$                     ;IF SO, DON'T REWRITE IT
```

TEST - MOVING INVERSIONS TEST FOR TAG STORE

```
18712 114656  022700  000006                 CMP     #6,     R0              ;OR PRIORITY
18713 114662  001523                         BEQ     16$                     ;
18714 114664  016737  177272  172352         MOV     CURDAT, @#KIPAR5        ;GET OLD PATTERN
18715 114672  005777  177272                 TST     @CURADD                 ;OLD DATA OK?
18716 114676  013767  177752  177244         MOV     @#HITMIS,RECDAT
18717 114704  032767  000004  177236  5$:    BIT     #BIT02, RECDAT          ;IF ACCESS WAS A MISS
18718 114712  001002                         BNE     6$                      ;THEN
18719 114714  000167  000346                 JMP     EXBAD2                  ;ERROR, EXIT
18720 114720  016737  177234  172352  6$:    MOV     NEWDAT, @#KIPAR5        ;GET NEW PATTERN
18721 114726  005077  177236                 CLR     @CURADD                 ;REGISTER AND WRITE LOCATION
18722 114732  013767  177752  177210         MOV     @#HITMIS,RECDAT
18723 114740  032767  000004  177202  8$:    BIT     #BIT02, RECDAT          ;IF ACCESS WAS A HIT
18724 114746  001406                         BEQ     9$                      ;THEN
18725 114750  032767  000010  177172  10$:   BIT     #BIT03, RECDAT          ;MISS?
18726 114756  001402                         BEQ     9$                      ;IF SO, CONTINUE
18727 114760  000167  000302                 JMP     EXBAD2                  ;ERROR
18728 114764  005777  177200          9$:    TST     @CURADD                 ;REGISTER AND READ LOCATION
18729 114770  013767  177752  177152         MOV     @#HITMIS,RECDAT
18730 114776  032767  000004  177144  11$:   BIT     #BIT02, RECDAT          ;IF ACCESS WAS A MISS
18731 115004  001002                         BNE     12$                     ;THEN
18732 115006  000167  000254                 JMP     EXBAD2                  ;ERROR, EXIT
18733 115012  026767  177150  177150  12$:   CMP     LSTADD, CURADD          ;IF CURRENT ADDRESS IS LAST ADDRESS
18734 115020  001044                         BNE     16$                     ;THEN
18735 115022  016767  177132  177132         MOV     NEWDAT,CURDAT           ;NEW KIPAR5
18736 115030  022767  000001  177106         CMP     #1,     DCOUNT          ;IF LOOP COUNTER LESS THAN 1
18737 115036  003406                         BLE     13$                     ;THEN
18738 115040  006367  177112                 ASL     RITEDA                  ;UPDATE OF NEW DATA....
18739 115044  056767  177106  177106         BIS     RITEDA, NEWDAT          ;BY SETTING BITS
18740 115052  000421                         BR      15$                     ;ENDIF
18741 115054  022767  000001  177062  13$:   CMP     #1,     DCOUNT          ;CHANGE PATTERN?
18742 115062  001007                         BNE     14$                     ;IF SO, BRANCH
18743 115064  012767  022200  177064         MOV     #22200, RITEDA          ;START WITH THE SAME
18744 115072  012767  155400  177060         MOV     #155400,NEWDAT          ;DON'T DO ALL 1'S
18745 115100  000406                         BR      15$                     ;
18746 115102  006367  177050          14$:   ASL     RITEDA                  ;UPDATE OF NEW DATA....
18747 115106  046767  177044  177044         BIC     RITEDA, NEWDAT          ;BY CLEARING BITS
18748 115114  000400                         BR      15$                     ;ELSE
18749 115116  005267  177022          15$:   INC     DCOUNT                  ;INCREMENT THE LOOP COUNTER
18750 115122  016767  177036  177040         MOV     FSTADD, CURADD          ;FINISH FIRST CURRENT ADDRESS
18751 115130  000426                         BR      18$                     ;ELSE
18752 115132  005767  177014          16$:   TST     FWDSEQ                  ;IF ADDRESSING UPWARD
18753 115136  001412                         BEQ     17$                     ;THEN
18754 115140  066767  177010  177022         ADD     ADDLSB, CURADD          ;ADD THE VALUE OF THE CURRENT LSB
18755 115146  020567  177016                 CMP     R5,     CURADD          ;IF CURRENT ADDRESS GREATER THAN HIGHEST
18756 115152  002015                         BGE     18$                     ;VIRTUAL ADDRESS THEN
18757 115154  162767  007776  177006         SUB     #7776,  CURADD          ;ROLL IT BACK
18758 115162  000411                         BR      18$                     ;ELSE
18759 115164  166767  176764  176776  17$:   SUB     ADDLSB, CURADD          ;IF ADDRESSING DOWNWARD THEN SUBTRACT LSB
18760 115172  020467  176772                 CMP     R4,     CURADD          ;IF CURRENT ADDRESS LESS THEN LOWEST
18761 115176  003403                         BLE     18$                     ;VIRTUAL ADDRESS THEN
18762 115200  062767  007776  176762         ADD     #7776,  CURADD          ;ROLL IT BACK
18763 115206  022767  000005  176730  18$:   CMP     #5,     DCOUNT          ;IF LOOP COUNTER LESS THAN 7
18764 115214  003402                         BLE     181$                    ;THEN LOOP BACK FOR NEXT BIT POSITION
18765 115216  000167  177416                 JMP     3$
18766 115222  005767  176724          181$:  TST     FWDSEQ                  ;IF ADDRESSING UPWARD
18767 115226  001404                         BEQ     19$                     ;THEN
18768 115230  005067  176716                 CLR     FWDSEQ                  ;START ADDRESSING DOWNWARD
```

TEST MOVING INVERSIONS TEST FOR TAG STORE

```
18769 115234  000167  177316                    JMP     TSLOOP                  ;ELSE
18770 115240  012767  000001  176704  19$:      MOV     #1,     FWDSEQ          ;START ADDRESSING UPWARD
18771 115246  006167  176702                    ROL     ADDLSB                  ;ROTATE LSB TO NEXT POSITION
18772 115252  022767  020000  176674            CMP     #20000, ADDLSB          ;ALL DONE?
18773 115260  001406                           BEQ     TSEND                   ;EXIT
18774 115262  000167  177270                    JMP     TSLOOP                  ;ENDDO
18775 115266  013701  172352          EXBAD2:   MOV     @#KIPAR5,R1             ;STORE PAR
18776 115272  016700  176672                    MOV     CURADD, R0              ;AND ADDRESS
18777 115276  012737  001200  172352  TSEND:    MOV     #1200,  @#KIPAR5        ;RESTORE PAR
18778 115304  042737  000400  177520            BIC     #BIT08, @#BCSR          ;OUT OF STANDALONE MODE
18779 115312  000207                           RTS     PC                      ;RETURN
18780 115314                          ENDTAG:
18781                                 .ENABL AMA
18782
```

```
18784                                   .SBTTL  TEST    PCR READ/WRITE BITS
18785                                   ;PCR AND BCSR READ/WRITE BITS
18786                                   ;THE FIRST TEST WILL CHECK THAT PCR REGISTER IS BOTH WORD AND
18787                                   ;BYTE ADDRESSABLE. BITS 14 09 AND 06-01 WILL BE WRITTEN AND READ
18788                                   ;AS ZEROES AND ONES. THE REST OF THE BITS HAVE TO BE ALL 0'S.
18789                                   ;ROUTINE TEST
18790                                   ;.      SAVE PCR
18791                                   ;.      LET PCR=0
18792                                   ;.      DO FOR PATTERN=001111,110011,101010,010101
18793                                   ;.      .       WRITE PCR<14 09>=PATTERN
18794                                   ;.      .       WRITE PCR<06 01>=PATTERN
18795                                   ;.      .       IF PCR<14 09> NE PATTERN OR PCR<06-01> NE PREVIOUS
18796                                   ;.      .                                               PATTERN
18797                                   ;.      .       .       THEN ERROR
18798                                   ;.      .       ENDIF
18799                                   ;.      .       WRITE PCR<06 01>=PATTERN
18800                                   ;.      .       IF PCR<14 09> NE PATTERN OR PCR<06 01> NE PATTERN
18801                                   ;.      .       .       THEN ERROR
18802                                   ;.      .       ENDIF
18803                                   ;.      ENDDO
18804                                   ;.      WRITE PCR=0101010101010101
18805                                   ;.      IF PCR NE 0101010001010100 THEN
18806                                   ;.      .       ERROR
18807                                   ;.      ENDIF
18808                                   ;ENDROUTINE
18809
18810                                   ;;*****************************************************************
        115314  000004                  TST30:  SCOPE
18811   115316  005037  177522                  CLR     PCR                     ;INITIALIZE PCR TO 0
18812   115322  012702  115461                  MOV     #SIXBIT+1,R2            ;R2->TABLE OF PATTERNS FOR 6 R/W BITS IN EACH
18813   115326  012704  115460                  MOV     #SIXBIT,R4              ;R3 POINTER TO PREVIOUS PATTERN
18814   115332  012703  000004                  MOV     #4,R3                   ;DO 4 TIMES
18815                                   ;
18816                                   ; WRITE TO HIGH BYTE FIRST
18817                                   ;
18818   115336  111237  177523          1$:     MOVB    (R2),PCR+1              ;WRITE TO HIGH BYTE
18819   115342  121237  177523                  CMPB    (R2),PCR+1              ;BYTE WRITTEN OK?
18820   115346  001003                          BNE     2$                      ;IF NOT, BRANCH
18821   115350  121437  177522                  CMPB    (R4),PCR                ;LOW BYTE CHANGED?
18822   115354  001405                          BEQ     3$                      ;IF NOT, BRANCH
18823   115356  111237  001125          2$:     MOVB    (R2),$GDDAT+1           ;EXPECTED PATTERN HIGH BYTE
18824   115362  111437  001124                  MOVB    (R4),$GDDAT             ;LOW BYTE
18825   115366  104051                          ERROR   +51                     ;ERROR PCR READ/WRITE BITS
18826   115370  105724                  3$:     TSTB    (R4)+                   ;INCREMENT POINTER FOR OLD
18827                                   ;
18828                                   ; WRITE TO LOW BYTE
18829                                   ;
18830   115372  111237  177522                  MOVB    (R2),PCR                ;WRITE TO LOW BYTE
18831   115376  121237  177522                  CMPB    (R2),,PCR               ;BYTE WRITTEN OK?
18832   115402  001003                          BNE     4$                      ;IF NOT,BRANCH
18833   115404  121237  177522                  CMPB    (R2),PCR                ;HIGH BYTE CHANGED?
18834   115410  001405                          BEQ     5$                      ;IF NOT, BRANCH
18835   115412  111237  001124          4$:     MOVB    (R2),$GDDAT             ;EXPECTED PATTERN HIGH BYTE
18836   115416  111237  001124                  MOVB    (R2),$GDDAT             ;LOW BYTE
18837   115422  104051                          ERROR   +51                     ;ERROR PCR READ/WRITE BITS
18838   115424  105722                  5$:     TSTB    (R2)+                   ;INCREMENT POINTER FOR NEW PATTERN
18839   115426  077335                          SOB     R3,1$                   ;DO FOR ALL 4 PATTERNS
```

K11

SEQ 0347

TEST PCR READ/WRITE BITS

```
18840                                           ;
18841                                           ; NOW TRY WORD ADDRESSING
18842                                           ;
18843 115430  012737  052525  177522            MOV     #52525,PCR              ;WRITE A PATERN
18844 115436  022737  052124  177522            CMP     #52124,PCR              ;ALL BUT BITS <8,7,0> OK?
18845 115444  001404                            BEQ     6$                      ;IF SO, BRANCH
18846 115446  112737  052124  001124            MOVB    #52124,$GDDAT           ;EXPECTED PATTERN
18847 115454  104051                            ERROR   +51                     ;ERROR PCR READ/WRITE BITS
18848 115456                                    6$:
      115456  000403                            BR      TST31           ;;GO TO NEXT TEST
18849
18850 115460     000     036     146  SIXBIT: .BYTE 0,36,146,124,52               ;001111,110011,101010,010101
      115463     124     052
18851                                                                           ;BINARY DIVIDE FOR 6 BITS
18852                                           .EVEN
18853
```

```
18855                                   .SBTTL  TEST    BCSR READ/WRITE BITS
18856                                   ;THE SECOND TEST WILL CHECK THAT BCSR<7 5;2-0> CAN BE WRITTEN AND
18857                                   ;READ AS ZEROES AND ONES.  BCSR<14,03> SHOULD BE 0 S.  BCSR<04>
18858                                   ;SHOULD BE CLEARED BY RESET INSTRUCTION.
18859                                   ;ROUTINE TEST
18860                                   ;.      FOR PATTERN=011,010,101 DO
18861                                   ;.      .       WRITE BCSR<7 5>=PATTERN
18862                                   ;.      .       IF BCSR<7-5> NE PATTERN THEN
18863                                   ;.      .       .       ERROR
18864                                   ;.      .       ENDIF
18865                                   ;.      .       WRITE BCSR<2-0>=PATTERN
18866                                   ;.      .       IF BCSR<2-0> NE PATTERN THEN
18867                                   ;.      .       .       ERROR
18868                                   ;.      .       ENDIF
18869                                   ;.      ENDDO
18870                                   ;.      IF BCSR<14,03> NE <0,0> THEN
18871                                   ;.      .       ERROR
18872                                   ;.      ENDIF
18873                                   ;.      LET BCSR<04>=1
18874                                   ;.      IF BCSR<04> NE #1 THEN
18875                                   ;.      .       ERROR
18876                                   ;.      ENDIF
18877                                   ;.      EXECUTE "RESET"
18878                                   ;.      IF BCSR<04> NE 0 THEN
18879                                   ;.      .       ERROR
18880                                   ;.      ENDIF
18881                                   ;.      LET BCSR<04>=0 (THIS BIT IS WRITE ENABLE FOR EAROM)
18882                                   ;ENDROUTINE
18883
18884                                   ;;*****************************************************************
     115466  000004                    TST31:  SCOPE
18885 115470  013737  177520  002730          MOV     BCSR,SAVBR                      ;SAVE BCSR
18886 115476  005037  177520                  CLR     BCSR                            ;CLEAR BCSR
18887                                   ;
18888                                   ; WRITE TO BITS <7-5> AND <2-0>
18889                                   ;
18890 115502  012703  115726                  MOV     #THRBIT,R3                      ;POINTER FOR PATTERN TABLE
18891 115506  005037  001124                  CLR     $GDDAT                          ;CLEAR A LOCATION
18892 115512  012702  000003                  MOV     #3,R2                           ;DO FOR ALL PATTERNS
18893 115516  111337  177520          1$:     MOVB    (R3),BCSR                       ;WRITE TO BITS <7-5>
18894 115522  111337  001124                  MOVB    (R3),$GDDAT                     ;EXPECTED PATTERN
18895 115526  122337  177520                  CMPB    (R3)+,BCSR                      ;BITS WRITTEN OK?
18896 115532  001401                          BEQ     2$                              ;IF SO, BRANCH
18897 115534  104052                          ERROR   .52                             ;ERROR IN BCSR READ/WRITE BITS
18898 115536  111337  177520          2$:     MOVB    (R3),BCSR                       ;WRITE TO BITS <2-0>
18899 115542  111337  001124                  MOVB    (R3),$GDDAT                     ;EXPECTED PATTERN FOR ERRORS
18900 115546  122337  177520                  CMPB    (R3)+,BCSR                      ;BITS WRITTEN OK?
18901 115552  001401                          BEQ     3$                              ;IF SO, BRANCH
18902 115554  104052                          ERROR   .52                             ;ERROR IN BCSR READ/WRITE BITS
18903 115556  077221                  3$:     SOB     R2,1$                           ;CONTINUE TILL ALL PATTERNS DONE
18904                                   ;
18905                                   ; CHECK UNUSED BITS <3>
18906                                   ;
18907 115560  012737  000010  177520          MOV     #10,BCSR                        ;WRITE TO BIT <3>
18908 115566  032737  000010  177520          BIT     #BIT03,BCSR                     ;ALL ZEROES?
18909 115574  001403                          BEQ     4$                              ;IF YES, BRANCH
18910 115576  005037  001124                  CLR     $GDDAT                          ;EXPECTED PATTERN
```

TEST BCSR READ/WRITE BITS

```
18911 115602  104052                          ERROR   ·52                             ;ERROP IN BCSR READ/WRITE BITS
18912                                         ;
18913                                         ; CHECK THAT BIT <4> CLEARS BY RESET
18914                                         ;
18915 115604  052737  000020  177520  4$:     BIS     #BIT04,BCSR                     ;SET BIT 4
18916 115612  032737  000020  177520          BIT     #BIT04,BCSR                     ;WRITTEN OK?
18917 115620  001005                          BNE     5$                              ;IF SO BRANCH
18918 115622  012737  000020  001124          MOV     #BIT04,$GDDAT                   ;EXPECTED PATTERN
18919 115630  104052                          ERROR   ·52                             ;ERROR IN BCSR READ/WRITE BITS
18920 115632  000415                          BR      7$                              ;EXIT TEST
18921 115634  042737  000020  177520  5$:     BIC     #BIT04,BCSR                     ;TRY TO CLEAR BIT 4
18922 115642  032737  000020  177520          BIT     #BIT04,BCSR                     ;CLEARED OK?
18923 115650  001403                          BEQ     6$                              ;IF SO BRANCH
18924 115652  005037  001124                  CLR     $GDDAT                          ;EXPECTED PATTERN
18925 115656  104052                          ERROR   ·52                             ;ERROR IN BCSR READ/WRITE BITS
18926 115660  005737  001206          6$:     TST     $PASS                           ;FIRST PASS?
18927 115664  001014                          BNE     8$                              ;IF  NOT FIRST PASS,EXIT
18928 115666  052737  000020  177520  7$:     BIS     #BIT04,BCSR                     ;SET BIT 4 AGAIN
18929 115674  000005                          RESET                                   ;EXECUTE RESET
18930 115676  032737  000020  177520          BIT     #BIT04,BCSR                     ;BIT 4 CLEARED?
18931 115704  001404                          BEQ     8$                              ;IF YES, BRANCH
18932 115706  104053                          ERROR   ·53                             ;RESET DOESN'T CLEAR BCSR<4>
18933 115710  042737  000020  177520          BIC     #BIT04,BCSR                     ;CLEAR BIT 4
18934 115716  013737  002730  177520  8$:     MOV     SAVBR,BCSR                      ;RESTORE BCSR
18935 115724  000403                          BR      TST32           ;;GO TO NEXT TEST
18936
18937 115726      140       003       100  THRBIT: .BYTE   140,3,100,2,240,5          ;011,010,101 FOR BITS <7 5,2 0>
      115731      002       240       005
18938
```

```
                                        .SBTTL  TEST    16 BIT ROM CHECKSUM TEST
                                        ;ROM'S CHECKSUMS
                                        ;
                                        ;16 BIT ROM TEST
                                        ;THE FIRST TEST WILL CLEAR BCSR<07>, LOAD PCR<14 09> WITH
                                        ;ROM ADDRESS BITS <14 09>, AND CHECK CHECKSUMS OF 16-BIT ROM
                                        ;BY ACCESSING IT THRU BUS ADDRESSES 173000-173776.  THEN WITH
                                        ;BCSR<06;05> BOTH CLEAR, PCR<06 01> USED AS ADDRESS BITS 14-09,
                                        ;THE SAME THING WILL BE DONE BY ADDRESSING 16 BIT ROM THRU BUS ADDRESSES
                                        ;165000 165776. THE RESULTS SHOULD BE THE SAME AND SHOULD COMPARE
                                        ;WITH THAT STORED IN THE BOOT AND DIAGNOSTIC ROM.
                                        ;
                                        ;BCSR    <07>    DISABLE 17773000
                                        ;        <06>    DISABLE 17765000
                                        ;        <05>    ROM SOCKET 3 AT 17765000
                                        ;
                                        ;ROUTINE TEST
                                        ;.      LET BCSR<7,6,5>=0,0,0
                                        ;.      DO FOR R1 FROM #0 TO #31. BY #1 DO
                                        ;.      .       LET PCR<14-09>=R1
                                        ;.      .       DO FOR R2 FROM #0 TO %776 BY 2
                                        ;.      .       .       CALCULATE CHECKSUM THRU 173000
                                        ;.      .       ENDDO
                                        ;.      ENDDO
                                        ;.      IF CHECKSUM NE #0 THEN
                                        ;.      .       ERROR
                                        ;.      ENDIF
                                        ;.      DO FOR R1 FROM #0 TO #31. BY #1
                                        ;.      .       LET PCR<06-01>=R1
                                        ;.      .       DO FOR R2 FROM #0 TO #776 BY 2
                                        ;.      .       .       CALCULATE CHECKSUM THRU 165000
                                        ;.      .       ENDDO
                                        ;.      ENDDO
                                        ;.      IF CHECKSUM NE #0 THEN
                                        ;.      .       ERROR
                                        ;.      ENDIF
                                        ;ENDROUTINE
                                        ;
                                        ;;*****************************************************************
                                        TST32:  SCOPE
18979 115736  013737  177520  002730          MOV     BCSR,SAVBR                      ;SAVE BCSR
18980 115744  042737  000340  177520          BIC     #BIT07!BIT06!BIT05,BCSR         ;READ 16 BIT ROM
18981 115752  052737  001000  177520          BIS     #1000,BCSR                      ; enable HOB, for APT
                                        ;
                                        ; CALCULATE LOW BYTE CHECKSUM'S THRU 173000 AND 165000
                                        ;
18985 115760  005001                    1$:     CLR     R1                              ;PAGE COUNT FOR ALL 8K
18986 115762  005037  177522                    CLR     PCR                             ;CLEAR PAGE CONTROL REGISTER
18987 115766  005037  002724                    CLR     ACTCHS                          ;CLEAR CHECKSUM AT 173000
18988 115772  005037  001160                    CLR     $TMP0                           ;AT 165000
18989 115776  005002                            CLR     R2                              ;CLEAR COUNTER THRU A PAGE
18990 116000  016203  173000            2$:     MOV     173000(R2),R3                   ;GET LOW BYTE THRU 173000
18991 116004  016204  165000                    MOV     165000(R2),R4                   ;THRU 165000
18992 116010  060337  002724                    ADD     R3,ACTCHS                       ;CALCULATE CHECKSUM THRU 173000
18993 116014  060437  001160                    ADD     R4,$TMP0                        ;CALCULATE CHEKCSUM THRU 165000
18994 116020  005722                            TST     (R2)+                           ;GET NEXT WORD
18995 116022  022702  000776                    CMP     #776,R2                         ;WORD BEFORE LAST?
```

B12

TEST - 16 BIT ROM CHECKSUM TEST

```
18996 116026  001004                    BNE     3$                      ;IF NOT, BRANCH
18997 116030  020337  177522              CMP     R3,PCR                  ;PAGE NUMBER STORED OK?
18998 116034  001401                    BEQ     3$                      ;IF YES, BRANCH
18999 116036  104054                    ERROR   ·54                     ;PAGE NUMBER STORED WRONG
19000 116040  022702  001000    3$:     CMP     #1000,R2                ;LAST WORD IN A PAGE?
19001 116044  003355                    BGT     2$                      ;IF NOT, BRANCH
19002 116046  005737  002724            TST     ACTCHS                  ;CHECKSUM 0?
19003 116052  001401                    BEQ     4$                      ;IF YES, BRANCH
19004 116054  104054                    ERROR   ·54                     ;IN CHECKSUM AT 173000
19005 116056  005737  001160    4$:     TST     $TMP0                   ;CHECKSUM 0 AT 165000?
19006 116062  001401                    BEQ     5$                      ;IF YES, BRANCH
19007 116064  104054                    ERROR   ·54                     ;IN CHECKSUM AT 165000
19008 116066  062701  000002    5$:     ADD     #2,R1                   ;GET NEW PAGE
19009 116072  110137  177522            MOVB    R1,PCR                  ;STORE IN PCR<6 1>
19010 116076  110137  177523            MOVB    R1,PCR+1                ;STORE IN PCR<14-9>
19011 116102  022701  000160            CMP     #160,R1                 ;ALL  PAGES IN R2 FROM BIT1?
19012 116106  003327                    BGT     1$                      ;IF NOT BRANCH
19013                                   ;
19014                                   ; CHECK THE LAST 2K OF ASCII TEXT
19015                                   ;
19016 116110  005037  002724    LASTCH: CLR     ACTCHS                  ·CLEAR CHECKSUM AT 173000
19017 116114  005037  001160            CLR     $TMP0                   ;AT 165000
19018 116120  005002            1$:     CLR     R2                      ;CLEAR COUNTER THRU A PAGE
19019 116122  016203  173000    2$:     MOV     173000(R2),R3           ;GET LOW BYTE THRU 173000
19020 116126  016204  165000            MOV     165000(R2),R4           ;THRU 165000
19021 116132  060337  002724            ADD     R3,ACTCHS               ;CALCULATE CHECKSUM THRU 173000
19022 116136  060437  001160            ADD     R4,$TMP0                ;CALCULATE CHEKCSUM THRU 165000
19023 116142  005722                    TST     (R2)+                   ;GET NEXT WORD
19024 116144  022702  001000    3$:     CMP     #1000,R2                ;LAST WORD IN A PAGE?
19025 116150  003364                    BGT     2$                      ;IF NOT, BRANCH
19026 116152  022701  000176            CMP     #176,R1                 ;LAST PAGE?
19027 116156  001005                    BNE     4$                      ;IF NOT, BRANCH
19028 116160  023737  173774  177522    CMP     173774,PCR              ;PROPER PAGE NUMBER?
19029 116166  001401                    BEQ     4$                      ;IF YES, BRANCH
19030 116170  104054                    ERROR   ·54                     ;ERROR IN ROM
19031 116172  062701  000002    4$:     ADD     #2,R1                   ;GET NEW PAGE
19032 116176  110137  177522            MOVB    R1,PCR                  ;STORE IN PCR<6-1>
19033 116202  110137  177523            MOVB    R1,PCR+1                ;STORE IN PCR<14-9>
19034 116206  022701  000200            CMP     #200,R1                 ;ALL  PAGES IN R2 FROM BIT1?
19035 116212  003342                    BGT     1$                      ;IF NOT BRANCH
19036 116214  005737  002724            TST     ACTCHS                  ;CHECKSUM 0?
19037 116220  001401                    BEQ     5$                      ;IF YES, BRANCH
19038 116222  104054                    ERROR   ·54                     ;IN CHECKSUM AT 173000
19039 116224  005737  001160    5$:     TST     $TMP0                   ;CHECKSUM 0 AT 165000?
19040 116230  001401                    BEQ     6$                      ;IF YES, BRANCH
19041 116232  104054                    ERROR   ·54                     ;IN CHECKSUM AT 165000
19042 116234  005037  177522    6$:     CLR     PCR                     ;CLEAR PCR
19043 116240  042737  001000  177520    BIC     #1000, BCSR             ; reset HOB for APT
19044 116246  013737  177520  002730    MOV     BCSR,SAVBR              ;RESTORE BCSR
19045
19046
19047
```

```
19049                                   .SBTTL  TEST  - 8 BIT EEROM CHECKSUM (105dec bytes) TEST
19050                                   ;THIS TEST WILL CLEAR BCSR<6>, SET BCSR<5>, LOAD PCR<5-1>
19051                                   ;WITH ADDRESS BITS 13 09, AND CHECK CRC PATTERNS OF 8-BIT EEROM BY
19052                                   ;ACCESSING IT THRU ADDRESSES 165000-165776. THIS TEST VERIFIES THE
19053                                   ;RESPONSE ONLY OF THE BASE AREA.
19054                                   ;ROUTINE TEST
19055                                   ;.      LET BCSR<5>=1
19056                                   ;.      LET OLDCRC=#0
19057                                   ;.      LET PCR<05-01>=#0
19058                                   ;.      CALCULATE CHECKSUM FOR THE FIRST 320 LOCATIONS
19059                                   ;.      IF RESULTING CHECKSUM NOT ZERO THEN
19060                                   ;.          ERROR
19061                                   ;.      ENDIF
19062                                   ;ENDROUITNE
19063
19064                                   ;;*****************************************************************
     116254  000004                    TST33:  SCOPE
19065 116256  013737  177520  002730          MOV     BCSR,SAVBR                      ;SAVE BCSR
19066 116264  042737  000100  177520          BIC     #BIT06,BCSR                     ;ENABLE INTERNAL RESPONSE
19067 116272  052737  000040  177520          BIS     #BIT05,BCSR                     ;SELECT 8-BIT ROM
19068 116300  005037  001160                  CLR     $TMP0                           ;CLEAR SUM
19069                                   ;
19070                                   ; CALCULATE LOW BYTE CHECKSUM'S THRU 165000
19071                                   ;
19072 116304  005001                    1$:     CLR     R1                              ;PAGE COUNT FOR ALL 8K
19073 116306  005037  177522                    CLR     PCR                             ;CLEAR PAGE CONTROL REGISTER
19074 116312  005037  002724            2$:     CLR     ACTCHS                          ;CLEAR CHECKSUM AT 165000
19075 116316  005002                            CLR     R2                              ;CLEAR COUNTER THRU A PAGE
19076 116320  116204  165000            3$:     MOVB    165000(R2),R4                   ;GET A BYTE THRU 165000
19077 116324  060437  001160                    ADD     R4,$TMP0                        ;CALCULATE CHEKCSUM THRU 165000
19078 116330  005722                            TST     (R2)+                           ;GET NEXT WORD
19079 116332  022702  000316                    CMP     #316,R2                         ;WORD BEFORE LAST?
19080 116336  001004                            BNE     4$                              ;IF NOT, BRANCH
19081 116340  122704  000252                    CMPB    #252,R4                         ;314 SHOULD HAVE 252
19082 116344  001765                            BEQ     3$                              ;IF YES, BRANCH
19083 116346  104055                            ERROR   +55                             ;PAGE NUMBER STORED WRONG
19084 116350  022702  000322            4$:     CMP     #322,R2                         ;LAST WORD IN A PAGE?
19085 116354  003361                            BGT     3$                              ;IF NOT, BRANCH
19086 116356  113737  165010  001162          MOVB    165010,$TMP1                    ;STORE SIZE,<3>=1 2K
19087 116364  105737  001160                  TSTB    $TMP0                           ;CHECKSUM 0 AT 165000?
19088 116370  001401                            BEQ     5$                              ;IF YES, BRANCH
19089 116372  104055                            ERROR   +55                             ;IN CHECKSUM AT 165000
19090 116374  005037  177522            5$:     CLR     PCR
19091 116400  013737  002730  177520          MOV     SAVBR,BCSR                      ;RESTORE BCSR
19092
```

```
                                .SBTTL  TEST - 8 BIT EAROM READ WRITE   TEST

                                ; This test writes a checkerboard pattern of alternating 1's and 0 s in
                                ; 8-bit EAROM.
                                ; All base area 316 octal locations are saved. (105 dec plus 4 customer loc.)
                                ; It will run only if bit 12 is set in software switch register.
                                ; This is the eerom test enable, setable at runtime by an operator.  Loop
                                ; on error is prohibited, as it would be possible to destroy the EEROM
                                ; in a relativly short time with repeated write cycles.  The option to
                                ; continue beyond a single error is selected by by asserting the halt on
                                ; error switch at which time the operator may truncate the test.
                                ;
                                ; The switches which could affect the EEROM's life are masked out and
                                ; restored in the test immediatly following this one. <<<<<*********
                                ;
                                ; This test will be not be done if there is an APT system present, or if
                                ; bit #12 of the Software Switch register is not set.  If Bit 12 is set and
                                ; there is no APT system present (or in load mode), the EEROM test will be
                                ; run 1 time, all subsequent passes will not include the EEROM test.
                                ;
                                ;
                                                        -- SECTION 1
                                ;+  SAVE BASE AREA
                                ;+
                                ;+
                                ;+
                                ;+ TEST                          SECTION 2
                                ;+
                                ;+ 2K PROCEDURE
                                ;+    do while NOT$DONE$2000
                                ;+       Write 10101010 (252 oct)
                                ;+       delay
                                ;+       read  10101010 (252 oct)
                                ;+       update error list if incorrect read
                                ;+    enddo
                                ;+    do while NOT$DONE$2000
                                ;+       write 01010101 (125 oct)
                                ;+       delay
                                ;+       read  01010101 (125 oct)
                                ;+       update error list
                                ;+    enddo
                                ;+

                                ; CLEANUP -- SECTION 3
                                ; communicate failure mode and loc info
                                ; restore base area (105) + 4 customer locations
                                ; restore registers
                                ; RESTORE BACK BASE AREA
                                ;
                                ;
                                ;*****************************************************************************

                                ;;*************************************************************
 116406  000004                TST34:  SCOPE
 116410  017737  062524  003050         MOV     @SWR,SAVSWR             ;SAVE SOFTSWITCH SETTINGS
 116416  005737  001206                 TST     $PASS                   ;do only once, iff all cond. O.K.
 116422  001006                        BNE     1$                      ;  on first pass ( 0)
```

E12


```
19150 116424  000240                          NOP                     ; no, do againa
19151                                 ;       CMPB    @APTENV,$ENV    ;IN APT MODE?
19152                                 ;       BEQ     1$              ; <IF YES, EXIT TEST>
19153 116426  032777  010000  062504          BIT     @BIT12,@SWR     ; bit 12 (do EEROM test) set?
19154 116434  001016                          BNE     2$              ;
19155 116436  000240                          NOP
19156 116440  000137  116770          1$:     JMP     TSTEND          ; exit po'nt for program
19157 116444     015     012     116  33$:    .ASCIZ  <15><12>/NOW TESTING EEROM/<15><12>
      116447     117     127     040
      116452     124     105     123
      116455     124     111     116
      116460     107     040     105
      116463     105     122     117
      116466     115     015     012
      116471     000
19158                                         .EVEN
19159                                 ; save base area, switch registers
19160
19161 116472  104401  116444          2$:     TYPE    ,33$            ; INDICATE EEROM TEST UNDERWAY
19162 116476  017737  062436  003050          MOV     @SWR,SAVSWR     ; save softswitch settings
19163 116504  042777  041777  062426          BIC     @41777, @SWR    ; clear any interferring
19164 116512  013737  177520  002730          MOV     BCSR,SAVBR      ;SAVE REGISTER
19165 116520  052737  001060  177520          BIS     @1060,BCSR      ;ENABLE INTERNAL ROM'S
19166 116526  000240                          NOP                     ;ENABLE 8-BIT ROM, HOB
19167 116530  012701  143762                  MOV     @$POWER+10,R1   ;LAST LOCATION IN PROGRAM
19168 116534  005037  177522                  CLR     PCR             ;START WITH PAGE 0
19169 116540  005002                          CLR     R2              ;DISPLACEMENT 0
19170 116542  016221  165000          3$:     MOV     165000(R2),(R1)+        ;STORE A WORD
19171 116546  005722                          TST     (R2)+           ;GET NEXT WORD
19172 116550  022702  000334                  CMP     @334,R2         ;ALL 332 LOCATIONS DONE?
19173 116554  003372                          BGT     3$              ;IF NOT, CONTINUE
19174
19175                                 ; test 2K section
19176
19177 116556  112703  000252                  MOVB    @252, R3        ; first pattern, 10 101 010
19178 116562  005037  003022                  CLR     ERRCNT          ; zero the cumulative error count
19179
19180 116566  000240                  201$:   NOP
19181 116570  005037  177522                  CLR     PCR             ; clears page register (start @ 165000)
19182 116574  000240                  202$:   NOP
19183 116576  005002                          CLR     R2              ; first location each page of EEROM
19184 116600  000240                  203$:   NOP
19185 116602  110362  165000                  MOVB    R3, 165000(R2)  ; write the test pattern
19186 116606  004737  116754                  JSR     PC, DELAY       ; wait for write time
19187 116612  120362  165000                  CMPB    R3,165000(R2)   ; read back the written word
19188 116616  001413                          BEQ     204$            ; if O. K. readback, skip handle error
19189
19190 116620  005237  003022                  INC     ERRCNT          ; update cumulative error count
19191 116624  116237  165000  001126          MOVB    165000(R2),$BDDAT       ; put the read data in display area
19192 116632  013704  177522                  MOV     PCR,R4
19193 116636  010237  001122                  MOV     R2,$BDADR       ; also address location info
19194 116642  104134                          ERROR   +134            ; do EEROM read/write error report
19195 116644  000240                          NOP                     ; continue testing
19196
19197 116646  005722                  204$:   TST     (R2)+
19198 116650  022702  000776                  CMP     @776,R2         ; last location checked in a page
19199 116654  003351                          BGT     203$            ; continue till all page loc. tested
```

F12

TEST - 8 BIT EAROM READ-WRITE - TEST

```
19200 116656  062737  000002  177522          ADD     #2,PCR                  ; change PCR every 512 dec. bytes
19201 116664  122737  000020  177522          CMPB    #20,PCR                 ; 8 (256 Byte pgs in 2K) * 2 (holes)=20
19202 116672  001340                          BNE     202$                    ; finish page
19203 116674  122703  000125                  CMPB    #125,R3                 ; test for both patterns written
19204 116700  001403                          BEQ     205$                    ; Exit point
19205 116702  112703  000125                  MOVB    #125,R3                 ; invert the test pattern
19206 116706  000727                          BR      201$                    ; do the test over with new pattern
19207
19208
19209 116710  000240                  205$:   NOP                             ; test over
19210
19211 116712  012701  143762                  MOV     #$POWER+10,R1           ;LAST LOCATION IN PROGRAM
19212 116716  005037  177522                  CLR     PCR                     ;START WITH PAGE 0
19213 116722  005002                          CLR     R2                      ;DISPLACEMENT 0
19214 116724  012162  165000          11$:    MOV     (R1)+,165000(R2)        ;RESTORE A WORD
19215 116730  004737  116754                  JSR     PC, DELAY               ; wait for write time
19216 116734  005722                          TST     (R2)+                   ;GET NEXT WORD
19217 116736  022702  000334                  CMP     #334,R2                 ;ALL 332 LOCATIONS DONE?
19218 116742  003370                          BGT     11$                     ;IF NOT, CONTINUE
19219 116744  013737  002730  177520          MOV     SAVDR,BCSR              ;RESTORE REGISTER
19220 116752  000406                          BR      TSTEND                  ; goto next test (swr restored there)
19221
19222                               ;          - - - - -  SUBROUTINES AREA    -        -----
19223                               ; the following subroutine causes a delay of a certain no of milliseconds
19224                               ; the number of ms. delayed is based on the type of EEROM as indicated by R5
19225
19226 116754  010046                  DELAY:  MOV     R0, (SP)                ; save R0
19227 116756  012700  023420                  MOV     #10000., R0             ; ALL ELSE = 10 MS DELAY
19228 116762  077001                  1$:     SOB     R0, 1$                  ; actual delay
19229 116764  012600                          MOV     (SP)+, R0               ; restore R0
19230 116766  000207                          RTS     PC
19231
19232                               ;-  - ------- -- ------  --   -      -- ---   --      --
19233 116770  000240                  TSTEND: NOP                             ; everything done
19234
```

```
                                        .SBTTL  TEST    LKS BIT 7
                                        ; This test restores the software switch register used in the EEROM
                                        ; test above, and therefore must follow it directly !! <<<<<<<<<*****
                                        ;THESE TESTS WILL CHECK FUNCTIONALITY OF LKS<7-6>
                                        ;
                                        ;ACCESS LKS, CHECK THAT READY LKS<07> CAN BE 0 AND 1.
                                        ;
                                        ;LKS     <07>    LINE CLOCK MONITOR
                                        ;        <06>    LINE CLOCK INTERRUPT ENABLE
                                        ;ROUTINE TEST
                                        ;IF UFD AND LKS IS DISABLED THEN
                                        ;.       EXIT TEST
                                        ;ENDIF
                                        ;.       READ LKS TO SEE IF IT TIMEOUTS
                                        ;.(CHECK LKS<07>)
                                        ;.       LET R1=#77777(WORST CASE COUNTER FOR SLOW CLOCK)
                                        ;.       LET 100=#ADDRESS OF LINE CLOCK_INTERRUPT
                                        ;.       CLEAR INTERRUPT FLAG
                                        ;.       DO 3 TIMES
                                        ;.       .       REPEAT
                                        ;.       .       .       DECREMENT R1
                                        ;.       .       UNTIL R1 NE #0 OR LOW BYTE OF LKS LT #0
                                        ;.       .       IF LOW BYTE OF LKS GE #0 THEN (READY DIDN'T COME UP)
                                        ;.       .       .       ERROR
                                        ;.       .       ENDIF
                                        ;.       ENDDO
                                        ;.       IF INTERRUPT_FLAG NE #0 THEN (INTERUPT W/O LKS<6>=1)
                                        ;.       .       ERROR
                                        ;.       ENDIF
                                        ;ENDROUTINE
                                        ;;*****************************************************************
    116772  000004                     TST35:  SCOPE
19268 116774  013777  003050  062136            MOV     SAVSWR,@SWR                     ; restore here FROM EEROM test
19269 117002  032737  000100  000052            BIT     #BIT06,@#52                     ;UFD MODE?
19270 117010  001404                            BEQ     1$                              ;IF NOT, GO DO TEST
19271 117012  032737  010000  177520            BIT     #BIT12,BCSR                     ;LKS DISABLED?
19272 117020  001056                            BNE     TST36           ;;IF DISABLED, EXIT TEST
19273                                   ;
19274                                   ; TRY TO ACCESS LKS WITHOUT TIMEOUT
19275                                   ;
19276 117022  013737  000004  001160    1$:     MOV     ERRVEC,$TMP0                    ;SAVE TIMEOUT VECTOR
19277 117030  012737  117052  000004            MOV     #2$,ERRVEC                      ;POINT NEW VECTOR TO PROGRAM
19278 117036  012737  000340  000006            MOV     #340,ERRVEC+2                   ;AT PRIORITY 7
19279 117044  005737  177546                    TST     LKS                             ;ACCESS LKS
19280 117050  000403                            BR      3$                              ;IF NO TIMEOUT, CONTINUE
19281 117052  104056                    2$:     ERROR   +56                             ;TIMEOUT READING LKS
19282 117054  005726                            TST     (SP)+                           ;ADJUST STACK POINTER
19283 117056  005726                            TST     (SP)+
19284 117060  013737  001160  000004    3$:     MOV     $TMP0,ERRVEC                    ;RESTORE TIMEOUT VECTOR
19285                                   ;
19286                                   ; CHECK THAT READY BIT LKS<7> CAN BE 1
19287                                   ;
19288 117066  042737  000200  177546            BIC     #BIT07,LKS                      ; START WITH CLEAN SLATE
19289 117074  005037  002722                    CLR     LKSFL                           ;CLEAR INTERRUPT FLAG
19290 117100  012737  137062  000100            MOV     #LKSINT,100                     ;POINT VECTOR TO INERRUPT ROUTINE
19291 117106  012737  000340  000102            MOV     #340,102                        ;AT PRIORITY 7
```

H12

TEST LKS BIT 7

```
 19292 117114  012702  000003            MOV     #3,R2                   ;DO 3 TIMES TO SYNCHRONISE
 19293 117120  012701  077777      4$:   MOV     #77777,R1               ;COUNTER FOR SLOW CLOCKS
 19294 117124  105737  177546      5$:   TSTB    LKS                     ;READY LKS<7>=1?
 19295 117130  100401                    BMI     6$                      ;IF SO, DO NEXT LOOP
 19296 117132  077104                    SOB     R1,5$                   ;OTHERWISE, GO THRU COUNT
 19297 117134  105737  177546      6$:   TSTB    LKS                     ;WAS READY 1?
 19298 117140  100401                    BMI     7$                      ;IF YES, BRANCH
 19299 117142  104057                    ERROR   +57                     ;LKS<07> DOES NOT BECOME 1
 19300 117144  077213                7$: SOB     R2,4$                   ;DO ALL 3 TIMES
 19301 117146  005737  002722      8$:   TST     LKSFL                   ;ANY INTERRUPTS W/O LKS<6>=1?
 19302 117152  001401                    BEQ     TST36           ;;IF NONE, EXIT TEST
 19303 117154  104061                    ERROR   +61                     ;ILLEGAL CLOCK INTERRUPTS
 19304
```

```
                                .SBTTL  TEST  - LKS INTERRUPT PRIORITY
                                ;CHECK THAT LKS INTERRUPTS HAPPEN AT PRIORITY 5 CLEARING LKS<07>
                                ;AND DON'T HAPPEN AT PRIORITY 6.
                                ;ROUTINE TEST
                                ;IF UFD AND LKS IS DISABLED THEN
                                ;.      EXIT TEST
                                ;ENDIF
                                ;.      SET PRIORITY TO 5
                                ;.      CLEAR INTERRUPT_FLAG
                                ;.      LET LKS<06>=#1 (ENABLE INTERRUPTS)
                                ;.      SET COUNTER TO WAIT FOR 3 INTERRUPTS
                                ;.      REPEAT
                                ;.      .       DECREMENT COUNTER
                                ;.      UNTIL INTERRUPT_FLAG EQ #3 OR COUNTER EQ #0
                                ;.      CLEAR LKS<06>
                                ;.      IF LKS<07> EQ #1 THEN
                                ;.      .       ERROR (WAS NOT CLEARED ON INTERRUPT)
                                ;.      ENDIF
                                ;.      IF COUNTER LT TIME_REQUIRED_FOR_3_INTERRUPTS_FOR_800HZ
                                ;.      .       ERROR (INTERRUPTS NEVER GO LOW)
                                ;.      ENDIF
                                ;.      IF INTERRUPT_FLAG LT #3 THEN
                                ;.      .       ERROR (INTERRUPTS DON'T HAPPEN)
                                ;.      ENDIF
                                ;.      CLEAR INTERUPT_FLAG
                                ;.      WAIT FOR LKS<7>=1
                                ;.      LET LKS<7>=0
                                ;.      IF LKS<7> NE #0 THEN
                                ;.      .       ERROR (LKS<7> NOT CLEARED
                                ;.      ENDIF
                                ;.      SET PRIORITY TO 6
                                ;.      SET COUNTER TO 1 SLOW CLOCK INTERRUPT
                                ;.      SET LKS<06>
                                ;.      REPEAT
                                ;.      .       DECREMENT COUNTER
                                ;.      UNTIL COUNTER EQ #0 OR INTERRUPT_FLAG NE #0
                                ;.      IF INTERRUPT_FLAG NE #0 THEN
                                ;.      .       ERROR (INTERRUPT WAS AT WRONG PRIORITY)
                                ;.      ENDIF
                                ;.      RESTORE ORIGINAL PRIORITY
                                ;ENDROUTINE
                                ;
                                ;ROUTINE LINE_CLOCK_INTERUPT
                                ;.      INCREMENT INTERRUPT_FLAG
                                ;ENDROUTINE

                                ;;*****************************************************************
117156  000004                  TST36:  SCOPE
117160  032737  000100  000052          BIT     #BIT06,@#52                     ;UFD MODE?
117166  001404                        BEQ     1$                              ;IF NOT, GO DO TEST
117170  032737  010000  177520        BIT     #BIT12,BCSR                     ;LKS DISABLED?
117176  001132                        BNE     TST37           ;;IF DISABLED, EXIT TEST
                                ;
                                ; WAIT FOR 3 INTERRUPTS AND CHECK LKS<7> TO BE 0 AFTER INTERRUPT
                                ;
117200  042737  000100  177546  1$:     BIC     #BIT06,LKS                      ; FROM END OF TEST 42?? PROBLEM
117206  005037  002722                  CLR     LKSFL                           ;CLEAR INTERRUPT FLAG
```

J12

TEST - LKS INTERRUPT PRIORITY

```
19362 117212  012737  137062  000100          MOV     #LKSINT,100             ;POINT VECTOR TO ROUTINE
19363 117220  012701  077777                  MOV     #77777,R1               ;COUNTER FOR SLOW CLOCK
19364 117224  052737  000100  177546          BIS     #BIT06,LKS              ;SET INTERRUPT ENABLE BIT
19365 117232  032737  000100  177546          BIT     #BIT06,LKS              ;BIT SET OK?
19366 117240  001001                          BNE     2$                      ;IF YES, BRANCH
19367 117242  104131                          ERROR   +131                    ;ERROR WRITING 1 TO LKS<6>
19368 117244  106427  000240          2$:     MTPS    #240                    ;SET PRIORITY TO 5
19369 117250  022737  000003  002722  3$:     CMP     #3,LKSFL                ;3 INTERRUPTS HAPPENED?
19370 117256  001401                          BEQ     4$                      ;IF YES, BRANCH
19371 117260  077105                          SOB     R1,3$                   ;STAY IN A LOOP
19372                                 ;
19373                                 ; DISABLE INTERRUPTS AND CHECK THAT PROPER CONDITIONS ARE MET
19374                                 ;
19375 117262  042737  001000  177520  4$:     BIC     #1000,BCSR              ;DISABLE HALT ON BREAK
19376 117270  106427  000340                  MTPS    #340                    ;RAISE PRIORITY
19377 117274  042737  000100  177546          BIC     #BIT06,LKS              ;DISABLE INTERRUPTS
19378 117302  032737  000200  177546          BIT     #BIT07,LKS              ;LKS<7> CLEARED AFTER INTERRUPTS?
19379 117310  001401                          BEQ     5$                      ;IF 0, BRANCH
19380 117312  104062                          ERROR   +62                     ;INTERRUPTS DON'T CLEAR LKS<7>
19381 117314  105737  177546          5$:     TSTB    LKS                     ;LKS<7>=1?
19382 117320  100375                          BPL     5$                      ;IF NOT, WAIT
19383 117322  005037  177546                  CLR     LKS                     ;CLEAR LKS<7>
19384 117326  032737  000200  177546          BIT     #BIT07,LKS              ;LKS<7> CLEARED?
19385 117334  001401                          BEQ     6$                      ;IF YES, BRANCH
19386 117336  104060                          ERROR   +60                     ;LKS<7> NOT CLEARED ON WRITE
19387 117340  032737  000100  177546  6$:     BIT     #BIT06,LKS              ;LKS<6>=0?
19388 117346  001401                          BEQ     7$                      ;IF YES, BRANCH
19389 117350  104131                          ERROR   +131                    ;ERROR WRITING 0 TO LKS<6>
19390 117352  052737  001000  177520  7$:     BIS     #1000,BCSR              ;ENABLE HALT ON BREAK
19391 117360  022701  077737                  CMP     #77737,R1               ;COUNTER AT LESS THAN 800HZ?
19392 117364  002001                          BGE     8$                      ;IF NOT, BRANCH
19393 117366  104063                          ERROR   +63                     ;READY LINE DOES NOT GO LOW
19394 117370  022737  000003  002722  8$:     CMP     #3,LKSFL                ;DID 3 INTERRUPTS HAPPEN?
19395 117376  001404                          BEQ     9$                      ;IF YES, BRANCH
19396 117400  012737  000003  001124          MOV     #3,$GDDAT               ;3 INTERRUPTS EXPECTED
19397 117406  104064                          ERROR   +64                     ;INTERRUPTS DON'T HAPPEN
19398                                 ;
19399                                 ; CHECK WHETHER INTERRUPTS HAPPEN AT PRIORITY 6
19400                                 ;
19401 117410  005037  002722          9$:     CLR     LKSFL                   ;CLEAR INTERRUPT FLAG
19402 117414  106427  000300                  MTPS    #300                    ;RAISE PRIORITY TO 6
19403 117420  012701  077777                  MOV     #77777,R1               ;COUNTER FOR SLOW CLOCK
19404 117424  052737  000100  177546          BIS     #BIT06,LKS              ;SET INTERRUPT ENABLE BIT
19405 117432  005737  002722          10$:    TST     LKSFL                   ;ANY INTERRUPTS?
19406 117436  001001                          BNE     11$                     ;IF YES, EXIT LOOP
19407 117440  077104                          SOB     R1,10$                  ;CONTINUE WITH COUNT
19408 117442  005737  002722          11$:    TST     LKSFL                   ;ANY INTERRUPTS?
19409 117446  001404                          BEQ     12$                     ;IF NO, BRANCH
19410 117450  012737  000005  001124          MOV     #5,$GDDAT               ;STORE PRIORITY FOR TYPE OUT
19411 117456  104065                          ERROR   +65                     ;INTERUPTS HAPPEN AT WRONG PRIORITY
19412 117460  106427  000340          12$:    MTPS    #340                    ;RESTORE PRIORITY
19413
```

K12

·Q 0360

```
                                .SBTTL  TEST    LINE CLOCK DISABLE

                                ;LINE CLOCK DISABLE(*)
                                ;THIS TEST WILL CHECK THAT BCSR<12> DISABLES RESPONSE
                                ;OF LKS REGISTER.
                                ;
                                ;BCSR   <12>    LINE CLOCK STATUS REGISTER DISABLE
                                ;
                                ;
                                ;ROUTINE TEST
                                ;
                                ;IF UFD AND LKS IS NOT DISABLED THEN
                                ;.      EXIT TEST
                                ;ENDIF
                                ;
                                ;.      WRITE BCSR<12>=1
                                ;.      LET 4=ADDRESS OF LKS_TRAP
                                ;.      LET TRAP_LKS=0
                                ;.      READ LKS
                                ;.      IF TRAP_LKS NE 1 THEN
                                ;.              ERROR
                                ;.      ENDIF
                                ;
                                ;ENDROUTINE
                                ;
                                ;
                                ;ROUTINE LKS_TRAP
                                ;
                                ;.      LET TRAP_LKS=1
                                ;.      LET BCSR<12>=0
                                ;
                                ;RTI
                                ;

                                ;;**********`**********************************************************
    117464  000004              TST37:  SCOPE
19450 117466  032737  000100  000052        BIT     @#BIT06,@#52                    ;UFD MODE?
19451 117474  001404                        BEQ     1$                              ;IF NOT, BRANCH
19452 117476  032737  010000  177520        BIT     @#BIT12,BCSR                    ;LKS DISABLED?
19453 117504  001052                        BNE     TST40           ;;IF DISABLED, EXIT TEST
19454                                   ;
19455                                   ; CHECK BCSR<12> TO BE 0 AND 1
19456                                   ;
19457 117506  013737  177520  002730  1$:   MOV     BCSR,SAVBR                      ;SAVE BCSR REGISTER
19458 117514  042737  010000  177520        BIC     @#BIT12,BCSR                            ;CLEAR BCSR
19459 117522  032737  010000  177520        BIT     @#BIT12,BCSR                    ;<12>=0?
19460 117530  001403                        BEQ     2$                              ;IF OK, BRANCH
19461 117532  005037  001124                CLR     $GDDAT                          ;CLEAR EXPECTED PATTERN
19462 117536  104052                        ERROR   .52                             ;ERROR BCSR READ/WRITE BITS
19463 117540  052737  010000  177520  2$:   BIS     @#BIT12,BCSR                    ;SET BIT 12
19464 117546  032737  010000  177520        BIT     @#BIT12,BCSR                    ;GOT SET OK?
19465 117554  001004                        BNE     3$                              ;IF OK, BRANCH
19466 117556  012737  010000  001124        MOV     @#BIT12,$GDDAT                  ;EXPECTED PATTERN
19467 117564  104052                        ERROR   .52                             ;ERROR BCSR READ/WRITE BITS
19468                                   ;
19469                                   ; TRY TO ACCESS LKS TO GET A TIMEOUT WITH BCSR<12>=1
19470                                   ;
```

L12

SEQ 0361

TEST - LINE CLOCK DISABLE

```
 1947  117566  013701  000004          3$:     MOV     ERRVEC,R1               ;SAVE TIMEOUT VECTOR
 19472 117572  012737  117614  000004          MOV     #4$,ERRVEC              ;POINT TO PROGRAM AREA
 19473 117600  012737  000340  000006          MOV     #340,ERRVEC+2           ;AT PRIORITY 7
 19474 117606  005737  177546                  TST     LKS                     ;ACCESS LKS REGISTER
 19475 117612  104066                          ERROR   +66                     ;BCSR<12> DOES NOT DISABLE LKS
 19476 117614  005726                  4$:     TST     (SP)+                   ;RESTORE STACK POINTER
 19477 117616  005726                          TST     (SP)+
 19478 117620  010137  000004                  MOV     R1,ERRVEC               ;RESTORE TIMEOUT VECTOR
 19479 117624  013737  002730  177520          MOV     SAVBR,BCSR              ;RESTORE BCSR
 19480
 19481
```

```
                                .SBTTL  TEST    UNCONDITIONAL CLOCK LINE INTERRUPTS
                                ;UNCONDITIONAL CLOCK LINE INTERRUPTS(*)
                                ;THIS TEST WILL CHECK THAT SETTING BCSR<13> TO 1 WILL
                                ;REQUEST INTERRUPTS WHENEVER A CLOCK LINE IS ASSERTED.  THIS
                                ;SHOULD HAPPEN WITHOUT ACCESSING LKS, THEREFORE, EVEN WITH BCSR<12>=1
                                ;INTERRUPTS SHOULD HAPPEN.
                                ;
                                ;BCSR   <13>    FORCE LINE CLOCK INTERRUPT ENABLE
                                ;

                                ;ROUTINE TEST
                                ;IF UFD AND FORCE LKS NOT DISABLED THEN
                                ;.      EXIT TEST
                                ;ENDIF
                                ;.      LET 100=ADDRESS OF UNCONDITIONAL_INTERRUPT_ROUTINE
                                ;.      DO FOR BCSR<12> FROM #0 TO #1(LKS DISABLED AND ENABLED)
                                ;.      (IF UFD DO ONLY FOR SELECTED LINE CLOCK)
                                ;.      .       CLEAR UNCONDITIONAL_INTERRUPT
                                ;.      .       SET COUNTER TO WAIT  FOR 3 INTERRUPTS
                                ;.      .       LET BCSR<13>=1
                                ;.      .       REPEAT
                                ;.      .       .       DECREMENT COUNTER
                                ;.      .       UNTIL UNCONDITIONAL_INTERUPT EQ #3 OR COUNTER EQ #0
                                ;.      .       IF COUNTER GT TIME_REQUIRED_FOR_3_INTERRUPTS_FOR_800HZ
                                ;.      .       .       ERROR (INTERRUPTS NEVER GO LOW)
                                ;.      .       ENDIF
                                ;.      .       IF UNCONDITIONAL_INTERRUPT LT #3 THEN
                                ;.      .       .       ERROR (INTERRUPTS DON'T HAPPEN)
                                ;.      .       ENDIF
                                ;.      LET BCSR<13>=#0
                                ;.      ENDDO
                                ;ENDROUTINE
                                ;
                                ;ROUTINE UNCONDITIONAL_INTERRUPT_ROUTINE
                                ;.      INCREMENT UNCONDITIONAL_INTERRUPT
                                ;RETURN

                                ;;*****************************************************************
    117632  000004              TST40:  SCOPE
19521 117634  032737  000100  000052          BIT     #BIT06,@#52                     ;UFD MODE?
19522 117642  001404                          BEQ     1$                              ;IF NOT, BRANCH
19523 117644  032737  020000  177520          BIT     #BIT13,BCSR                     ;FORCE INTERRUPT SET?
19524 117652  001530                          BEQ     TST41           ;;IF SET, EXIT TEST
19525                           ;
19526                           ; CHECK BSCR<13> TO BE 0 AND 1
19527                           ;
19528 117654  013737  177520  002730  1$:     MOV     BCSR,SAVBR                      ;SAVE BCSR REGISTER
19529 117662  042737  001000  177520          BIC     #1000,BCSR                      ;DISABLE HALT ON BREAK
19530 117670  042737  020000  177520          BIC     #BIT13,BCSR                     ;CLEAR BCSR
19531 117676  032737  020000  177520          BIT     #BIT13,BCSR                     ;<13>=0?
19532 117704  001403                          BEQ     2$                              ;IF OK, BRANCH
19533 117706  005037  001124                  CLR     $GDDAT                          ;CLEAR EXPECTED PATTERN
19534 117712  104052                          ERROR   +52                             ;ERROR BCSR READ/WRITE BITS
19535 117714  052737  020000  177520  2$:     BIS     #BIT13,BCSR                     ;SET BIT 13
19536 117722  032737  020000  177520          BIT     #BIT13,BCSR                     ;GOT SET OK?
19537 117730  001004                          BNE     3$                              ;IF OK, BRANCH
19538 117732  012737  020000  001124          MOV     #BIT13,$GDDAT                   ;EXPECTED PATTERN
```

N12

TEST - UNCONDITIONAL CLOCK LINE INTERRUPTS

```
 19539 117740  104052                          ERROR   +52                     ;ERROR BCSR READ/WRITE BITS
 19540                                  ;
 19541                                  ; SET UP  TO DO UNCONDITIONAL INTERRUPTS
 19542                                  ;
 19543 117742  012737  137062  000100  3$:     MOV     #LKSINT,@#100           ;SET UP INTERRUPT VECTOR
 19544 117750  012737  000340  000102          MOV     #340,@#102              ;AT PRIORITY 7
 19545 117756  052737  010000  177520          BIS     #BIT12,BCSR             ;FOR 1ST TIME DISABLE LKS
 19546 117764  000403                          BR      5$                      ;GO DO IT
 19547 117766  042737  010000  177520  4$:     BIC     #BIT12,BCSR             ;FOR THE 2ND TIME, ENABLE LKS
 19548 117774  005037  002722          5$:     CLR     LKSFL                   ;CLEAR INTERRUPTS FLAG
 19549 120000  012702  077777                  MOV     #77777,R2               ;COUNTER TO WAIT FOR INTERRUPTS
 19550 120004  106427  000240                  MTPS    #240                    ;LOWER PRIORITY TO 5
 19551 120010  022737  000003  002722  6$:     CMP     #3,LKSFL                ;3 INTERRUPTS HAPPENED?
 19552 120016  001401                          BEQ     7$                      ;EXIT LOOP, IF SO
 19553 120020  077205                          SOB     R2,6$                   ;OTHERWISE, KEEP WAITING
 19554 120022  106427  000340          7$:     MTPS    #340                    ;RAISE PRIORITY TO 7
 19555 120026  022702  077700                  CMP     #77700,R2               ;INTERRUPTS HAPPEN TOO OFTEN?
 19556 120032  002001                          BGE     8$                      ;IF NOT, BRANCH
 19557 120034  104063                          ERROR   +63                     ;READY LINE DOESN'T GO LOW
 19558 120036  022737  000003  002722  8$:     CMP     #3,LKSFL                ;AT LEAST 3 INTERRUPTS HAPPENED?
 19559 120044  002004                          BGE     9$                      ;IF SO, BRANCH
 19560 120046  012737  000003  001124          MOV     #3,$GDDAT               ;EXPECTED DATA
 19561 120054  104064                          ERROR   +64                     ;INTERRUPTS DON'T HAPPEN
 19562 120056  032737  010000  177520  9$:     BIT     #BIT12,BCSR             ;SECOND TIME THRU THE LOOP?
 19563 120064  001340                          BNE     4$                      ;IF NOT, DO IT AGAIN
 19564 120066  032737  000100  000052          BIT     #BIT06,@#52             ;UFD MODE?
 19565 120074  001404                          BEQ     10$                     ;IF NOT, BRANCH
 19566 120076  032737  010000  177520          BIT     #BIT12,BCSR             ;IF UFD AND LKS DISABLED?
 19567 120104  001010                          BNE     12$                     ;DON'T CHECK LKS
 19568 120106  032737  000100  177546  10$:    BIT     #BIT06,LKS              ;INTERRUPT ENABLE LINE HOLD 1?
 19569 120114  001001                          BNE     11$                     ;IF SO, BRANCH
 19570 120116  104067                          ERROR   +67                     ;BCSR<13> DOESN'T SET LKS<6>
 19571 120120  042737  000100  177546  11$:    BIC     #BIT06,LKS              ;DISABLE LKS INTERRUPTS
 19572 120126  013737  002730  177520  12$:    MOV     SAVBR,BCSR              ;RESTORE BCSR
 19573
 19574
```

```
                                .SBTTL  TEST - RESETTING LKS
                                ;RESETTING LKS(*)
                                ;THIS TEST WILL PROVE THAT RESET INSTRUCTION SETS LKS<07> AND
                                ;CLEARS LKS<06>.
                                ;ROUTINE TEST
                                ;IF UFD AND LKS IS DISABLED THEN
                                ;.      EXIT TEST
                                ;ENDIF
                                ;.      POINT LKS VECTOR 100 TO ERROR_LKS_ILLEGAL_INTERRUPT
                                ;.      SYNCHRONIZE LKS BY WAITING FOR 3 PULSES
                                ;.      LET LKS<06>=#1
                                ;.      CLEAR LKS (CLEARS LKS<07>
                                ;.      EXECUTE "RESET"
                                ;.      IF LKS<7> NE #1 OR LKS<6> NE #0 THEN
                                ;.      .       ERROR
                                ;.      ENDIF
                                ;.      IF ILLEGAL_LINE_CLOCK_INTERRUPT NE 0 THEN
                                ;.      .       ERROR
                                ;.      ENDIF
                                ;ENDROUTINE
                                ;
                                ;ROUTINE ERROR_LKS_ILLEGAL_INTERRUPT
                                ;.      FLAG ILLEGAL_LINE_CLOCK_INTERRUPT
                                ;RETURN

                                ;;*****************************************************************
 120134  000004                 TST41:  SCOPE
19602 120136  005737  001206            TST     $PASS                   ;FIRST PASS?
19603 120142  001057            BNE     TST42           ;;IF NOT FIRST PASS, EXIT TEST
19604 120144  032737  000100  000052    BIT     #BIT06,@#52             ;UFD MODE?
19605 120152  001404            BEQ     1$              ;IF NOT, BRANCH
19606 120154  032737  010000  177520    BIT     #BIT12,BCSR             ;LKS DISABLED?
19607 120162  001447            BEQ     TST42           ;;IF DISABLED, EXIT TEST
19608                           ;
19609                           ; SYNCHRONISE WITH LINE TIME CLOCK BY WAITING FOR 3 INTERRUPTS
19610                           ;
19611 120164  013737  177520  002730  1$:     MOV     BCSR,SAVBR              ;SAVE BCSR
19612 120172  042737  010000  177520        BIC     #BIT12,BCSR             ;ENABLE LKS RESPONSE
19613 120200  012737  137062  000100        MOV     #LKSINT,@#100           ;SET UP INTERRUPT VECTOR
19614 120206  012737  000340  000102        MOV     #340,@#102              ;AT PROIRITY 7
19615 120214  052737  000100  177546        BIS     #BIT06,LKS              ;SET INTERRUPT ENABLE BIT
19616 120222  005037  002722            CLR     LKSFL                   ;CLEAR INTERRUPTS FLAG
19617 120226  012702  077777            MOV     #77777,R2               ;COUNTER TO WAIT FOR INTERRUPTS
19618 120232  106427  000240            MTPS    #240                    ;LOWER PRIORITY TO 5
19619 120236  022737  000003  002722  2$:     CMP     #3,LKSFL                ;3 INTERRUPTS HAPPENED?
19620 120244  001401            BEQ     3$                      ;EXIT LOOP, IF SO
19621 120246  077205            SOB     R2,2$                   ;OTHERWISE, KEEP WAITING
19622 120250  106427  000340    3$:     MTPS    #340                    ;RAISE PRIORITY TO 7
19623 120254  000005            RESET                           ;EXECUTE RESET
19624 120256  032737  000200  177546        BIT     #BIT07,LKS              ;READY BIT SET?
19625 120264  001001            BNE     4$                      ;IF SO, BRANCH
19626 120266  104070            ERROR   .70                     ;RESET DOESN'T SET LKS<07.
19627 120270  032737  000100  177546  4$:     BIT     #BIT06,LKS              ;INTERRUPT ENABLE BIT CLEARED?
19628 120276  001401            BEQ     TST42           ;;IF SO, EXIT TEST
19629 120300  104071            ERROR   .71                     ;RESET DOESN'T CLEAR LKS
19630
19631
```

```
                                .SBTTL  TEST  - LINE CLOCK INTERRUPTS
                                ;LINE CLOCK INTERRUPTS(*)
                                ;BY SETTING TO 1 LKS<06>, THIS TEST WILL CHECK FOR INTERRUPTS
                                ;FROM BEVENT LINE AND FROM KDJ11-B 50H?  60HZ, 800HZ ON BOARD SIGNALS
                                ;(THE LATTER SIGNALS WILL BE ACCESSED BY SETTING BCSR<11-10>).
                                ;
                                ;BCSR   <11>    <10>            CLOCK SELECT BITS 1 AND 0
                                ;
                                ;        0       0              EXTERNAL BEVENT LINE
                                ;        0       1              ON BOARD 50 HZ
                                ;        1       0              ON BOARD 60 HZ
                                ;        1       1              ON-BOARD 800 HZ
                                ;
                                ;ROUTINE TEST
                                ;IF UFD THEN
                                ;.      IF LKS DISABLED THEN
                                ;.      .       EXIT TEST
                                ;.      ENDIF
                                ;.      SET FLAGS TO RUN ONLY WHAT SPECIFIED IN EAPROM
                                ;ENDIF
                                ;.      LET 100=ADDRESS OF LKS_INTERRUPT
                                ;.      DO FOR BCSR<11;10> FROM #0 TO #3
                                ;.      .       LET LKS<06>=#1
                                ;.      .       WAIT FOR 10 INTERRUPTS FOR EACH CLOCK
                                ;.      .       STORE ACTUAL NUMBER OF INTERRUPTS FOR EACH CLOCK
                                ;.      .       LET INTERRUPT_FLAG=0
                                ;.      ENDDO
                                ;.      COMPARE NUMBER OF INTERRUPTS FOR EACH CLOCK
                                ;ENDROUTINE
                                ;
                                ;ROUTINE LKS_INTERRUPT
                                ;.      INCREMENT INTERRUPT_FLAG
                                ;RETURN

                                ;;*****************************************************************
                                TST42:  SCOPE
                                        NOP                             ; THIS CODE ADDED FOR APT DEFAULT BREAK
                                !                                       ; PURPOSES-- TEST TIME LONGER THAN SOME APT BREAK INT
120306  005737  003032                  TST     CCHPAS                  ;have done enough inclusive passes?
120312  001003                          BNE     99$                     ; not yet
120314  000240                          NOP                             ; debug aid
120316  000137  120564                  JMP     10$                     ; yes skip this
                                99$:    NOP
120322  032737  000100  000052          BIT     #BIT06,@#52                     ;UFD MODE?
120332  001411                          BEQ     1$                      ;IF NOT, GO DO THE TEST
120334  032737  010000  177520          BIT     #BIT12,BCSR             ;LKS IS DISABLED?
120342  001123                          BNE     TST43           ;;IF DISABLED, EXIT TESTS
120344  005002                          CLR     R2                      ;CLEAR R2 TO SET FLAGS
120346  053702  177520                  BIS     BCSR,R2                 ;SET R2 ACCORDING TO BCSR
120352  042702  171777                  BIC     #171777,R2              ;LEAVE ONLY BITS <11 10>
                                ;
                                ; SETUP DELAY VALUES FOR INTERRUPTS, IN UFD MODE ONLY FROM THE CLOCK SPECIFIED IN BCSR<11-10>
                                ;
120356  013737  177520  002730  1$:     MOV     BCSR,SAVBR              ;STORE BCSR
120364  012737  137062  000100          MOV     #LKSINT,100             ;SET UP LKS VECTOR
120372  012737  000340  000102          MOV     #340,102                ;AT PRIORITY 7
120400  012705  120602                  MOV     #TIMDEL,R5              ;POINTER TO DEL
```

```
TEST  - LINE CLOCK INTERRUPTS

 19689 120404  012704  000004                    MOV     #4,R4                   ;R4 IS THE COUNTER FOR ALL CLOCKS
 19690 120410  005037  177520                    CLR     BCSR                    ;DO FOR BEVENT LINE INTERRUPTS
 19691 120414  000403                            BR      3$                      ;GO DO THE LOOP
 19692 120416  062737  002000  177520  2$:       ADD     #2000,BCSR              ;SET UP FOR THE NEXT CLOCK LINE
 19693 120424  032737  000100  000052  3$:       BIT     #BIT06,@#52             ;UFD MODE?
 19694 120432  001402                            BEQ     4$                      ;IF NOT, BRANCH
 19695 120434  010237  177520                    MOV     R2,BCSR                 ;IN UFD, DO ONLY FOR SPECIFIED
 19696 120440  005037  002722            4$:     CLR     LKSFL                   ;CLEAR INTERRUPT FLAG
 19697 120444  052737  000100  177546            BIS     #BIT06,LKS              ;SET INTERRUPT ENABLE BIT
 19698 120452  012703  000010                    MOV     #10,R3                  ;START COUNTER FOR 10 INTERURRUPTS
 19699 120456  012701  177777            5$:     MOV     #177777,R1              ;START COUNTER TO WAIT FOR INTERRUPT
 19700 120462  106427  000240                    MTPS    #240                    ;LOWER PRIORITY TO 5
 19701 120466  023703  002722            6$:     CMP     LKSFL,R3                ;NEW INTERRUPT HAPPENED?
 19702 120472  001401                            BEQ     7$                      ;IF SO, EXIT WAIT LOOP
 19703 120474  077104                            SOB     R1,6$                   ;OTHERWISE, KEEP WAITING
 19704 120476  010125            7$:     MOV     R1,(R5)+                ;STORE DELAY FOR EACH CLOCK
 19705 120500  032737  000100  000052            BIT     #BIT06,@#52             ;UFD MODE?
 19706 120506  001026                            BNE     10$                     ;IF UFD, DON'T DO FOR ANY OTHER
 19707 120510  077436                            SOB     R4,2$                   ;ALL LINE CLOCKS DONE?
 19708                                 ;
 19709                                 ; CHECK THE DELAY VALUES FOR ALL CLOCKS
 19710                                 ;
 19711 120512  106427  000340                    MTPS    #340                    ;RAISE PRIORITY
 19712 120516  042737  000100  177546            BIC     #BIT06,LKS              ;DISABLE INTERRUPTS
 19713 120524  012705  120602                    MOV     #TIMDEL,R5              ;POINTER TO DELAY TABLE
 19714 120530  021565  000006                    CMP     (R5),6(R5)              ;DELAY FOR BEVENT AND 800HZ?
 19715 120534  103401                            BLO     8$                      ;BEVENT IS NOT 800HZ
 19716 120536  104064                            ERROR   .64                     ;WRONG # OF INTERRUPTS
 19717 120540  005725            8$:     TST     (R5)+                   ;INCREMENT POINTER
 19718 120542  021565  000002                    CMP     (R5),2(R5)              ;DELAY FOR 50HZ AND 60HZ?
 19719 120546  103401                            BLO     9$                      ;IF FIRST BIGGER, BRANCH
 19720 120550  104064                            ERROR   .64                     ;WRONG # OF INTERRUPTS
 19721 120552  005725            9$:     TST     (R5)+                   ;INCREMENT POINTER
 19722 120554  021565  000002                    CMP     (R5),2(R5)              ;DELAY FOR 50HZ AND 800HZ
 19723 120560  103401                            BLO     10$                     ;IF FIRST BIGGER, BRANCH
 19724 120562  104064                            ERROR   .64                     ;WRONG # OF INTERRUPTS
 19725 120564  013737  002730  177520  10$:     MOV     SAVBR,BCSR              ;RESTORE BCSR
 19726 120572  042737  000100  177546            BIC     #BIT06,LKS              ;DISABLE INTERRUPTS
 19727 120600  000404                            BR      TST43           ;;EXIT TEST
 19728
 19729 120602                          TIMDEL: .BLKW 4
 19730
```

E13

TEST - MAINTENANCE REGISTER TEST

```
19732                                   .SBTTL  TEST    MAINTENANCE REGISTER TEST
19733                                   ;MAINTENANCE REGISTER TEST
19734                                   ;THIS TEST WILL ADDRESS MAINTENANCE REGISTER AND CHECK BITS
19735                                   ;7-4 TO BE 0010, 2 1 TO BE 10, AND READ BITS 10 08, 03, 00
19736                                   ;FOR FUTURE USE.  THOSE BITS REPRESENT THE FOLLOWING SIGNALS:
19737                                   ;MULTIPROCESSOR SLAVE, UNIBUS SYSTEM, FPA AVAILABLE, HALT/TRAP
19738                                   ;OPTION, AND AC POWER OKAY.
19739                                   ;ROUTINE TEST
19740                                   ;.      IF MAINT. REG. BITS <7 4> NE 0010 OR <2-1> NE 10 THEN
19741                                   ;.          .       ERROR
19742                                   ;.      ENDIF
19743                                   ;.      READ MAINT.REG. BITS <10 08,03,00>
19744                                   ;ENDROUTINE
19745
19746                                   ;;*****************************************************************
    120612  000004                      TST43:  SCOPE
19747 120614  032737  174000  177750          BIT     #174000,MAIREG                  ;UNUSED BITS ALL ZEROS?
19748 120622  001401                          BEQ     1$                              ;IF OK, BRANCH
19749 120624  104132                          ERROR   +132                            ;MAINTENANCE REGISTER ERROR
19750 120626  032737  000044  177750  1$:     BIT     #44,MAIREG                      ;<5,2> SET ?
19751 120634  001001                          BNE     2$                              ;IF SO, BRANCH
19752 120636  104132                          ERROR   +132                            ;MAINTENANCE REGISTER ERROR
19753 120640  032737  000322  177750  2$:     BIT     #322,MAIREG                     ;<7,6,4,1> CLEAR?
19754 120646  001401                          BEQ     TST44           ;;IF YES, BRANCH
19755 120650  104132                          ERROR   +132
19756
19757
```

```
19759                                   .SBTTL  TEST    SERIAL LINE UNIT REGISTERS
19760                                   ;SERIAL LINE UNIT TEST(*)
19761                                   ;BCR<2 0> WILL BE READ TO FIND OUT BAUD RATE. SLU WILL BE PROG
19762                                   ;RAMMED TO CHECK THE INTERRUPT LEVELS BY SETTING BIT<06> IN
19763                                   ;RCSR AND XMIT. LOOP BACK CAPABILITIES WILL BE TESTED BY SETTING
19764                                   ;TO 1 XCSR<02>.   THE LINE CLOCK INTERRUPT SUBROUTINE WILL BE
19765                                   ;USED TO RETURN TO THE EXECUTION OF THE DIAGNOSTICS. IF THE
19766                                   ;PROGRAM HANGS IN THE LOOP BACK MODE.
19767                                   ;ROUTINE TEST
19768                                   ;IF UFD AND CONSOLE NOT PRESENT
19769                                   ;.      GO TO TEST 22
19770                                   ;ENDIF
19771                                   ;.      IF BCR<07> EQ #0 THEN
19772                                   ;.      .       READ BCR<2 0> TO GET BAUD RATE
19773                                   ;.      ENDIF
19774                                   ;.      LET 4=ADDRESS_OF_TIMEOUT_ROUTINE
19775                                   ;.      DO FOR RCSR,XCSR,RBUF,XBUF
19776                                   ;.      .       READ XRCSR,XCSR,RBUF,XBUF
19777                                   ;.      .       IF TIMEOUT_FLAG NE #0 THEN
19778                                   ;.      .       .       ERROR
19779                                   ;.      .       ENDIF
19780                                   ;.      ENDDO
19781                                   ;ENDROUTINE
19782                                   ;
19783                                   ;ROUTINE TIMEOUT
19784                                   ;.      LET TIMEOUT_FLAG=#1
19785                                   ;ENDROUTINE
19786
19787                                   ;;*****************************************************************
            120652  000004            TST44:  SCOPE
19788 120654  032737  000100  000052          BIT     #BIT06,@#52             ;UFD MODE?
19789 120662  001406                        BEQ     1$                      ;IF NOT, GO DO THE TEST
19790 120664  032737  000200  177524        BIT     #BIT07,BCR              ;IF UFD AND CONSOLE NOT PRESENT
19791 120672  001402                        BEQ     1$                      ;NOT TRUE, DO THE TEST
19792 120674  000137  122620              JMP     SLEND                   ;IF TRUE, SKIP ALL SLU TESTS
19793                                   ;
19794                                   ; TRY TO ACCESS SLU REGISTERS
19795                                   ;
19796 120700  013701  000004          1$:     MOV     ERRVEC,R1               ;SAVE TIMEOUT VECTOR
19797 120704  012737  120730  000004        MOV     #3$,ERRVEC              ;POINT NEW ONE TO PROGRAM AREA
19798 120712  012737  000340  000006        MOV     #340,ERRVEC+2           ;AT PRIORITY 7
19799 120720  012702  177560              MOV     #RCSR,R2                ;START ACCESSING WITH RCSR
19800 120724  005712                  2$:     TST     (R2)                    ;ACCESS SLU REGISTER
19801 120726  000403                        BR      4$                      ;IF NO TIMEOUT, CONTINUE
19802 120730  010237  001126          3$:     MOV     R2,$BDDAT               ;STORE ADDRESS THAT TIMED OUT
19803 120734  104072                        ERROR   +72                     ;TIMEOUT ACCESSING SLU REGISTER
19804 120736  022722  177566          4$:     CMP     #XBUF,(R2)+             ;LAST REGISTER ACCESSED?
19805 120742  103770                        BLO     2$                      ;IF NOT, BRANCH
19806 120744  010137  000004              MOV     R1,ERRVEC               ;RESTORE TIMEOUT VECTOR
19807
```

```
                                .SBTTL  TEST  - XCSR BIT 7
                                ;CHECK THAT XCSR<07> CAN BE 0 AND 1.
                                ;
                                ;XCSR   <07>            TRANSMITTER READY
                                ;
                                ;ROUTINE TEST
                                ;.      WAIT FOR XCSR<07>=#1 NO MORE THAN 200MSEC
                                ;.      IF XCSR<07> NE #1 THEN
                                ;.          ERROR
                                ;.      ENDIF
                                ;.      LET XBUF=#NULL
                                ;.      WAIT FOR XCSR<07>=#1
                                ;.      LET XBUF=#NULL
                                ;.      IF XCSR<07> NE 0 THEN
                                ;.          ERROR (READY DIDN'T GO LOW)
                                ;.      ENDIF
                                ;ENDROUTINE

                                ;;*****************************************************************
120750  000004                  TST45:  SCOPE
19828 120752  012701  001000            MOV     #1000,R1                        ;COUNTER FOR ABOUT 200MICROSEC.
19829 120756  122737  000001  001220      CMPB    #APTENV,$ENV                    ;RUNNING IN APT MODE?
19830 120764  001003                      BNE     1$                              ;NO, GO DO TEST
19831 120766  005737  001206              TST     $PASS                           ;FIRST PASS?
19832 120772  001017                      BNE     TST46           ;;IF APT AND NOT FIRST PASS, EXIT TEST
19833 120774  105737  177564      1$:     TSTB    XCSR                            ;XCSR<7> READY 1?
19834 121000  100401                      BMI     2$                              ;IF SO, EXIT WAIT LOOP
19835 121002  077104                      SOB     R1,1$                           ;IF NOT 1, CONTINUE WAITING
19836 121004  105737  177564      2$:     TSTB    XCSR                            ;XCSR<7>=1?
19837 121010  100401                      BMI     3$                              ;IF YES, BRANCH
19838 121012  104073                      ERROR   +73                             ;XCSR<7> DOES NOT BECOME 1
19839 121014  012737  000000  177566  3$: MOV     #NULL,XBUF                      ;TRY TO TRANSMIT NULL CHARACTER
19840 121022  105737  177564              TSTB    XCSR                            ;XCSR<7>=0
19841 121026  100001                      BPL     TST46           ;;IF YES, EXIT TEST
19842 121030  104073                      ERROR   +73                             ;XMIT READY DIDN'T GO LOW
19843
19844
```

TEST - RCSR BIT 7 AND XCSR BIT 2

```
                                .SBTTL  TEST    RCSR BIT 7 AND XCSR BIT 2
                                ;CHECK THAT RCSR<07> CAN BE 0 AND 1 AND THAT XCSR<02> WORKS PROPERLY
                                ;
                                ;RCSR   <07>    RECEIVER DONE
                                ;XCSR   <02>    MAINTENANCE
                                ;
                                ;ROUTINE TEST
                                ;.(CHECK RCSR<07> AND XCSR<07>)
                                ;.      WAIT FOR XCSR<07>=#1
                                ;.      LET XCSR<02>=#1 (LOOP BACK MODE)
                                ;.      LET XBUF=#125
                                ;.      WAIT FOR RCSR<07>=#1 NO MORE THAN 200MSEC
                                ;.      IF RCSR<07> NE #1 THEN
                                ;.      .       ERROR (RCSR<07> DOES NOT BECOME 1 OR XCSR<02>DOES NOT
                                ;.      .               WORK)
                                ;.      ENDIF
                                ;.      IF RBUF NE #125 THEN
                                ;.      .       ERROR
                                ;.      ENDIF
                                ;.      IF RCSR<07> NE #0 THEN
                                ;.      .       ERROR (RCSR<07>DOES NOT GO LOW)
                                ;.      ENDIF
                                ;.      LET XCSR<02>=#0
                                ;ENDROUITNE

                                ;;*****************************************************************
 121032  000004                 TST46:  SCOPE
                                        MOV     #13,R1                          ;COUNTER FOR ABOUT 200MICROSEC.
 121034  012701  000013                 CMPB    #APTENV,$ENV                    ;RUNNING IN APT MODE?
 121040  122737  000001  001220         BNE     1$                              ;NO, GO DO TEST
 121046  001003                         TST     $PASS                           ;FIRST PASS?
 121050  005737  001206                 BNE     TST47           ;;IF APT AND NOT FIRST PASS, EXIT TEST
 121054  001074                 1$:     TSTB    XCSR                            ;XCSR<7> READY 1?
 121056  105737  177564                 BPL     1$                              ;IF NOT 1, CONTINUE WAITING
 121062  100375                 2$:     BIS     #BIT02,XCSR                     ;SET LOOP BACK MODE
 121064  052737  000004  177564         BIT     #BIT02,XCSR                     ;GOT SET OK?
 121072  032737  000004  177564         BNE     3$                              ;IF YES, BRANCH
 121100  001004                         CLR     XCSR                            ;RESET TO PRINT ERROR
 121102  005037  177564                 ERROR   +114                            ;XCSR<2> DOES NOT BECOME 1
 121106  104114                         BR      TST47           ;;EXIT TEST
 121110  000456                 ;
                                ; STALL FOR A WHILE IN CASE XCSR<2> CAUSES RCSR<7> TO BE 1
                                ;
 121112  012701  060000         3$:     MOV     #60000,R1                       ;STALL IN CASE XCSR<2> SETS READY
 121116  105737  177560         4$:     TSTB    RCSR                            ;IF RECEIVER READY SET?
 121122  100402                         BMI     5$                              ;IF SET, BRANCH
 121124  077104                         SOB     R1,4$                           ;OTHERWISE, STAY FOR A WHILE
 121126  000402                         BR      6$                              ;IF NOT READY, BRANCH
 121130  005737  177562         5$:     TST     RBUF                            ;READ RBUF
                                ;
                                ; TRANSMIT XON AND CHECK RCSR<7>
                                ;
 121134  012737  000021  177566 6$:     MOV     #21,XBUF                        ;TRANSMIT A CHARACTER
 121142  012701  060000                 MOV     #60000,R1                       ;COUNTER TO WAIT
 121146  105737  177560         7$:     TSTB    RCSR                            ;RCSR<7> READY 1?
 121152  100401                         BMI     8$                              ;IF YES, EXIT WAIT LOOP
 121154  077104                         SOB     R1,7$                           ;OTHERWISE, CONTINUE WAITING
```

I13

TEST - RCSR BIT 7 AND XCSR BIT 2

```
19902 121156  105737  177560          8$:     TSTB    RCSR                    ;RCSR<7>=1?
19903 121162  100403                          BMI     9$                      ;IF YES, BRANCH
19904 121164  005037  177564                  CLR     XCSR                    ;RESET XCSR<2>
19905 121170  104074                          ERROR   .74                     ;RECEIVER READY DIDN T COME UP
19906 121172  013737  177562  001126  9$:     MOV     RBUF,$BDDAT             ;STORE RECEIVED DATA
19907 121200  022737  000021  001126          CMP     #21,$BDDAT              ;DATA RECEIVED OK?
19908 121206  001406                          BEQ     10$                     ;IF YES, BRANCH
19909 121210  012737  000021  001124          MOV     #21,$GDDAT
19910 121216  005037  177564                  CLR     XCSR                    ;RESET TO ENABLE SLU
19911 121222  104075                          ERROR   .75                     ;WRONG CHARACTER RECEIVED
19912 121224  105737  177560          10$:    TSTB    RCSR                    ;RCSR<7>=0?
19913 121230  100003                          BPL     11$                     ;IF ZERO, BRANCH
19914 121232  005037  177564                  CLR     XCSR                    ;RESET TO ENABLE SLU
19915 121236  104076                          ERROR   .76                     ;RCSR<07><>0 AFTER READING RBUF
19916 121240  042737  000004  177564  11$:    BIC     #BIT02,XCSR             ;DISABLE LOOP BACK MODE
19917
19918
```

J13

TEST RESET AND XCSR<2!0>

```
19920                                   .SBTTL  TEST    RESET AND XCSR<2!0>
19921                                   ;CHECK THAT RESET CLEARS XCSR<0!2>.
19922                                   ;ROUTINE TEST
19923                                   ;.(CHECK RCSR<07> AND XCSR<07> AND RESET)
19924                                   ;.      LET XCSR<02,00>=#1 (LOOP BACK MODE)
19925                                   ;.      EXECUTE "RESET"
19926                                   ;.      IF XCSR<02!00> NE #0 THEN
19927                                   ;.          .       ERROR
19928                                   ;.      ENDIF
19929                                   ;.      LET XCSR<02>=#0
19930                                   ;ENDROUTINE
19931
19932                                   ;;*****************************************************************
     121246  000004                     TST47:  SCOPE
19933 121250  005737  001206                     TST     $PASS                   ;FIRST PASS?
19934 121254  001011                     BNE     TST50           ;;IF NOT FIRST PASS, EXIT TEST
19935 121256  052737  000005  177564  1$:      BIS     #BIT02!BIT00,XCSR       ;LOOP BACK MODE
19936                                   ;
19937                                   ; EXECUTE RESET AND VALIDATE THAT XCSR<7,2> BECOMES <1,0>
19938                                   ;
19939 121264  000005                     RESET                                   ;EXECUTE RESET
19940 121266  032737  000005  177564     BIT     #BIT02!BIT00,XCSR       ;XCSR<2,0> CLEAR?
19941 121274  001401                     BEQ     TST50           ;;IF YES, BRANCH
19942 121276  104102                     ERROR   .102                    ;XCSR<2,0> NOT CLEARED ON RESET
19943
19944
```

K13


```
19946                                   .SBTTL  TEST    RESET AND INTERRUPT ENABLE BITS
19947                                   ;CHECK THAT INTERRUPTS DON'T HAPPEN AT PRIORITY 4 AND THAT RESET
19948                                   ;CLEARS XCSR<06> AND RCSR<06>.
19949                                   ;
19950                                   ;RCSR   <06>    RECEIVER INTERRUPT ENABLE
19951                                   ;XCSR   <06>    TRANSMITTER INTERRUPT ENABLE
19952                                   ;
19953                                   ;ROUTINE TEST
19954                                   ;.      LET 60=#ADDRESS_OF ILLEGAL_INTERRUPT_XRCSR
19955                                   ;.      LET 64=#ADDRESS_OF ILLEGAL_INTERRUPT XRCSR
19956                                   ;.      SET PRIORITY TO 4
19957                                   ;.      LET XCSR<02>=#1 (LOOPBACK MODE)
19958                                   ;.      LET XCSR<06>=#1 (ENABLE TRANSMIT INTERRUPTS)
19959                                   ;.      LET RCSR<06>=#1 (ENABLE RECEIVE INTERRUPTS)
19960                                   ;.      WAIT FOR XCSR<07>=#1 (READY TO TRANSMIT)
19961                                   ;.      LET XBUF=#NULL (SEND A CHARACTER)
19962                                   ;.      WAIT FOR ILLEGAL INTERRUPTS (ABOUT 200MSEC)
19963                                   ;.      EXECUTE "RESET"
19964                                   ;.      IF XCSR<06> NE #0 OR RCSR<06> NE #0 OR XRCSR NE #0 THEN
19965                                   ;.              ERROR
19966                                   ;.      ENDIF
19967                                   ;.      RESTORE PRIORITY TO NORMAL
19968                                   ;ENDROUTINE
19969                                   ;
19970                                   ;ROUTINE ILLEGAL_INTERRUPT_XRCSR
19971                                   ;.      INCREMENT XRCSR
19972                                   ;ENDROUTINE
19973
19974                                   ;;*****************************************************************
     121300  000004                     TST50:  SCOPE
19975 121302  005737  001206                     TST     $PASS                   ;FIRST PASS?
19976 121306  001033                     BNE     5$                      ;SKIP RESET PART OF THE TEST
19977                                   ;
19978                                   ; CHECK THAT INTERRUPTS ENABLE BITS FOR RECEIVER AND TRASMITTER OF SLU
19979                                   ; ARE CLEARED BY RESET
19980                                   ;
19981 121310  052737  000100  177564   1$:     BIS     #BIT06,XCSR             ;SET INTERRUPT ENABLE BIT IN XCSR
19982 121316  032737  000100  177564           BIT     #BIT06,XCSR             ;GOT SET OK?
19983 121324  001001                     BNE     2$                      ;IF YES, BRANCH
19984 121326  104110                     ERROR   .110                    ;IN BIT 6 OF XCSR
19985 121330  052737  000100  177560   2$:     BIS     #BIT06,RCSR             ;SET INTERRUPT ENABLE BIT IN RCSR
19986 121336  032737  000100  177560           BIT     #BIT06,RCSR             ;GOT SET OK?
19987 121344  001001                     BNE     3$                      ;IF YES, BRANCH
19988 121346  104110                     ERROR   .110                    ;IN BIT 6 OF RSCR
19989 121350  000005                     3$:     RESET                   ;INLINE BUS RESET
19990 121352  032737  000100  177564           BIT     #BIT06,XCSP             ;XMIT INTERRUPT ENABLE BIT CLEARED?
19991 121360  001401                     BEQ     4$                      ;IF CLEARED, BRANCH
19992 121362  104102                     ERROR   .102                    ;INTERRUPT ENABLE NOT CLEARED ON RESET
19993 121364  032737  000100  177560   4$:     BIT     #BIT06,RCSR             ;RECEIVE INTERRUPT ENBLE CLEARED?
19994 121372  001401                     BEQ     5$                      ;IF CLEARED, BRANCH
19995 121374  104102                     ERROR   .102                    ;INTERRUPT ENABLE NOT CLEARED ON RESET
19996                                   ;
19997                                   ; CHECK THAT TRANSMIT INTERRUPTS DON'T HAPPEN AT PRIORITY HIGHER THAN 3
19998                                   ;
19999 121376  012737  121444  000064   5$:     MOV     #9$,@#64                ;POINT XMIT VECTOR TO PROGRAM AREA
20000 121404  012737  000340  000066           MOV     #340,@#66               ;AT PRIORITY 7
20001 121412  052737  000100  177564           BIS     #BIT06,XCSR             ;SET INTERRUPT ENABLE BIT IN XCSR
```

TEST - RESET AND INTERRUPT ENABLE BITS

```
20002 121420  012702  000340                    MOV     #340,R2                 ;SET PRIORITY TO 7
20003 121424  000402                            BR      7$                      ;GO WAIT IN CASE OF INTERRUPTS
20004 121426  162702  000040            6$:     SUB     #40,R2                  ;LOWER PRIORITY LEVEL
20005 121432  106402                    7$:     MTPS    R2                      ;SET PRIORITY
20006 121434  012703  000026                    MOV     #26,R3                  ;TIME DELAY
20007 121440  077301                    8$:     SOB     R3,8$                   ;WAIT FOR INTERRUPTS
20008 121442  000403                            BR      10$                     ;IF INTERRUPTS DIDN'T HAPPENED, BRANCH
20009 121444  104101                    9$:     ERROR   ^101                    ;INTERRUPTS HAPPEN AT WRONG PRIORITY
20010 121446  005726                            TST     (SP)+                   ;CLEAN UP THE STACK
20011 121450  005726                            TST     (SP)+
20012 121452  022702  000200            10$:    CMP     #200,R2                 ;AT PRIORITY 4?
20013 121456  001363                            BNE     6$                      ;IF NOT LAST ONE, CONTINUE
20014 121460  106427  000340                    MTPS    #340                    ;RESTORE PRIORITY 7
20015 121464  042737  000100  177564            BIC     #BIT06,XCSR             ;CLEAR INTERRUPT ENABLE BIT
20016                                   ;
20017                                   ; CHECK THAT RECEIVE INTERRUPTS DON'T HAPPEN AT PRIORITY HIGHER THAN 3
20018                                   ;
20019 121472  012737  121562  000060            MOV     #15$,@#60               ;POINT RECEIVE VECTOR TO PROGRAM AREA
20020 121500  012737  000340  000062            MOV     #340,@#62               ;AT PRIORITY 7
20021 121506  105737  177564            11$:    TSTB    XCSR                    ;TRANSMITTER READY
20022 121512  100375                            BPL     11$                     ;IF NOT, WAIT
20023 121514  012737  000000  177566            MOV     #NULL,XBUF              ;TRY TO TRANSMIT NULL
20024 121522  052737  000100  177560            BIS     #BIT06,RCSR             ;SET INTERRUPT ENABLE BIT IN RCSR
20025 121530  012702  000340                    MOV     #340,R2                 ;SET PRIORITY TO 7
20026 121534  000402                            BR      13$                     ;GO WAIT IN CASE OF INTERRUPTS
20027 121536  162702  000040            12$:    SUB     #40,R2                  ;LOWER PRIORITY LEVEL
20028 121542  052737  000004  177564    13$:    BIS     #BIT02,XCSR             ;SET LOOP BACK MODE
20029 121550  106402                            MTPS    R2                      ;SET PRIORITY
20030 121552  012703  000100                    MOV     #100,R3                 ;TIME DELAY
20031 121556  077301                    14$:    SOB     R3,14$                  ;WAIT FOR INTERRUPTS
20032 121560  000406                            BR      16$                     ;IF INTERRUPTS DIDN'T HAPPENED, BRANCH
20033 121562  042737  000004  177564    15$:    BIC     #BIT02,XCSR             ;CLEAR LOOP BACK MODE
20034 121570  104101                            ERROR   ^101                    ;INTERRUPTS HAPPEN AT WRONG PRIORITY
20035 121572  005726                            TST     (SP)+                   ;CLEAN UP THE STACK
20036 121574  005726                            TST     (SP)+
20037 121576  022702  000200            16$:    CMP     #200,R2                 ;AT PRIORITY 4?
20038 121602  001355                            BNE     12$                     ;IF NOT LAST ONE, CONTINUE
20039                                   ;
20040                                   ; CLEAN UP BEFORE NEXT TEST
20041                                   ;
20042 121604  106427  000340                    MTPS    #340                    ;RESTORE PRIORITY 7
20043 121610  042737  000100  177560            BIC     #BIT06,RCSR             ;CLEAR INTERRUPT ENABLE BIT
20044 121616  012702  000300                    MOV     #300,R2                 ;STALL DELAY
20045 121622  105737  177560            17$:    TSTB    RCSR                    ;RECEIVE READY?
20046 121626  100401                            BMI     18$                     ;STOP WAITING, IF SO
20047 121630  077204                            SOB     R2,17$                  ;OTHERWISE, STAY IN THE LOOP
20048 121632  005737  177562            18$:    TST     RBUF                    ;READ CHARACTER TRANSMITTED
20049 121636  042737  000004  177564            BIC     #BIT02,XCSR             ;CLEAR LOOP BACK MODE
20050
```

```
                                .SBTTL  TEST    INTERRUPT PRIORITY FOR SLU
                                ;CHECK THAT INTERRUPTS HAPPEN AT PRIORITY 3 AND THAT THEY CLEAR
                                ;RCSR<06> AND XCSR<06>.
                                ;
                                ;ROUTINE TEST
                                ;.      LET 60=@ADDRESS_OF_LEGAL_RINTERRUPT
                                ;.      LET 64=@ADDRESS OF_LEGAL_XINTERRUPT
                                ;.      LET XCSR<02>=@1
                                ;.      SET PRIORITY TO @3
                                ;.      WAIT FOR XINTERRUPT=@3
                                ;.      IF XCSR<07> EQ @1 THEN
                                ;.      .       ERROR
                                ;.      ENDIF
                                ;.      WAIT FOR RINTERRUPT=@3
                                ;.      IF RCSR<07> EQ @0 THEN
                                ;.      .       ERROR
                                ;.      ENDIF
                                ;.      LET XCSR<02>=@0
                                ;.      SET PRIORITY TO NORMAL
                                ;ENDROUTINE
                                ;
                                ;ROUTINE LEGAL_XINTERRUPT
                                ;.      LET XBUF=@CHARACTER
                                ;.      INCREMENT XINTERRUPT
                                ;ENDROUTINE
                                ;
                                ;ROUTINE LEGAL_RINTERRUPT
                                ;.      READ RCSR
                                ;.      INCREMENT RINTERRUPT
                                ;ENDROUTINE
                                ;
                                ;;*****************************************************************
    121644  000004              TST51:  SCOPE
20084 121646  122737  000001  001220          CMPB    @APTENV,$ENV                    ;RUNNING IN APT MODE?
20085 121654  001003                          BNE     100$                            ;NO, GO DO TEST
20086 121656  005737  001206                  TST     $PASS                           ;FIRST PASS?
20087 121662  001113                          BNE     TST52           ;;IF APT AND NOT FIRST PASS, EXIT TEST
                                ;
                                ; GET READY FOR INTERRUPTS
                                ;
20091 121664  012737  122060  000060  100$:   MOV     @8$,@#60                        ;STORE RECEIVER VECTOR
20092 121672  012737  122004  000064          MOV     @6$,@#64                        ;STORE TRANSMITTER VECTOR
20093 121700  012737  000340  000062          MOV     @340,@#62                       ;AT PRIORITY 7
20094 121706  012737  000340  000066          MOV     @340,@#66                       ;FOR RECEIVER AND TRANSMITTER
20095 121714  052737  000004  177564          BIS     @BIT02,XCSR                     ;SET LOOP BACK MODE
20096 121722  012701  000100                  MOV     @100,R1                         ;DELAY FOR UNEXPECTED CHARACTERS
20097 121726  105737  177560          1$:     TSTB    RCSR                            ;RECEIVER READY?
20098 121732  100401                          BMI     2$                              ;IF YES, BRANCH
20099 121734  077104                          SOB     R1,1$                           ;OTHERWISE, WAIT JUST IN CASE
20100 121736  005737  177562          2$:     TST     RBUF                            ;READ RECEIVER
                                ;
                                ; SET PRIORITIES AND XMIT INTERRUPTS
                                ;
20104 121742  012702  000140                  MOV     @140,R2                         ;START WITH PRIORITY 3
20105 121746  000402                          BR      4$                              ;TRY TO DO IT
20106 121750  162702  000040          3$:     SUB     @40,R2                          ;LOWER PRIORITY
20107 121754  106402          4$:     MTPS    R2                                      ;TRY TO DO AT LOWER PRIORITY
```

N13

TEST - INTERRUPT PRIORITY FOR SLU

```
20108 121756  052737  000100  177564          BIS     #BIT06,XCSR                     ;LOOP BACK & INTERRUPT ENABLE
20109 121764  012703  001000                  MOV     #1000,R3                        ;WAIT DELAY FOR INTERRUPTS
20110 121770  077301                  5$:     SOB     R3,5$                           ;WAIT FOR XMIT INTERRUPTS
20111 121772  042737  000004  177564          BIC     #BIT02,XCSR                     ;CLEAR LOOP BACK BIT
20112 122000  104107                          ERROR   +107                            ;NO XMIT INTERRUPTS
20113 122002  000443                          BR      TST52           ;;IF ERROR, EXIT TEST
20114                                 ;
20115                                 ; TRANSMITTER INTERRUPT HERE
20116                                 ;
20117 122004  005726                  6$:     TST     (SP)+                           ;CLEAN UP STACK
20118 122006  005726                          TST     (SP)+
20119 122010  042737  000100  177564          BIC     #BIT06,XCSR                     ;CLEAR INTERRUPT ENABLE
20120 122016  012737  000000  177566          MOV     #NULL,XBUF                      ;TRANSMIT NULL
20121 122024  052737  000100  177560          BIS     #BIT06,RCSR                     ;SET RECEIVE INTERRUPT
20122 122032  106402                          MTPS    R2                              ;SET NEXT PRIORITY
20123 122034  012703  100000                  MOV     #100000,R3                      ;WAIT DELAY FOR INTERRUPTS
20124 122040  077301                  7$:     SOB     R3,7$                           ;WAIT FOR RECEIVE INTERUPTS
20125 122042  042737  000004  177564          BIC     #BIT02,XCSR                     ;CLEAR LOOP BACK MODE BIT
20126 122050  104107                          ERROR   +107                            ;NO RECEIVE INTERRUPTS
20127 122052  000406                          BR      9$                              ;DON'T TOUCH STACK
20128 122054  106427  000340                  MTPS    #340                            ;RAISE PRIORITY
20129                                 ;
20130                                 ; RECEIVER INTERRUPT HERE
20131                                 ;
20132 122060  005726                  8$:     TST     (SP)+                           ;CLEAN UP STACK
20133 122062  005726                          TST     (SP)+
20134 122064  005737  177562                  TST     RBUF                            ;READ RECEIVER BUFFER
20135 122070  005702                  9$:     TST     R2                              ;PRIORITY 0
20136 122072  001326                          BNE     3$                              ;IF NOT YET, CONTINUE
20137 122074  106427  000340                  MTPS    #340                            ;RAISE PRIORITY TO 7
20138 122100  042737  000100  177560          BIC     #BIT06,RCSR                     ;CLEAR RECEIVE INTER. ENABLE
20139 122106  005037  177564                  CLR     XCSR                            ;CLEAR XCSR
20140
```

```
                                .SBTTL  TEST  - BREAK CONDITION
                                ;CHECK THAT SENDING BREAK CAUSES FRAMING ERROR.
                                ;
                                ;RCSR   <15>    ERROR
                                ;       <13>    FRAMING ERROR
                                ;       <11>    RECEIVED BREAK
                                ;
                                ;XCSR   <00>    TRANSMIT BREAK
                                ;
                                ;ROUTINE TEST
                                ;.      LET XCSR<02>=#1
                                ;.      LET XCSR<00>=#1
                                ;.      WAIT FOR RCSR<07>=#1
                                ;.      IF RBUF<15!13!11> NE #1 THEN
                                ;.      .       ERROR (ERROR, FRAMING ERROR, RECEIVE BREAK NE 1)
                                ;.      ENDIF
                                ;.      LET XCSR<00>=#0
                                ;.      IF XCSR<00> NE #0 THEN
                                ;.      .       ERROR (XCSR<00> DOES NOT GO LOW)
                                ;.      ENDIF
                                ;.      WAIT FOR XCSR<07>=#1
                                ;.      LET XBUF=#NULL (SEND NULL CHARACTER TO SEE ERROR CLEARED)
                                ;.      WAIT FOR RCSR<07>=#1
                                ;.      IF RBUF<15!13!11> NE #0 THEN
                                ;.      .       ERROR
                                ;.      ENDIF
                                ;.      LET XCSR<00>=#1
                                ;.      EXECUTE "RESET"
                                ;.      IF XCSR<00> NE #0 THEN
                                ;.      .       ERROR
                                ;.      ENDIF
                                ;.      LET XCSR<02>=#0
                                ;ENDROUTINE
                                ;;*****************************************************************
    122112  000004              TST52:  SCOPE
                                ;
                                ; DECIDE WHETHER TO RUN THIS TEST
                                ;
122114  032737  000200  000052        BIT     #BIT07,@#52                     ;UFD MODE?
122122  001127                      BNE     TST53           ;;IN UFD MODE, EXIT TEST
122124  005737  001206              TST     $PASS                           ;FIRST PASS?
122130  001124                      BNE     TST53           ;;IF APT AND NOT FIRST PASS, EXIT TEST
                                ;
                                ; SEND BREAK AND CHECK ERROR BITS IN RBUF
                                ;
122132  052737  000004  177564  1$:     BIS     #BIT02,XCSR                     ;TRANSMIT IN LOOP BACK
122140  013737  177520  002730        MOV     BCSR,SAVBR                      ;SAVE BCSR
122146  042737  001000  177520        BIC     #BIT09,BCSR                     ;DISABLE HALT ON BREAK
122154  052737  000001  177564        BIS     #BIT00,XCSR                     ;SET SEND BREAK BIT
122162  032737  000001  177564        BIT     #BIT00,XCSR                     ;GOT SET OK?
122170  001001                      BNE     2$                              ;IF YES, BRANCH
122172  104110                      ERROR   +110                            ;WRITING 1 TO XCSR<0>
122174  012701  000100          2$:     MOV     #100,R1                         ;STALL DELAY
122200  105737  177560          4$:     TSTB    RCSR                            ;RECEIVER READY?
122204  100401                      BMI     5$                              ;IF YES, BRANCH
122206  077104                      SOB     R1,4$                           ;WAIT JUST IN CASE OF A CHARACTER
```

C14

TEST - BREAK CONDITION

```
20198 122210  005737  177562          5$:     TST     RBUF                            ;READ A CHARACTER
20199 122214  052737  000001  177564          BIS     #BIT00,XCSR                     ;TRANSMIT BREAK
20200 122222  012701  001000                  MOV     #1000,R1                        ;ANOTHER DELAY TO GET BREAK
20201 122226  077101                  6$:     SOB     R1,6$                           ;WAIT A WHILE
20202 122230  105737  177560          7$:     TSTB    RCSR                            ;RECEIVER READY?
20203 122234  100375                          BPL     7$                              ;IF NOT, WAIT
20204 122236  013737  177562  001126          MOV     RBUF,$BDDAT                     ;STORE WHATEVER RECEIVED
20205 122244  022737  124000  001126          CMP     #BIT15!BIT13!BIT11,$BDDAT       ;ALL ERROR BITS SET?
20206 122252  001405                          BEQ     8$                              ;IF YES, BRANCH
20207 122254  042737  000004  177564          BIC     #BIT02,XCSR                     ;RESET TO ENABLE SLU
20208 122262  104105                          ERROR   +105                            ;BREAK DOES NOT CAUSE ERRORS
20209 122264  000446                          BR      TST53           ;;EXIT
20210 122266  042737  000001  177564  8$:     BIC     #BIT00,XCSR                     ;CLEAR TRANSMIT BREAK
20211 122274  032737  000001  177564          BIT     #BIT00,XCSR                     ;GOT CLEARED OK?
20212 122302  001405                          BEQ     9$                              ;IF YES, BRANCH
20213 122304  042737  000004  177564          BIC     #BIT02,XCSR                     ;RESET TO ENABLE SLU
20214 122312  104110                          ERROR   +110                            ;ERROR WRITING 0 TO XCSR<0>
20215 122314  000432                          BR      TST53           ;;EXIT
20216                                 ;
20217                                 ; CHECK THAT BREAK CONDITION IS CLEARED
20218                                 ;
20219 122316  013737  002730  177520  9$:     MOV     SAVBR,BCSR                      ;RESTORE BCSR
20220 122324  105737  177564          10$:    TSTB    XCSR                            ;XMIT READY?
20221 122330  100375                          BPL     10$                             ;IF NOT, WAIT
20222 122332  012737  000177  177566          MOV     #177,XBUF                       ;TRY TO TRANSMIT DELETE
20223 122340  105737  177560          11$:    TSTB    RCSR                            ;RECEIVER READY
20224 122344  100375                          BPL     11$                             ;IF NOT, WAIT
20225 122346  013737  177562  001126          MOV     RBUF,$BDDAT                     ;STORE RECEIVE BUFFER
20226 122354  032737  124000  001126          BIT     #BIT15!BIT13!BIT11,$BDDAT       ;ERRORS CLEARED?
20227 122362  001404                          BEQ     12$                             ;IF YES, BRANCH
20228 122364  042737  000004  177564          BIC     #BIT02,XCSR                     ;RESET TO ENABLE SLU
20229 122372  104106                          ERROR   +106                            ;BREAK NOT CLEARED ON NEXT CHARACTER
20230 122374  042737  000004  177564  12$:    BIC     #BIT02,XCSR                     ;CLEAR LOOP BACK MODE
20231
20232
```

TEST - OVERRUN CONDITION

```
                                .SBTTL  TEST  - OVERRUN CONDITION
                                ;CHECK OVERRUN CONDITION
                                ;
                                ;RCSR   <14>    OVERRUN ERROR
                                ;
                                ;ROUTINE TEST
                                ;.      LET XCSR<02>=#1 (LOOPBACK MODE)
                                ;.      WAIT FOR XCSR<07>=#1
                                ;.      LET XBUF=#252
                                ;.      WAIT FOR XCSR<07>=#1
                                ;.      LET XBUF=#125 (SEND THE 2ND W/O READING THE 1ST CHARACTER)
                                ;.      WAIT FOR RCSR<07>=#1
                                ;.      STALL FOR LOWEST BAUD RATE TO GET 2ND CHARACTER
                                ;.      IF LOW BYTE OF RBUF NE #125 THEN
                                ;.      .       ERROR (1ST CHARACTER WASN'T OVERRUN)
                                ;.      ENDIF
                                ;.      IF RBUF<15!14> NE #1 THEN
                                ;.      .       ERROR (NO OVERRUN BIT SET)
                                ;.      ENDIF
                                ;.      WAIT FOR XCSR<07>=#1
                                ;.      LET XBUF=#NULL
                                ;.      WAIT FOR RCSR<07>=#1
                                ;.      IF RBUF<15!14> NE #0 THEN
                                ;.      .       ERROR (WASN'T CLEARED ON THE NEXT CHARACTER RECEIVED)
                                ;.      ENDIF
                                ;.      LET XCSR<02>=#0
                                ;ENDROUTINE

                                ;;*****************************************************************
 122402  000004                TST53:  SCOPE
 122404  122737  000001  001220        CMPB    #APTENV,$ENV                    ;RUNNING IN APT MODE?
 122412  001003                        BNE     100$                            ;NO, GO DO TEST
 122414  005737  001206                TST     $PASS                           ;FIRST PASS?
 122420  001077                        BNE     TST54           ;;IF APT AND NOT FIRST PASS, EXIT TEST
 122422  052737  000004  177564  100$:  BIS     #BIT02,XCSR                     ;SET LOOP BACK MODE
 122430  105737  177564          1$:   TSTB    XCSR                            ;READY TO TRANSMIT?
 122434  100375                        BPL     1$                              ;IF NOT, WAIT
 122436  012737  000021  177566        MOV     #21,XBUF                        ;TRANSMIT A CHARACTER
 122444  105737  177560          2$:   TSTB    RCSR                            ;RECEIVE READY?
 122450  100375                        BPL     2$                              ;IF NOT, WAIT
 122452  105737  177564          3$:   TSTB    XCSR                            ;READY TO TRANSMIT?
 122456  100375                        BPL     3$                              ;IF NOT, WAIT
 122460  012737  000177  177566        MOV     #177,XBUF                       ;TRANSMIT THE 2ND CHARACTER
 122466  012703  175000                MOV     #175000,R3                      ;STALL FOR THE 2ND CHARACTER $$$
 122472  077301                4$:     SOB     R3,4$                           ;WAIT A WHILE
 122474  013737  177562  001126        MOV     RBUF,$BDDAT                     ;STORE RECEIVED DATA
 122502  012737  140177  001124        MOV     #140177,$GDDAT                  ;EXPETED PATTERN
 122510  122737  000177  001126        CMPB    #177,$BDDAT                     ;2ND CHARACTER RECEIVED?
 122516  001404                        BEQ     5$                              ;IF YES, BRANCH
 122520  042737  000004  177564        BIC     #BIT02,XCSR                     ;RESET TO ENABLE SLU
 122526  104111                        ERROR   +111                            ;2ND CHARACTER DIDN'T OVERRUN 1ST
 122530  122737  000300  001127  5$:   CMPB    #BIT7!BIT6,$BDDAT+1             ;OVERRUN ERROR BITS SET?
 122536  001404                        BEQ     6$                              ;IF YES, BRANCH
 122540  005037  177564                CLR     XCSR                            ;RESET TO ENABLE SLU
 122544  104112                        ERROR   +112                            ;OVERRUN DOES NOT SET ERRORS BITS
 122546  000424                        BR      TST54           ;;EXIT
                                ;
```

E14

TEST OVERRUN CONDITION

```
20290                                   ; SEND NEXT CHARACTER TO CLEAR OVERRUN CONDITIONS
20291                                   ;
20292 122550 105737 177564      6$:     TSTB    XCSR                    ;TRANSMITTER READY?
20293 122554 100375                     BPL     6$                      ;IF NOT, BRANCH AND WAIT
20294 122556 012737 000000 177566       MOV     #NULL,XBUF              ;TRANSMIT NULL CHARACTER
20295 122564 105737 177560      7$:     TSTB    RCSR                    ;RECEIVER READY?
20296 122570 100375                     BPL     7$                      ;IF NOT, BRANCH AND WAIT
20297 122572 032737 140000 177562       BIT     #BIT15!BIT14,RBUF       ;ANY ERRORS SET?
20298 122600 001404                     BEQ     8$                      ;IF NOT, BRANCH
20299 122602 042737 000004 177564       BIC     #BIT02,XCSR             ;RESET TO ENABLE SLU
20300 122610 104113                     ERROR   +113                    ;OVERRUN NOT CLEARED ON NEXT CHAR.
20301 122612 042737 000004 177564 8$:   BIC     #BIT02,XCSR             ;CLEAR LOOP BACK MODE BIT
20302
20303 122620                    SLEND:                                  ;LAST SLU TEST
```

F14

TEST LED'S ON

```
20305                                   .SBTTL  TEST  - LED'S ON
20306                                   ;LED'S ON
20307                                   ;THIS TEST WILL INITIALIZE BDR TO CONTAIN A ROTATING PATTERN
20308                                   ;DISPLAYED IN LED'S.
20309                                   ;
20310                                   ;ROUTINE TEST
20311                                   ;.      WHILE A KEY NOT RECEIVED FROM KEYBOARD DO
20312                                   ;.      .       STALL ALLOWING TIME TO SEE PATTERN
20313                                   ;.      .       ROTATE LEFT TO LIGHT UP NEXT LED'S
20314                                   ;.      ENDDO
20315                                   ;ENDROUTINE
20316
20317                                   ;;*****************************************************************
     122620  000004                     TST54:  SCOPE
20318 122622  052737  001000  177520            BIS     #1000,BCSR              ;ENABLE HOB FOR APT
20319 122630  005005                            CLR     R5                      ;FLAG IN NO INTERRRUPT MODE
20320 122632  032737  000001  000052            BIT     #BIT00,@#52             ;IF RUNNING IN CHAIN MODE
20321 122640  001427                            BEQ     1$                      ;SKIP PRINTOUTS
20322 122642  005737  001206                    TST     $PASS                   ;1ST PASS?
20323 122646  001024                            BNE     1$                      ;IF NOT, SKIP PRINTOUTS
20324 122650  122737  000001  001220            CMPB    #APTENV,$ENV            ;APT MODE?
20325 122656  001420                            BEQ     1$                      ;YES, SKIP PRINTOUT'S
20326 122660  005105                            COM     R5                      ;CLEAR FLAG IN INTERRUPT MODE
20327 122662  104401  001175                    TYPE    ,$CRLF
20328 122666  104401  123016                    TYPE    ,LEDS                   ;IDENTIFY THE TEST
20329 122672  012737  122770  000060            MOV     #5$,@#60                ;RECEIVE SLU VECTOR
20330 122700  012737  000340  000062            MOV     #340,@#62               ;AT PRIORITY 7
20331 122706  052737  000100  177560            BIS     #BIT06,RCSR             ;ENABLE INTERRUPTS
20332 122714  106427  000140                    MTPS    #140                    ;LOWER PRIORITY
20333 122720  012704  000006          1$:       MOV     #6,R4                   ;FOR EACH LOOP
20334 122724  012701  000076                    MOV     #76,R1                  ;START WITH 1
20335 122730  110137  177524          2$:       MOVB    R1,BDR                  ;TURN OFF FIRST LED
20336 122734  012703  000004                    MOV     #4,R3                   ;STALL DELAY
20337 122740  012702  177777          3$:       MOV     #177777,R2              ;STALL DELAY
20338 122744  077201                  4$:       SOB     R2,4$                   ;WAIT A WHILE
20339 122746  077304                            SOB     R3,3$                   ;WAIT A WHILE
20340 122750  000261                            SEC                             ;SET CARRY
20341 122752  006101                            ROL     R1                      ;GET ANOTHER LED
20342 122754  077413                            SOB     R4,2$                   ;DO A FEW TIMES
20343 122756  005705                            TST     R5                      ;RUNNING IN INTERACTIVE MODE?
20344 122760  001407                            BEQ     6$                      ;IF NOT, EXIT
20345 122762  104401  001170                    TYPE    ,$BELL                  ;
20346 122766  000754                            BR      1$                      ;REPEAT PATTERN
20347 122770  005737  177562          5$:       TST     RBUF                    ;READ BUFFER
20348 122774  062706  000004                    ADD     #4,SP                   ;ADJUST STACK
20349 123000  112737  000377  177524  6$:       MOVB    #377,BDR                ;NO MORE
20350 123006  042737  001000  177520            BIC     #1000,BCSR              ;DISABLE HOB FOR APT
20351 123014  000452                            BR      TST55           ;;EXIT TEST
20352
20353 123016      012     015     124  LEDS:     .ASCII  <12><15>/THIS IS A TEST FOR ON-BOARD LED'S/<12><15>
      123021      110     111     123
      123024      040     111     123
      123027      040     101     040
      123032      124     105     123
      123035      124     040     106
      123040      117     122     040
      123043      117     116     055
```

G14

SEQ 0382

TEST - LED'S ON

```
                123046  102     117     101
                123051  122     104     040
                123054  114     105     104
                123057  047     123     012
                123062  015
 20354          123063  124     131     120             .ASCIZ  /TYPE ANY CHARACTER ON A KEYBOARD TO CONTINUE/<12><15>
                123066  105     040     101
                123071  116     131     040
                123074  103     110     101
                123077  122     101     103
                123102  124     105     122
                123105  040     117     116
                123110  040     101     040
                123113  113     105     131
                123116  102     117     101
                123121  122     104     040
                123124  124     117     040
                123127  103     117     116
                123132  124     111     116
                123135  125     105     012
                123140  015     000
 20355                                          .EVEN
 20356
```

```
                                .SBTTL  TEST    MEMORY MAPPING
                                ;MEMORY MAPPING
                                ;THIS TEST WILL AUTOSIZE MEMORY IN 2K BYTES. EVERY PAGE WILL BE
                                ;CHECKED FOR WHAT TYPE IT IS: Q BUS, UNIBUS OR PMI BUS MEMORY BY
                                ;SEEING HOW IT CAN BE CACHED.
                                ;ROUTINE TEST
                                ;.  LET 4=ADDRESS OF NON-EXISTENT_MEMORY
                                ;.  IF KMCR<05 00> NE #1 THEN (NOT ALL UNIBUS MEMORY)
                                ;.      TURN ON MMU AND REMAP THE PROGRAM AREA
                                ;.      REPEAT
                                ;.      .  DO FOR KDPAR0 FROM #0 TO #177600
                                ;.      .       DO FOR R1 FROM #0 TO #20000 BY #4000
                                ;.      .       .       READ (R1)
                                ;.      .       .       IF ABORT_NON-EXISTENT_MEMORY NE 1
                                ;.      .       .       .       MEMORY EXISTS
                                ;.      .       .       .       READ (R1)+
                                ;.      .       .       .       READ (R1)
                                ;.      .       .       .       IF HIT/MISS EQ 2_HITS THEN
                                ;.      .       .       .       .       PMI MEMORY
                                ;.      .       .       .       ELSE
                                ;.      .       .       .       .       IF HIT/MISS EQ HIT THEN
                                ;.      .       .       .       .       .       Q BUS MEMORY
                                ;.      .       .       .       .       ELSE
                                ;.      .       .       .       .       .       ERROR
                                ;.      .       .       .       .       ENDIF
                                ;.      .       .       .       ENDIF
                                ;.      .       .       ENDIF
                                ;.      .       .       LET ABORT_NON EXISTENT MEMORY=#0
                                ;.      .       ENDDO
                                ;.      .  ENDDO
                                ;.      UNTILL ABORT NON-EXISTANT MEMORY EQ #1
                                ;.  ENDIF
                                ;.
                                ;ENDROUTINE
                                ;
                                ;ROUTINE NON-EXISTENT_MEMORY
                                ;.      LET ABORT_NON-EXISTENT_MEMORY=#1
                                ;.      RETURN
                                ;ENDROUTINE

                                ;;*****************************************************************
    123142  000004              TST55:  SCOPE
20399 123144  032737  000001  000052        BIT     #BIT00,@#52                     ;CHAIN MODE?
20400 123152  001561                        BEQ     TST56           ;;IF SO, EXIT TEST
20401 123154  005737  001206                TST     $PASS                           ;FIRST PASS?
20402 123160  001156                        BNE     TST56           ;;IF APT AND NOT FIRST PASS, EXIT TEST
20403 123162  122737  000001  001220        CMPB    #APTENV,$ENV                    ;APT MODE?
20404 123170  001552                        BEQ     TST56           ;;YES, SKIP PRINTOUT'S
20405                               ;
20406                               ; SETUP ALL REGISTERS FOR MAPPING
20407                               ;
20408 123172  012737  000400  177746        MOV     #400,CCR                        ;FLUSH THE CACHE
20409 123200  013704  000004                MOV     ERRVEC,R4                       ;STORE TIMEOUT VECTOR
20410 123204  012737  123354  000004        MOV     #7$,@#ERRVEC                    ;POINT TO PROGRAM AREA
20411 123212  012737  000340  000006        MOV     #340,ERRVEC+2                   ;AT PRIORITY 4
20412 123220  004737  136574                JSR     PC,INITMM                       ;REMAP PROGRAM AREA
20413 123224  005037  172354                CLR     KIPAR6                          ;USED FOR MAPPING MEMORY
```

```
TEST  - MEMORY MAPPING

 20414 123230  005002                          CLR     R2                      ;CLEAR PAGE COUNT FOR PMI
 20415 123232  005003                          CLR     R3                      ;CLEAR PAGE COUNT FOR QBUS
 20416 123234  005237  177572                  INC     MMR0                    ;ENABLE MEMORY MANGEMENT
 20417 123240  052737  000020  172516          BIS     #BIT04,MMR3             ;ENABLE 22 BITS
 20418 123246  000403                          BR      2$                      ;GO ACCESS
 20419                                 ;
 20420                                 ; TRY TO MAP ALL PAGES
 20421                                 ;
 20422 123250  062737  000200  172354  1$:     ADD     #200,KIPAR6             ;INCREMENT BY 4K WORDS
 20423 123256  012701  140000          2$:     MOV     #140000,R1              ;ACCESS THRL KIPAR6
 20424 123262  000402                          BR      4$                      ;START DOING IT
 20425 123264  062701  003776          3$:     ADD     #3776,R1                ;INCREMENT BY 1K WORDS
 20426 123270  005721                  4$:     TST     (R1)+                   ;ACCESS 1ST LOCATION
 20427 123272  042737  001000  177520          BIC     #1000,BCSR              ;DISABLE HALT ON BREAK
 20428 123300  005711                          TST     (R1)                    ;ACCESS 2ND LOCATION
 20429 123302  013737  177752  114150          MOV     HITMIS,RECDAT           ;STORE REGISTER
 20430 123310  052737  001000  177520          BIS     #1000,BCSR              ;ENABLE HALT ON BREAK
 20431 123316  032737  000004  114150          BIT     #BIT02,RECDAT           ;LAST (R1) HIT?
 20432 123324  001402                          BEQ     5$                      ;IF NOT, QBUS MEMORY
 20433 123326  005202                          INC     R2                      ;INCREMENT 1K COUNT FOR PMI
 20434 123330  000401                          BR      6$                      ;GO CONITNUE
 20435 123332  005203                  5$:     INC     R3                      ;INCREMENT COUNT FOR QBUS MEM.
 20436 123334  022701  154000          6$:     CMP     #154000,R1              ;LAST IN 4K PAGE?
 20437 123340  101351                          BHI     3$                      ;IF NOT, BRANCH
 20438 123342  022737  177600  172354          CMP     #177600,KIPAR6          ;2M BOUNDARY?
 20439 123350  001337                          BNE     1$                      ;IF NOT, BRANCH
 20440 123352  000402                          BR      8$                      ;IF 2M, DON'T TOUCH STACK
 20441                                 ;
 20442                                 ; MAPPING IS DONE, FIND OUT HOW MANY PAGES WERE THERE
 20443                                 ;
 20444 123354  005726                  7$:     TST     (SP)+                   ;ADJUST STACK
 20445 123356  005726                          TST     (SP)+
 20446 123360  010437  000004          8$:     MOV     R4,ERRVEC               ;RESTORE ERROR VECTOR
 20447 123364  005037  177572                  CLR     MMR0                    ;DISABLE MEMORY MANGEMENT
 20448 123370  042737  000020  172516          BIC     #BIT04,MMR3             ;DISABLE 22 BITS
 20449 123376  005702                          TST     R2                      ;ANY PMI MEMORY?
 20450 123400  001405                          BEQ     9$                      ;IF NOT, GO CHECK QBUS MEMORY
 20451 123402  006302                          ASL     R2                      ;TRANSLATE TO BYTES
 20452 123404  010246                          MOV     R2,-(SP)                ;STORE 1K # ON STACK
 20453 123406  104405                          TYPDS                           ;TYPE # OF PAGES
 20454 123410  104401  123434                  TYPE    ,MEMK                   ;TYPE ASCII
 20455 123414  005703                  9$:     TST     R3                      ;ANY Q BUS MEMORY?
 20456 123416  001405                          BEQ     10$                     ;IF NOT, BRANCH
 20457 123420  006303                          ASL     R3                      ;TRANSLATE TO BYTES
 20458 123422  010346                          MOV     R3,-(SP)                ;STORE 1K # ON STACK
 20459 123424  104405                          TYPDS                           ;TYPE # OF PAGES
 20460 123426  104401  123464                  TYPE    ,MEMQ                   ;TYPE ASCII
 20461 123432                          10$:
       123432  000431                          BR      TST56           ;;EXIT TEST
 20462
 20463 123434     113     040     102  MEMK:   .ASCIZ  /K BYTES OF PMI MEMORY/<12><15>
       123437     131     124     105
       123442     123     040     117
       123445     106     040     120
       123450     115     111     040
       123453     115     105     115
       123456     117     122     131
```

```
123461   012     015     000
123464   113     040     102    MEMQ:   .ASCIZ  /K BYTES OF Q BUS MEMORY/<12><15>
123467   131     124     105
123472   123     040     117
123475   106     040     121
123500   055     102     125
123503   123     040     115
123506   105     115     117
123511   122     131     012
123514   015     000
                                        .EVEN

                                        .SBTTL WRONG PARITY ABORT TEST
                                        ;WRONG PARITY ABORT
                                        ;THIS TEST VERIFIES ABORT TO 114 USING PARITY OR ECC MEMORY CSR.
                                        ;IF MORE THEN 1 CSR PRESENT, ALL OF THEM WILL BE WRITTEN AT THE
                                        ;SAME TIME.
                                        ;
                                        ;ROUTINE TEST
                                        ;.      SIZE FOR ALL POSSIBLE MEMORY CSR'S
                                        ;.      STORE UP TO 16 CSR STARTING FROM TEMP
                                        ;.      WRITE ALL WITH WRONG PARITY
                                        ;.      WRITE TO 0
                                        ;.      READ IT BACK
                                        ;.      IF NO ABORT TO 114 THEN
                                        ;.      .       ERROR IN PARITY ABORT LOGIC
                                        ;.      ENDIF
                                        ;.      RESTORE CSR'S
                                        ;ENDROUTINE

                                        ;;****************************************************************
123516  000004                          TST56:  SCOPE
                                        ;
                                        ; FIND OUT ALL POSSIBLE MEMORY CSR LOCATIONS UP TO 16
                                        ;
123520  013701  000004                          MOV     ERRVEC,R1               ;STORE TIMEOUT VECTOR
123524  012737  123562  000004                  MOV     #2$,ERRVEC              ;POINT NEW TO PROGRAM
123532  012737  000340  000006                  MOV     #340,ERRVEC+2           ;AT PRIORITY 7
123540  005004                                  CLR     R4                      ;COUNT FOR CSR'S
123542  012702  172100                          MOV     #172100,R2              ;FIRST POSSIBLE CSR
123546  012703  002740                          MOV     #TEMP,R3                ;STORAGE LOCATION
123552  005712                          1$:     TST     (R2)                    ;IS CSR THERE?
123554  010223                                  MOV     R2,(R3)+                ;IF THERE, STORE
123556  005204                                  INC     R4                      ;INCREMENT COUNT FOR CSR S
123560  000402                                  BR      3$                      ;BRANCH AROUND
123562  005726                          2$:     TST     (SP)+                   ;RESTORE STACK
123564  005726                                  TST     (SP)+
123566  062702  000002                  3$:     ADD     #2,R2                   ;POINT TO NEW CSR
123572  022702  172136                          CMP     #172136,R2              ;ALL DONE?
123576  101365                                  BHI     1$                      ;IF NOT, BRANCH
123600  010137  000004                          MOV     R1,ERRVEC               ;RESTORE ERROR VECTOR
123604  052737  001000  177746                  BIS     #BIT09,CCR              ;SET CACHE BYPASS
                                        ;
                                        ; WRITE ALL CSR'S WITH WRONG ECC CODE
                                        ; NOTE: IN PARITY MEMORY THOSE BITS ARE READ ONLY AND
                                        ;       DIAGNOSTIC MODE BIT FOR ECC IS THE SAME AS WRONG PARITY
                                        ;
```

WRONG PARITY ABORT TEST

```
20511 123612  013701  000114                 MOV     @#114,R1                ;STORE PARITY ABORT VECTOR
20512 123616  012737  123704  000114         MOV     #6$,@#114               ;POINT NEW TO PROGRAM
20513 123624  012737  000340  000116         MOV     #340,@#116              ;AT PRIORITY 7
20514 123632  010402                         MOV     R4,R2                   ;STORE CSR COUNT
20515 123634  012703  002740                 MOV     #TEMP,R3                ;START WITH 1ST CSR
20516 123640  052773  000005  000000  4$:    BIS     #BIT02!BIT00,@0(R3)     ;DIAGNOSTIC OR WRONG PARITY
20517 123646  042733  003740                 BIC     #3740,@(R3)+            ;CLEAR <10 5> CHECK BITS
20518 123652  077406                         SOB     R4,4$                   ;DO FOR ALL CSR S
20519 123654  005037  000000                 CLR     @#0                     ;CLEAR TEST LOCATION WITH WRONG PR
20520 123660  010204                         MOV     R2,R4                   ;RESTORE COUNTER
20521 123662  012703  002740                 MOV     #TEMP,R3                ;START WITH 1ST CSR
20522 123666  042733  000004          5$:    BIC     #BIT02,@(R3)+           ;CLEAR ALL WRONG PARITY
20523 123672  077203                         SOB     R2,5$                   ;IN ALL CSR S
20524 123674  005737  000000                 TST     @#0                     ;READ BACK WRONG PARITY
20525 123700  104034                         ERROR   +34                     ;NO WRONG PARITY ABORT
20526 123702  000402                         BR      7$
20527 123704  005726                 6$:     TST     (SP)+                   ;ADJUST STACK
20528 123706  005726                         TST     (SP)+
20529 123710  012703  002740          7$:    MOV     #TEMP,R3                ;START WITH 1ST CSR
20530 123714  042733  000001          8$:    BIC     #BIT00,@(R3)+           ;CLEAR ALL WRONG PARITY
20531 123720  077403                         SOB     R4,8$                   ;IN ALL CSR'S
20532 123722  032737  100000  177744         BIT     #BIT15,MSER             ;ABORT REFLECTED IN MSER?
20533 123730  001004                         BNE     9$                      ;IF YES, BRANCH
20534 123732  012737  100000  001124         MOV     #100000,$GDDAT          ;
20535 123740  104035                         ERROR   +35                     ;MSER<15> NOT SET ON PARITY ABORT
20536 123742  042737  001000  177746  9$:    BIC     #BIT09,CCR              ;CLEAR CACHE BYPASS
20537 123750  005037  000000                 CLR     @#0                     ;CLEAR WRONG PARITY
20538 123754  005037  177744                 CLR     MSER                    ;CLEAR MSER
20539 123760  010137  000114                 MOV     R1,@#114                ;RESTORE PARITY ABORT
20540
20541
```

TEST - DMA TAG PARITY IN STANDALONE MODE

```
20543                                   .SBTTL  TEST  - DMA TAG PARITY IN STANDALONE MODE
20544                                   ;CHECK DMA TAG STORE PARITY BIT.
20545                                   ;ROUTINE TEST
20546                                   ;.      CACHE DMA_PARITY
20547                                   ;.      LET BCSR<08>=#1
20548                                   ;.      REPORT ALL ERRORS
20549                                   ;ENDROUTINE
20550                                   ;
20551                                   ;ROUTINE DMA_PARITY
20552                                   ;.      GENERATE PARITY ERRORS
20553                                   ;.      CHECK MSER<13>
20554                                   ;.      LET BCSR<08>=#0
20555                                   ;ENDROUTINE
20556
20557                                   ;;*****************************************************************
     123764  000004                     TST57:  SCOPE
20558 123766  000240                             NOP
20559 123770  005737  003032                     TST     CCHPAS                  ;have done enough inclus've passes?
20560 123774  001002                             BNE     99$                     ; not yet
20561 123776  000240                             NOP                             ; debug aid
20562 124000  000456                             BR      TST60           ;;GO TO NEXT TEST
20563 124002  000240                     99$:    NOP
20564                                   ;
20565                                   ; ALLOCATE CODE IN CACHE
20566                                   ;
20567 124004  012702  124040                     MOV     #DMAPAR,R2                      ;POINT TO STANDALONE CODE
20568 124010  042737  001000  177520             BIC     #1000,BCSR                      ;DISABLE HALT ON BREAK
20569 124016  005722                     1$:     TST     (R2)+                           ;ALLOCATE IN CACHE
20570 124020  022702  124100                     CMP     #DPAREN,R2                      ;LAST ADDRESS?
20571 124024  001374                             BNE     1$                              ;IF NOT, BRANCH
20572 124026  005737  002740                     TST     TEMP                            ;ALLOCATE TEST LOCATION
20573 124032  013737  177520  002730             MOV     BCSR,SAVBR                      ;SAVE BCSR
20574                                   ;
20575                                   ; IN STANDALONE MODE TRY TO VERIFY RESPONSE TO PARITY ERRORS
20576                                   ;
20577 124040  052737  000400  177520   DMAPAR: BIS     #BIT08,BCSR                     ;SET STANDALONE MODE BIT
20578 124046  052737  002001  177746             BIS     #BIT10!BIT00,CCR                ;WRITE WRONG TAG PARITY
20579 124054  005037  002740                     CLR     TEMP                            ;WRITE HIT WITH WRONG PARITY
20580 124060  013705  177744                     MOV     MSER,R5                         ;STORE MSER
20581 124064  042737  002000  177746             BIC     #BIT10,CCR                      ;CLEAR WRONG PARITY BIT
20582 124072  042737  000400  177520   2$:     BIC     #BIT08,BCSR                     ;CLEAR STANDALONE BIT
20583                                   ;
20584                                   ; RETURN FROM STANDALONE MODE
20585                                   ;
20586 124100                            DPAREN:
20587 124100  022705  060020                     CMP     #60020,R5                       ;WRONG DMA PARITY?
20588 124104  001401                             BEQ     4$                              ;IF OK, BRANCH
20589 124106  104117                             ERROR   #117                            ;MSER NOT SET IN STANDALONE MODE
20590 124110  005037  177744             4$:     CLR     MSER
20591 124114  012737  000400  177746             MOV     #400,CCR                        ;FLUSH THE CACHE
20592 124122  013737  002730  177520             MOV     SAVBR,BCSR                      ;RESTORE BCSR
20593 124130  052737  001000  177520             BIS     #1000,BCSR                      ;ENABLE HALT ON BREAK
20594
```

```
                                        .SBTTL  TEST  - DMA TAG PARITY W/O STANDALONE MODE
                                        ;CHECK DMA TAG PARITY BIT WITHOUT GOING INTO STANDALONE MODE.
                                        ;
                                        ;CCR    <10>    WRITE WRONG TAG PARITY
                                        ;
                                        .ROUTINE TEST
                                        ;.      LET CCR<10>=#1
                                        ;.      ALLOCATE LOCATION IN CACHE
                                        ;.      INITIATE DMA WRITE TRANSFERS
                                        ;.      IF MSER<04> NE #1 THEN
                                        ;.              ERROR
                                        ;.      ENDIF
                                        ;ENDROUTINE

                                        ;;*****************************************************************
     124136  000004                     TST60:  SCOPE
20611 124140  000240                             NOP
20612 124142  005737  003032                     TST     CCHPAS                  ;have done enough inclus ve passes?
20613 124146  001002                             BNE     99$                     ; not yet
20614 124150  000240                             NOP                             ; debug aid
20615 124152  000471                             BR      TST61           ;;GO TO NEXT TEST
20616 124154  000240                     99$:    NOP
20617
20618 124156  032737  000200  000052             BIT     #BIT07,@#52                     ;UFD MODE?
20619 124164  001402                             BEQ     110$                            ;IF NOT, BRANCH
20620 124166  000137  140340                     JMP     $EOP                            ;OTHERWISE, NEXT PASS
20621 124172  005737  002664             110$:   TST     CSR1                            ;AT LEAST ONE Q22BE FOUND?
20622 124176  001457                             BEQ     TST61           ;;IF NOT, EXIT TEST
20623                                   ;
                                        ; ALLOCATE TEST IN CACHE
                                        ;
20626 124200  012703  124200             11$:   MOV     #.,R3                           ;START WITH CURRENT INSTRUCTION
20627 124204  005723                     10$:   TST     (R3)+                           ;READ A WORD
20628 124206  022703  124316                     CMP     #4$,R3                          ;ALL DONE?
20629 124212  001374                             BNE     10$                             ;IF NOT, CONTINUE
20630                                   ;
                                        ; WRITE A WORD WITH WRONG PARITY
                                        ;
20633 124214  013737  000114  001160     1$:    MOV     @#114,$TMP0                     ;STORE PARITY VECTOR
20634 124222  012737  124270  000114             MOV     #2$,@#114                       ;POINT TO TEST AREA
20635 124230  052737  002000  177746             BIS     #BIT10,CCR                      ;WRITE WRONG TAG PARITY
20636 124236  005037  002740                     CLR     TEMP                            ;WRITE MISS WITH WRONG TAG PARITY
20637 124242  042737  002000  177746             BIC     #BIT10,CCR                      ;CLEAR WRONG TAG PARITY
20638 124250  052737  000200  177746             BIS     #BIT07,CCR                      ;PARITY ABORT
20639 124256  005000                             CLR     R0                              ;FLAG TO DO 1 TRANSFER
20640 124260  004737  137370                     JSR     PC,DMATRN                       ;DO DMA WRITE TO TEMP THRU Q22BE
20641 124264  104116                             ERROR   +116                            ;NO PARITY ABORT
20642 124266  000413                             BR      4$                              ;BRANCH TO TEST MSER
20643 124270  013704  177744             2$:    MOV     MSER,R4                         ;STORE REGISTER
20644 124274  005037  177744                     CLR     MSER                            ;CLEAR MSER
20645 124300  005726                             TST     (SP)+                           ;
20646 124302  005726                             TST     (SP)+                           ;RESTORE STACK
20647 124304  005726                             TST     (SP)+
20648 124306  032704  000020             3$:    BIT     #BIT04,R4                       ;MSER OK?
20649 124312  001001                             BNE     4$                              ;IF SET, BRANCH
20650 124314  104116                             ERROR   +116                            ;MSER<4> NOT SET
20651 124316  005037  177744             4$:    CLR     MSER                            ;CLEAR MSER
```

N14

EQ 0389

TEST - DMA TAG PARITY W/O STANDALONE MODE

```
 20652 124322  012737  000400  177746          MOV     #400,CCR                ;FLUSH CACHE
 20653 124330  013737  001160  000114          MOV     $TMP0,@#114             ;RESTORE VECTOR
```

```
                                .SBTTL  TEST  - DMA WRITE HIT CYCLES
                                ;CHECK THAT DMA WRITE HITS INVALIDATE CACHE.
                                ;ROUTINE TEST
                                ;.      ALLOCATE A LOCATION IN CACHE
                                ;.      INITIATE DMA WRITE TO THIS LOCATION
                                ;.      READ THIS LOCATION BACK
                                ;.      IF IT IS CHANGED OR HIT/MISS EQ HIT
                                ;.          ERROR
                                ;.      ENDIF
                                ;.      WRITE TO THE FIRST 16 LOCATION THEIR ADDRESS
                                ;.      DO BLOCK MODE TRANSFER TO THOSE LOCATIONS
                                ;.      START READ WITH THE LAST LOCATION
                                ;.      IF RECORD ANY HITS THEN
                                ;.          ERROR
                                ;.      ENDIF
                                ;.      INITIATE READ DMA TO THE SAME TEST LOCATIONS
                                ;.      READ THEM BACK
                                ;.      IF HIT/MISS NE HIT THEN
                                ;.          ERROR
                                ;.      ENDIF
                                ;ENDROUTINE

                                ;;*****************************************************************
     124336  000004             TST61:  SCOPE
20678 124340  000240                    NOP
20679 124342  005737  003032            TST     CCHPAS                  ;have done enough inclusive passes?
20680 124346  001003                    BNE     99$                     ; not yet
20681 124350  000240                    NOP                             ; debug aid
20682 124352  000137  125304            JMP     ALLEND                  ; yes skip this
20683 124356  000240              99$:  NOP
20684
20685 124360  032737  000200  000052    BIT     #BIT07,@#52                             ;UFD MODE?
20686 124366  001402                    BEQ     1$                                      ;IF NOT, BRANCH
20687 124370  000137  140340            JMP     $EOP                                    ;OTHERWISE, NEXT PASS
20688 124374  005737  002664      1$:   TST     CSR1                                    ;AT LEAST 1 Q22BE?
20689 124400  001002                    BNE     2$                                      ;IF YES, BRANCH
20690 124402  000137  140340            JMP     $EOP                                    ;OTHERWISE, NEXT PASS
20691                                   ;
20692                                   ; TRY TO DO DMA WRITE AT ALL DIFFERENT PRIORITIES
20693                                   ;
20694 124406  012702  000340      2$:   MOV     #340,R2                                 ;START WITH 7
20695 124412  000402                    BR      4$                                      ;GO DO IT
20696 124414  162702  000040      3$:   SUB     #40,R2                                  ;LOWER PRIORITY
20697 124420  005037  002740      4$:   CLR     TEMP                                    ;CLEAR TEST LOCATION
20698 124424  005000                    CLR     R0                                      ;DO JUST 1 WORD
20699 124426  106402                    MTPS    R2                                      ;LOWER PRIORITY
20700 124430  004737  137370            JSR     PC,DMATRN                               ;DO DATO
20701 124434  042737  001000  177520    BIC     #1000,BCSR                              ;DISABLE HALT ON BREAK
20702 124442  005737  002740            TST     TEMP                                    ;STILL CACHED?
20703 124446  013737  177752  114150    MOV     HITMIS,RECDAT                           ;STORE HIT/MISS
20704 124454  052737  001000  177520    BIS     #1000,BCSR                              ;ENABLE HALT ON BREAK
20705 124462  032737  000004  114150    BIT     #BIT02,RECDAT                           ;LAST ACCESS HIT?
20706 124470  001401                    BEQ     5$                                      ;IF MISS, BRANCH
20707 124472  104120                    ERROR   +120
20708 124474  022737  012525  002740  5$:   CMP     #12525,TEMP                             ;DATO OK?
20709 124502  001401                    BEQ     6$                                      ;IF SO, BRANCH
20710 124504  104123                    ERROR   +123                                    ;DATO
```

```
TEST  - DMA WRITE HIT CYCLES

 20711 124506  005702                  6$:     TST     R2                      ;LAST PRIORITY 0?
 20712 124510  001341                          BNE     3$                      ;IF NOT, BRANCH
 20713 124512  106427  000340                  MTPS    #340                    ;RESTORE PRIORITY
 20714                                ;
 20715                                ;
 20716                                ; VERIFY THAT BYPASS WITH DMA DATI INVALIDATES CACHE
 20717                                ;
 20718 124516  012737  001000  177746  8$:     MOV     #BIT09,CCR              ;SET BYPASS
 20719 124524  004737  137370                  JSR     PC,DMATRN               ;DO DMA DATI
 20720 124530  005037  177746                  CLR     CCR                     ;CLEAR BYPASS
 20721 124534  042737  001000  177520        BIC     #1000,BCSR              ;DISABLE HALT ON BREAK
 20722 124542  005737  002740                  TST     TEMP                    ;IN CACHE?
 20723 124546  013737  177752  114150        MOV     HITMIS,RECDAT           ;STORE REGISTER
 20724 124554  052737  001000  177520        BIS     #1000,BCSR              ;ENABLE HALT ON BREAK
 20725 124562  032737  000004  114150        BIT     #BIT02,RECDAT           ;TEMP WAS A HIT?
 20726 124570  001401                        BEQ     DATBO                   ;IF NOT, BRANCH
 20727 124572  104123                        ERROR   +123
 20728                                ;
 20729                                ; DO DATBO
 20730                                ;
 20731 124574  012701  000010          DATBO:  MOV     #10,R1                  ;COUNTER FOR 8
 20732 124600  012702  002740                  MOV     #TEMP,R2                ;START WITH TEMP
 20733 124604  005022                  1$:     CLR     (R2)+                   ;CLEAR ALL 16
 20734 124606  077102                          SOB     R1,1$                   ;DO ALL 16
 20735 124610  005200                          INC     R0                      ;FLAG BLOCK MODE
 20736 124612  012777  002740  056050        MOV     #TEMP,@BA               ;LOAD DMA ADDRESS
 20737 124620  012777  177770  056044        MOV     #177770,@WC             ;DO 8 WORDS
 20738 124626  012777  001701  056030        MOV     #1701,@CSR1             ;16 WORDS FROM 32K
 20739 124634  004737  137370                  JSR     PC,DMATRN               ;DO DATBO
 20740 124640  032777  010000  056020        BIT     #BIT12,@CSR2            ;NO BLOCK MODE SLAVE?
 20741 124646  001401                        BEQ     2$                      ;IF NOT, BRANCH
 20742 124650  104123                        ERROR   +123                    ;NO BLOCK MODE SLAVE
 20743 124652  012701  000004          2$:     MOV     #4,R1                   ;COUNTER FOR 4 LOCATIONS
 20744 124656  012702  002740                  MOV     #TEMP,R2                ;START WITH TEMP
 20745 124662  042737  001000  177520  3$:     BIC     #1000,BCSR              ;DISABLE HALT ON BREAK
 20746 124670  005712                          TST     (R2)                    ;ACCESS A LOCATION
 20747 124672  013737  177752  114150        MOV     HITMIS,RECDAT           ;STORE REGISTER
 20748 124700  052737  001000  177520        BIS     #1000,BCSR              ;ENABLE HALT ON BREAK
 20749 124706  032737  000004  114150        BIT     #BIT02,RECDAT           ;HIT?
 20750 124714  001401                        BEQ     4$                      ;IF NOT, BRANCH
 20751 124716  104121                        ERROR   +121                    ;DMA DOES NOT INVALIDATE
 20752 124720  022722  012525          4$:     CMP     #12525,(R2)+            ;DATO OK?
 20753 124724  001401                        BEQ     5$                      ;IF SO,BRANCH
 20754 124726  104123                        ERROR   +123                    ;IN DMA
 20755 124730  062702  000002          5$:     ADD     #2,R2                   ;DO IN 2 WORDS
 20756 124734  077126                        SOB     R1,3$                   ;DO ALL 16
 20757
 20758                                ;
 20759                                ; DO 4K OF DATO AND CHECK FOR MISS
 20760                                ;
 20761 124736  004737  136574                  JSR     PC,INITMM               ;SET UP MMU
 20762 124742  012737  002000  172354        MOV     #2000,KIPAR6            ;START AT 32K
 20763 124750  042737  100000  172314        BIC     #BIT15,KIPDR6           ;NO BYPASS
 20764 124756  052737  000001  177572        BIS     #BIT00,MMR0             ;ENABLE MMU
 20765 124764  012737  125124  000004        MOV     #DATI,@#4               ;IF 32K NXW
 20766 124772  012702  140000                  MOV     #140000,R2              ;START WITH 32K
 20767 124776  005712                          TST     (R2)                    ;EXIT
```

TEST - DMA WRITE HIT CYCLES

```
20768 125000  012701  010000          MOV     #10000,R1               ;COUNTER FOR 4K
20769 125004  005022            6$:   CLR     (R2)+                   ;CLEAR ALL 16
20770 125006  077102                  SOB     R1,6$                   ;DO ALL 16
20771 125010  005200                  INC     R0                      ;FLAG BLOCK MODE
20772 125012  012777  000000  055650  MOV     #0,@BA                  ;LOAD DMA ADDRESS
20773 125020  012777  170000  055644  MOV     #-10000,@WC             ;DO 4K
20774 125026  012777  003701  055630  MOV     #3701,@CSR1             ;16 WORDS FROM 32K
20775 125034  004737  137370          JSR     PC,DMATRN               ;DO DATBO
20776 125040  012701  004000    7$:   MOV     #4000,R1                ;COUNTER FOR 4K LOCATIONS
20777 125044  012702  140000          MOV     #140000,R2              ;START WITH 0
20778 125050  042737  001000  177520  8$:   BIC     #1000,BCSR              ;DISABLE HALT ON BREAK
20779 125056  005712                  TST     (R2)                    ;ACCESS A LOCATION
20780 125060  013737  177752  114150  MOV     HITMIS,RECDAT           ;STORE REGISTER
20781 125066  052737  001000  177520  BIS     #1000,BCSR              ;ENABLE HALT ON BREAK
20782 125074  032737  000004  114150  BIT     #BIT02,RECDAT           ;HIT?
20783 125102  001401                  BEQ     9$                      ;IF NOT, BRANCH
20784 125104  104121                  ERROR   +121                    ;DMA DOES NOT INVALIDATE
20785 125106  022722  012525    9$:   CMP     #12525,(R2)+            ;DATO OK?
20786 125112  001401                  BEQ     10$                     ;IF SO,BRANCH
20787 125114  104123                  ERROR   +123                    ;IN DMA
20788 125116  062702  000002    10$:  ADD     #2,R2                   ;DO IN 2 WORDS
20789 125122  077126                  SOB     R1,8$                   ;DO ALL OF THEM
20790                                 ;
20791                                 ; CHECK DATI
20792                                 ;
20793 125124  005037  177572  DATI:   CLR     MMR0                    ;DISABLE MMU
20794 125130  012706  001100          MOV     #1100,SP                ;RESTORE STACK
20795 125134  012737  137542  000004  MOV     #TOUT,@#4               ;AND TIMEOUT
20796 125142  012737  052525  002740  MOV     #52525,TEMP             ;LOAD MEMORY
20797 125150  005000                  CLR     R0                      ;JUST 1 WORD
20798 125152  004737  137452          JSR     PC,DMARD                ;DO DATI
20799 125156  022777  052525  055510  CMP     #52525,@DATA            ;DATI OK?
20800 125164  001401                  BEQ     11$                     ;IF YES, BRANCH
20801 125166  104123                  ERROR   +123                    ;DATI
20802                                 ;
20803                                 ; DO DATBI
20804                                 ;
20805 125170  012701  000010    11$:  MOV     #10,R1                  ;COUNTER FOR 16 LOCATIONS
20806 125174  012702  002740          MOV     #TEMP,R2                ;START WITH TEMP
20807 125200  012703  002740          MOV     #TEMP,R3                ;START WITH TEMP
20808 125204  010322    12$:  MOV     R3,(R2)+                ;PUT ADDRESSES
20809 125206  005723                  TST     (R3)+                   ;INCREMENT ADDRESS
20810 125210  077103                  SOB     R1,12$                  ;DO ALL 16
20811 125212  005200                  INC     R0                      ;FLAG BLOCK MODE
20812 125214  004737  137452          JSR     PC,DMARD                ;DO DATBI
20813 125220  032777  010000  055440  BIT     #BIT12,@CSR2            ;NO BLOCK MODE SLAVE?
20814 125226  001401                  BEQ     13$                     ;IF NO, BRANCH
20815 125230  104123                  ERROR   +123
20816 125232  022777  002756  055434  13$:  CMP     #TEMP+16,@DATA          ;DATI OK?
20817 125240  001401                  BEQ     14$                     ;IF SO, BRANCH
20818 125242  104123                  ERROR   +123                    ;IN DATIB
20819 125244  042737  001000  177520  14$:  BIC     #1000,BCSR              ;DISABLE HALT ON BREAK
20820 125252  005737  002740          TST     TEMP                    ;ACCESS
20821 125256  013737  177752  114150  MOV     HITMIS,RECDAT           ;STORE REGISTER
20822 125264  052737  001000  177520  BIS     #1000,BCSR              ;ENABLE HALT ON BREAK
20823 125272  032737  000004  114150  BIT     #BIT02,RECDAT           ;HIT?
20824 125300  001001                  BNE     15$                     ;IF YES, BRANCH
```

E15

SEQ 0393

TEST - DMA WRITE HIT CYCLES

```
 20825 125302  104123                        ERROR   +123
 20826 125304                        15$:
 20827 125304                        ALLEND:
```

F15


```
20829                                   .SBTTL  TEST    DIFFERENT LEVELS OF INTERRUPTS
20830                                   ;DIFFERENT LEVELS OF INTERRUPTS
20831                                   ;THIS TEST WILL PROGRAM Q22 BUS EXERCISER TO INTERRUPT AT DIFFERENT
20832                                   ;LEVELS.  ARBITRATION BETWEEN DIFFERENT LEVELS OF INTERRUPTS AND
20833                                   ;PIRQ'S WILL BE TESTED.
20834                                   ;
20835                                   ;CHECK DIFFERENT LEVELS OF INTERRUPTS.
20836                                   ;ROUTINE TEST
20837                                   ;.      SET VECTOR TO INTERRUPT_DMA
20838                                   ;.      FOR INTERRUPTS FROM 4 TO 7 DO
20839                                   ;.      .       ENABLE INTERRUPTS
20840                                   ;.      .       SET PRIORITY=INTERUPT
20841                                   ;.      .       IF INTERRUPT_FLAG SET THEN
20842                                   ;.      .       .       ERROR
20843                                   ;.      .       ENDIF
20844                                   ;.      .       ENABLE INTERRUPTS
20845                                   ;.      .       SET PRIORITY=INTERRUPT 1
20846                                   ;.      .       IF INTERRUPT_FLAG NOTSET THEN
20847                                   ;.      .       .       ERROR
20848                                   ;.      .       ENDIF
20849                                   ;.      .       LET INTERRUPT_DMA=0
20850                                   ;.      ENDDO
20851                                   ;ENDROUTINE
20852                                   ;
20853                                   ;ROUTINE INTERUPT_DMA
20854                                   ;.      LET INTERRUPT_FLAG=1
20855                                   ;RETURN
20856                                   ;ENDROUTINE
20857
20858                                   ;;*****************************************************************
        125304  000004                  TST62:  SCOPE
20859   125306  032737  000200  000052          BIT     #BIT07,@#52                     ;UFD MODE?
20860   125314  001122                  BNE     TST63           ;;IF SO, EXIT TEST
20861   125316  005737  002664          TST     CSR1                                    ;AT LEAST ONE Q22BE FOUND?
20862   125322  001517                  BEQ     TST63           ;;IF NOT, EXIT TEST
20863                                   ;
20864                                   ; SETUP INITIAL PRIORITY TO 7
20865                                   ;
20866   125324  013703  002664          MOV     CSR1,R3                                 ;DO FOR FIRST FOUND Q22BE
20867   125330  012777  125424  055340  MOV     #5$,@VQ8E1                              ;POINT INTERRUPT VECTOR TO PROGRAM
20868   125336  012777  000340  055334  MOV     #340,@VQPR1                             ;AT PRIORITY 7
20869   125344  012700  003002          MOV     #Q22EN,R0                               ;START WITH 7 FOR INTERRUPTS
20870   125350  012701  000340          MOV     #340,R1                                 ;LOW BOUNDARY FOR NO INTERRUPTS
20871   125354  000402                  BR      2$                                      ;TRY TO DO IT FOR FIRST
20872   125356  162701  000040          1$:     SUB     #40,R1                          ;LOWER LOW BOUNDARY
20873                                   ;
20874                                   ; CHECK THAT INTERRUPTS DON'T HAPPEN AT PRIORITY HIGHER THAN BR
20875                                   ;
20876   125362  012737  000340  001160  2$:     MOV     #340,$TMP0                      ;TOP PRIORITY FOR NO INTERRUPTS
20877   125370  000403                  BR      4$                                      ;DO FIRST ONE
20878   125372  162737  000040  001160  3$:     SUB     #40,$TMP0                       ;DO AT NEXT LEVEL
20879   125400  106437  001160          4$:     MTPS    $TMP0                           ;SET PRIORITY NOT TO INTERRUPT
20880   125404  004737  137350          JSR     PC,Q22INT                               ;ENABLE INTERRUPTS
20881   125410  012077  055252          MOV     (R0)+,@CSR2                             ;CLEAR GO BIT
20882   125414  000240                  NOP
20883   125416  000240                  NOP
20884   125420  000240                  NOP
```

TEST - DIFFERENT LEVELS OF INTERRUPTS

```
20885 125422  000403                        BR      6$                      ;IF NO INTERUPT, BRANCH
20886 125424  104126                5$:     ERROR   +126                    ;INTERRUPTS HAPPEN
20887 125426  005726                        TST     (SP)+                   ;RESTORE STACK
20888 125430  005726                        TST     (SP)+
20889 125432  020137  001160          6$:     CMP     R1,$TMP0                ;LAST ONE?
20890 125436  001355                        BNE     3$                      ;IF NOT BRANCH
20891 125440  022710  000002                CMP     #2,(R0)                 ;AT BR4?
20892 125444  001344                        BNE     1$                      ;IF NOT LAST ONE, BRANCH
20893                               ;
20894                               ; INTERRUPT AT ALL LEVELS
20895                               ;
20896 125446  012777  125542  055222  INQ22: MOV     #5$,@VQBE1              ;POINT INTERRUPT VECTOR TO PROGRAM
20897 125454  012777  000340  055216        MOV     #340,@VQPR1             ;AT PRIORITY 7
20898 125462  012700  003002                MOV     #Q22EN,R0               ;START WITH 7 FOR INTERRUPTS
20899 125466  012701  000300                MOV     #300,R1                 ;TOP BOUNDARY FOR INTERRUPTS
20900 125472  000402                        BR      2$                      ;TRY TO DO IT FOR FIRST
20901 125474  162701  000040          1$:     SUB     #40,R1                  ;LOWER TOP BOUNDARY
20902                               ;
20903                               ; CHECK THAT INTERRUPTS HAPPEN AT PRIORITY LOWER THAN BR
20904                               ;
20905 125500  010137  001160          2$:     MOV     R1,$TMP0                ;PRIORITY FOR INTERRUPTS
20906 125504  000403                        BR      4$                      ;DO FIRST ONE
20907 125506  162737  000040  001160  3$:     SUB     #40,$TMP0               ;DO AT NEXT LEVEL
20908 125514  106437  001160          4$:     MTPS    $TMP0                   ;SET PRIORITY NOT TO INTERRUPT
20909 125520  004737  137350                JSR     PC,Q22INT               ;ENABLE INTERRUPTS
20910 125524  011077  055136                MOV     (R0),@CSR2              ;CLEAR GO BIT
20911 125530  000240                        NOP                             ;WAIT A WHILE
20912 125532  000240                        NOP
20913 125534  000240                        NOP
20914 125536  104126                        ERROR   +126                    ;INTERRUPTS DON'T HAPPEN
20915 125540  000402                        BR      6$                      ;DON'T RESTORE STACK
20916 125542  005726                5$:     TST     (SP)+                   ;RESTORE STACK
20917 125544  005726                        TST     (SP)+
20916 125546  005737  001160          6$:     TST     $TMP0                   ;LAST ONE 0?
20919 125552  001355                        BNE     3$                      ;IF NOT BRANCH
20920 125554  022720  000002                CMP     #2,(R0)+                ;AT BR4?
20921 125560  001345                        BNE     1$                      ;IF NOT LAST ONE, BRANCH
20922
20923
```

```
20925                                   .SBTTL  TEST    ARBITRATION BETWEEN PIRQ'S AND INTERRUPTS
20926                                   ;CHECK PRIORITY ORDER BETWEEN PIRQ'S AND INTERRUPTS.
20927                                   ;ROUTINE TEST
20928                                   ;.      IF UFD THEN
20929                                   ;.      .       EXIT TEST
20930                                   ;.      ENDIF
20931                                   ;.      DO FOR I FROM #6 DOWN TO #3
20932                                   ;.      .       SET PRIORITY TO I
20933                                   ;.      .       ENABLE INTERRUPT(I+1) AND PIRQ(I+1)
20934                                   ;.      .       IF INTERRUPT(I+1) WAS BEFORE PIRQ(I+1) THEN
20935                                   ;.      .       .       ERROR
20936                                   ;.      .       ENDIF
20937                                   ;.      ENDDO
20938                                   ;ENDROUTINE
20939
20940                                   ;;*****************************************************************
        125562  000004                  TST63:  SCOPE
20941   125564  032737  000200  000052          BIT     #BIT07,@#52                     ;UFD MODE?
20942   125572  001065                  BNE     TST64           ;;IF SO, EXIT TEST
20943   125574  005737  002664          TST     CSR1                            ;AT LEAST ONE Q22BE FOUND?
20944   125600  001462                  BEQ     TST64           ;;IF NOT, EXIT TEST
20945   125602  012777  125710  055066  MOV     #3$,@VQBE1                      ;SETUP Q22BE VECTOR
20946   125610  012777  000340  055062  MOV     #340,@VQPR1                     ;AT PRIORITY 7
20947   125616  012737  125720  000240  MOV     #4$,PIRQVEC                     ;SETUP PIRQ VECTOR
20948   125624  012737  000340  000242  MOV     #340,PIRQVEC+2                  ;AT PRIORITY 7
20949   125632  012700  003002          MOV     #Q22EN,R0                       ;POINT THRU PRIORITIES FOR Q22BE
20950   125636  012704  125736          MOV     #PIRQT,R4                       ;POINTER THRU PIRQ'S
20951   125642  013703  002664          MOV     CSR1,R3                         ;DO FOR FIRST Q22BE
20952   125646  012702  000300          MOV     #300,R2                         ;START WITH CPU PRIORITY AT 7
20953   125652  000402                  BR      2$                              ;DO FIRST ONE
20954   125654  162702  000040  1$:     SUB     #40,R2                          ;LOWER CPU PRIORITY
20955   125660  106427  000340  2$:     MTPS    #340                            ;RAISE PRIORITY TO 7
20956   125664  012437  177772          MOV     (R4)+,PIRQ                      ;SET PRIORITY FOR PIRQ'S
20957   125670  004737  137350          JSR     PC,Q22INT                       ;INITIALISE Q22BE TO INTERRUPT
20958   125674  012077  054766          MOV     (R0)+,@CSR2                     ;SET DONE BIT
20959   125700  106402                  MTPS    R2                              ;LOWER PRIORITY
20960   125702  000240                  NOP
20961   125704  000240                  NOP
20962   125706  000240                  NOP
20963   125710  104124  3$:     ERROR   +124                            ;PIRQ'S DON'T TAKE OVER BIRQ'S
20964   125712  005726                  TST     (SP)+                           ;CLEAN UP STACK
20965   125714  005726                  TST     (SP)+
20966   125716  000402                  BR      5$                              ;BRANCH AROUND PIRQ INTERRUPT
20967   125720  005726  4$:     TST     (SP)+                           ;CLEAN UP STACK
20968   125722  005726                  TST     (SP)+
20969   125724  022702  000140  5$:     CMP     #140,R2                         ;PRIORITY 3 LAST ONE?
20970   125730  001351                  BNE     1$                              ;IF NOT BRANCH
20971   125732  005037  177772          CLR     PIRQ                            ;CLEAR ANY REQUESTS
20972
20973   125736  100000  040000  020000  PIRQT:  .WORD   100000,40000,20000,10000        ;PIRQ'S<7-4>
        125744  010000
20974
```

```
                                        .SBTTL  TEST      POWER DOWN TEST
                                        ;USING Q22BE THIS TEST WILL CHECK THAT ON POWER DOWN CONDITION IF
                                        ;POWER UP CODE 00 IS SELECTED THE CPU TRAPS THRU 24
                                        ;ROUTINE TEST
                                        ;.      IF UFD OR POWER UP CODE 00 NOT SELECTED THEN
                                        ;.              EXIT TEST
                                        ;.      ENDIF
                                        ;.      SET 24 TO POINT TO TEST AREA
                                        ;.      LET CSR2<5> = #1 TO NEGATE BPOK
                                        ;.      IF NO TRAP TO 24 THEN
                                        ;.              ERROR IN POWER DOWN CYCLE
                                        ;.      ENDIF
                                        ;.      IF TRAP TO 24 THEN
                                        ;.              LET CSR2<5> = #0
                                        ;.      ENDIF
                                        ;ENDROUTINE

                                        ;;*****************************************************************
125746 000004                           TST64:  SCOPE
125750 032737 000200 000052                     BIT     #BIT07,@#52                             ;UFD MODE?
125756 001033                                   BNE     TST65           ;;EXIT TEST IN UFD  MODE
125760 005737 002664                            TST     CSR1                                    ;AT LEAST ONE Q22BE FOUND?
125764 001430                                   BEQ     TST65           ;;IF NOT, EXIT TEST
125766 013737 000024 001160                     MOV     PWRVEC,$TMP0                            ;SAVE POWER UP VECTOR
125774 012737 126030 000024                     MOV     #1$,PWRVEC                              ;POINT NEW TO PROGRAM
126002 012737 000340 000026                     MOV     #340,PWRVEC+2                           ;AT PRIORITY 7
126010 012777 000040 054650                     MOV     #BIT05,@CSR2                            ;DO POWER DOWN
126016 000240                                   NOP
126020 005077 054642                            CLR     @CSR2                                   ;CLEAR POWER DOWN BIT
126024 104125                                   ERROR   +125                                    ;NO POWER DOWN TRAP
126026 000404                                   BR      2$                                      ;SKIP RESTORING STACK
126030 005077 054632                    1$:     CLR     @CSR2                                   ;CLEAR POWER DOWN BIT
126034 005726                                   TST     (SP)+                                   ;RESTORE STACK POINTER
126036 005726                                   TST     (SP)+
126040 013737 001160 000024             2$:     MOV     $TMP0,PWRVEC                            ;RESTORE POWER VECTOR
```

```
21014                                   .SBTTL TEST    - ARBITRATION BETWEEN DIFFERENT LEVELS OF INTERRUPTS
21015                                   ;IF TWO Q22BE ARE AVAILABLE, THIS TEST WILL CHECK THAT HIGHER LEVEL
21016                                   ;INTERRUPT REQUESTS TAKE PRIORITY OVER LOWER LEVEL ONES
21017                                    ROUTINE TEST
21018                                   ;.      IF UFD OR 2ND Q22BE NOT AVAILABLE THEN
21019                                   ;.      .       EXIT TEST
21020                                   ;.      ENDIF
21021                                   ;.      DO FOR PRIORITY LEVELS FROM 4 TO 6
21022                                   ;.      .       SET UP 1ST Q22BE TO INTERRUPT AT PRIORITY_LEVEL
21023                                   ;.      .       SET UP 2ND Q22BE TO INTERRUPT AT PRIORITY_LEVEL+1
21024                                   ;.      .       SET UP CPU PRIORITY TO PRIORITY_LEVEL 1
21025                                   ;.      .       IF INTERRUPTS FORM 1ST Q22BE HAPPENED BEFORE 1ST THEN
21026                                   ;.      .               ERROR
21027                                   ;.      .       ENDIF
21028                                   ;.      ENDDO
21029                                   ;ENDROUTINE
21030
21031                                   ;;*****************************************************************
      126046  000004                    TST65:  SCOPE
21032 126050  032737  000200  000052            BIT     #BIT07,@#52                     ;UFD MODE?
21033 126056  001064                            BNE     TST66           ;;IF SO, EXIT TEST
21034 126060  005737  002704                    TST     CSR12                           ;SECOND Q22BE AVAILABLE?
21035 126064  001002                            BNE     1$                              ;IF YES, GO DO TEST
21036 126066  000137  140340                    JMP     $EOP                            ;OTHERWISE, GOTO EOP
21037                                   ;
21038                                   ; INITIALISE VECTORS FOR Q22BE'S
21039                                   ;
21040 126072  012777  126206  054576    1$:     MOV     #4$,@VQBE1                      ;VECTOR FOR 1ST ONE
21041 126100  012777  000340  054572            MOV     #340,@VQPR1                     ;AT PRIORITY 7
21042 126106  012777  126216  054602            MOV     #5$,@VQBE2                      ;VECTOR FOR 2ND ONE
21043 126114  012777  000340  054576            MOV     #340,@VQPR2                     ;AT PRIORITY 7
21044 126122  012700  003004                    MOV     #Q22EN+2,R0                     ;POINTER FOR Q22BE BR'S
21045 126126  012702  000240                    MOV     #240,R2                         ;CPU AT 5
21046 126132  000402                            BR      3$                              ;START DOING ARBITRATION
21047                                   ;
21048                                   ; DO FOR CPU PRIORITIES 5-3
21049                                   ;
21050 126134  162702  000040            2$:     SUB     #40,R2                          ;LOWER CPU PRIORITY
21051 126140  106427  000340            3$:     MTPS    #340                            ;CPU AT 7
21052 126144  013703  002704                    MOV     CSR12,R3                        ;Q22BE 2 AT HIGHER BR
21053 126150  004737  137350                    JSR     PC,Q22INT                       ;INITIALISE TO INTERRUPTS
21054 126154  011077  054506                    MOV     (R0),@CSR2                      ;START 1ST ONE
21055 126160  013703  002664                    MOV     CSR1,R3                         ;Q22BE 1 AT LOWER BR
21056 126164  005740                            TST     -(R0)                           ;HIGHER PRIORITY
21057 126166  004737  137350                    JSR     PC,Q22INT                       ;INITIALISE
21058 126172  012077  054510                    MOV     (R0)+,@CSR22                    ;START 2ND ONE
21059 126176  106402                            MTPS    R2                              ;LOWER CPU PRIORITY
21060 126200  000240                            NOP                                     ;WAIT A WHILE
21061 126202  000240                            NOP
21062 126204  000240                            NOP
21063 126206  104126            4$:     ERROR   +126                                    ;INTERRUPTS IN WRONG ORDER
21064 126210  005726                            TST     (SP)+                           ;RESTORE STACK
21065 126212  005726                            TST     (SP)+
21066 126214  000402                            BR      6$
21067 126216  005726            5$:     TST     (SP)+                           ;RESTORE STACK
21068 126220  005726                            TST     (SP)+
21069 126222  022702  000140            6$:     CMP     #140,R2                         ;PRIORITY 3 LAST?
```

K15

SEQ 0399

TEST ARBITRATION BETWEEN DIFFERENT LEVELS OF INTERRUPTS

```
21070 126226  001342                          BNE     2$                      ;IF NOT, BRANCH TO CONTINUE
21071
```

```
21073                                   .SBTTL  TEST     PMG COUNTER
21074                                   ;USING 2 Q22BE THIS TEST WILL VALIDATE THAT PMG COUNTER IS REALLY
21075                                   ;CAPABLE OF GRANTING CPU BUS MASTERSHIP WHEN DMA REQUESTS ARE STILL
21076                                   ;PENDING.
21077                                   ;ROUTINE TEST
21078                                   ;.      IF UFD OR NO 2ND Q22BE THEN
21079                                   ;.              EXIT TEST
21080                                   ;.      ENDIF
21081                                   ;.      SET PMG COUNT TO SOME VALUE
21082                                   ;.      PROGRAM BOTH Q22BE TO INTERRUPT AT THE SAME LEVEL
21083                                   ;.      DETERMINE WHICH HAS HIGHER PRIORITY ON THE BUS
21084                                   ;.      PROGRAM Q22BE WITH HIGHER PRIORITY TO DO HOG MODE
21085                                   ;.      PROGRAM 2ND ONE TO DO A CYCLE
21086                                   ;       CACHE INSTRUCTION THAT WILL STOP 2ND DMA
21087                                   ;.      INITIATE BOTH DMA CYCLES
21088                                   ;.      STOP 2ND DMA (RESET IS  STOLEN" CPU BUS CYCLE)
21089                                   ;.      IF 2ND DMA HAPPENED THEN
21090                                   ;.              ERROR
21091                                   ;.      ENDIF
21092                                   ;.      CLEAR PMG COUNT
21093                                   ;.      PROGRAM Q22BE WITH HIGHER PRIORITY TO DO HOG MODE
21094                                   ;.      PROGRAM 2ND ONE TO DO A CYCLE
21095                                   ;.      CACHE INSTRUCTION THAT WILL STOP 2ND DMA
21096                                   ;.      INITIATE BOTH DMA CYCLES
21097                                   ;.      STOP 2ND DMA (CLEARING OF CSR2<0> IS 'STOLEN' CPU BUS CYCLE)
21098                                   ;.      IF 2ND DMA DIDN'T HAPPENED THEN
21099                                   ;.              ERROR
21100                                   ;.      ENDIF
21101                                   ;ENDROUTINE
21102
21103                                   ;;*****************************************************************
      126230  000004                  TST66:  SCOPE
21104 126232  000240                          NOP
21105 126234  005737  003032                  TST     CCHPAS                  ;have done enough inclus've passes?
21106 126240  001003                          BNE     99$                     ; not yet
21107 126242  000240                          NOP                             ; debug a'd
21108 126244  000137  126750                  JMP     11$                     ; yes sk'p this
21109 126250  000240                  99$:    NOP
21110
21111 126252  032737  000200  000052        BIT     #BIT07,@#52                     ;UFD MODE?
21112 126260  001402                          BEQ     100$                            ;IF NOT, CONTINUE
21113 126262  000137  140340                  JMP     $EOP                            ;EXIT
21114 126266  005737  002704          100$:   TST     CSR12                           ;SECOND Q22BE AVAILABLE?
21115 126272  001002                          BNE     110$                            ;IF YES, GO DO TEST
21116 126274  000137  140340                  JMP     $EOP                            ;OTHERWISE, GOTO EOP
21117                                   ;
21118                                   ; DETERMINE WHICH Q22BE IS CLOSER TO CPU BY DOING DATO
21119                                   ; THE CSR1 OF THE ONE WITH HIGHER PRIORITY IS IN R4, THE SECOND IN R2
21120                                   ;
21121 126300  005077  054362          110$:   CLR     @CSR2                           ;CLEAR JUST IN CASE
21122 126304  005077  054376                  CLR     @CSR22
21123 126310  012777  002740  054352        MOV     #TEMP,@BA                       ;ADDRESS TO BE USED
21124 126316  012777  002740  054364        MOV     #TEMP,@BA2                      ;FOR BOTH OF THEM
21125 126324  012777  177777  054340        MOV     #177777,@WC                     ;DO FOR 1 WORD
21126 126332  012777  177777  054352        MOV     #177777,@WC2                    ;IN BOTH Q22BW'S
21127 126340  012777  012525  054326        MOV     #12525,@DATA                    ;DATA FOR 1ST
21128 126346  012777  052525  054340        MOV     #52525,@DATA2                   ;DATA FOR 2ND
```

M15


```
21129 126354 012777 001601 054302          MOV    #1601,@CSR1          ;1 DATO FOR 1ST
21130 126362 012777 001601 054314          MOV    #1601,@CSR12         ;AND 2ND
21131 126370 012777 000001 054304          MOV    #1,@SIMGOA           ;BOTH GO
21132 126376 105777 054262           1$:   TSTB   @CSR1                ;FIRST DONE?
21133 126402 100375                        BPL    1$                   ;IF NOT, WAIT
21134 126404 105777 054274           2$:   TSTB   @CSR12               ;SECOND DONE
21135 126410 100375                        BPL    2$                   ;WAIT FOR 2ND
21136 126412 022737 052525 002740          CMP    #52525,TEMP          ;SECOND FINISHED LAST?
21137 126420 001405                        BEQ    3$                   ;IF SO, BRANCH
21138 126422 013704 002704                 MOV    CSR12,R4             ;SECOND ONE AT HIGHER LEVEL
21139 126426 013702 002664                 MOV    CSR1,R2              ;FIRST AT LOWER
21140 126432 000404                        BR     4$                   ;DO PMG PART
21141 126434 013704 002664           3$:   MOV    CSR1,R4              ;FIRST ONE AT HIGHER LEVEL
21142 126440 013702 002704                 MOV    CSR12,R2             ;SECOND AT LOWER
21143                                      ;
21144                                      ; INITIATE DMA CYCLES TO WORK WITH PMG COUNTER
21145                                      ;
21146 126444 013737 177520 002730    4$:   MOV    BCSR,SAVBR           ;STORE BCSR REGISTER
21147 126452 012700 000001                 MOV    #BIT00,R0            ;FIRST VALUE FOR PMG 0.4msec
21148 126456 050037 177520           5$:   BIS    R0,BCSR              ;CHANGE PMG CONUTER
21149 126462 005737 126564                 TST    6$                   ;CACHE RESET INTRUCTION
21150 126466 005737 126566                 TST    6$+2                 ;AND THE FOLLOWING FEW
21151 126472 005737 126570                 TST    6$+4                 ;
21152 126476 005737 126572                 TST    6$+6                 ;
21153 126502 012714 001407                 MOV    #1407,(R4)           ;DATI IN HOG MODE FOR HIGHER ONE
21154 126506 005064 000002                 CLR    2(R4)                ;CLEAR CSR2
21155 126512 012764 002740 000004          MOV    #TEMP,4(R4)          ;ADDRESS TO START DATI
21156 126520 012764 000000 000006          MOV    #0,6(R4)             ;DO 128 DATI'S IN HOG MODE
21157 126526 012712 001407                 MOV    #1407,(R2)           ;DATO FOR LOWER LEVEL ONE
21158 126532 005062 000002                 CLR    2(R2)                ;CLEAR JUST IN CASE
21159 126536 012762 002740 000004          MOV    #TEMP,4(R2)          ;USE THE SAME ADDRESS
21160 126544 012762 177777 000006          MOV    #177777,6(R2)        ;DO JUST ONE DATO
21161 126552 005062 000010                 CLR    10(R2)               ;CLEAR DATA OF 2ND Q22BE
21162 126556 012762 000001 000016          MOV    #1,16(R2)            ;BOTH GO
21163 126564 000005                  6$:   RESET                       ;STOP Q22BE AT R4
21164 126566 023762 002740 000010          CMP    TEMP,10(R2)          ;SECOND DMA DONE?
21165 126574 001001                        BNE    7$                   ;IF SET, BRANCH
21166 126576 104127                        ERROR  -127                 ;NO CYCLE STEALING
21167 126600 062700 000003           7$:   ADD    #3,R0                ;DO FOR 1,3,7 IN PMG
21168 126604 040037 177520                 BIC    R0,BCSR              ;CLEAR PREVOIUS BITS IN BCSR
21169 126610 022700 000007                 CMP    #7,R0                ;LAST ONE?
21170 126614 002320                        BGE    5$                   ;IF NOT, BRANCH
21171                                      ;
21172                                      ; TRY WITHOUT PMG COUNTER OPERATING
21173                                      ;
21174 126616 042737 000007 177520          BIC    #7,BCSR              ;TURN OF PMG COUNTER
21175 126624 005737 126726                 TST    8$                   ;CACHE RESET FETCH
21176 126630 005737 126730                 TST    8$+2                 ;AND THE FOLLOWING FEW
21177 126634 005737 126732                 TST    8$+4                 ;
21178 126640 005737 126734                 TST    8$+6                 ;
21179 126644 012714 001407                 MOV    #1407,(R4)           ;DATI IN HOG MODE FOR HIGHER ONE
21180 126650 005064 000002                 CLR    2(R4)                ;CLEAR CSR2
21181 126654 012764 002740 000004          MOV    #TEMP,4(R4)          ;ADDRESS TO START DATI
21182 126662 012764 000000 000006          MOV    #0,6(R4)             ;DO 128 DATI'S IN HOG MODE
21183 126670 012712 001407                 MOV    #1407,(R2)           ;DATO FOR LOWER LEVEL ONE
21184 126674 005062 000002                 CLR    2(R2)                ;CLEAR CRS2 OF 2ND
21185 126700 012762 002740 000004          MOV    #TEMP,4(R2)          ;USE THE SAME ADDRESS
```

N15

TEST - PMG COUNTER

```
21186 126706  012762  177777  000006          MOV     #177777,6(R2)                   ;DO JUST ONE DATO
21187 126714  005062  000010                  CLR     10(R2)                          ;CLEAR DATA OF 2ND Q22BE
21188 126720  012777  000001  053754          MOV     #1,@SIMGOA                      ;BOTH GO
21189 126726  000005                8$:       RESET                                   ;IF NOT WORKING, STOPS 2ND
21190 126730  023762  002740  000010          CMP     TEMP,10(R2)                     ;2ND DMA HAPPENED?
21191 126736  001401                          BEQ     9$                              ;IF YES, BRANCH
21192 126740  104127                          ERROR   #127                            ;IN PMG COUNTER
21193 126742  013737  002730  177520  9$      MOV     SAVBR,BCSR                      ;RESTORE BCSR
21194 126750  000240                11$:      NOP
21195 126752  000137  140340                  jmp     $eop
21196
21197 126756  123727  001220  000001  VIREOP: CMPB    $ENV,#1                         ; if not APT, don't worry about
21198 126764  001005                          BNE     1$                              ;
21199 126766  005737  003032                  TST     CCHPAS                          ; maintain cache routin pascnt
21200 126772  001402                          BEQ     1$
21201 126774  005337  003032                  DEC     CCHPAS
21202
21203                                         ; This VIREOP ROUTINE to provide common End of Pass exit point
21204 127000  000205                1$:       rts     r5
21205
```

```
21207                                           .SBTTL  GLOBAL ERROR MESSAGES
21208 127002  102  101  123  EM1:    .ASCIZ  /BASIC INSTRUCTION SET ERROR/
      127005  111  103  040
      127010  111  116  123
      127013  124  122  125
      127016  103  124  111
      127021  117  116  040
      127024  123  105  124
      127027  040  105  122
      127032  122  117  122
      127035  000
21209 127036  115  115  125  EM2:    .ASCIZ  /MMU ERROR/
      127041  040  105  122
      127044  122  117  122
      127047  000
21210 127050  106  120  120  EM3:    .ASCIZ  /FPP ERROR/
      127053  040  105  122
      127056  122  117  122
      127061  000
21211 127062  105  122  122  EM4:    .ASCIZ  /ERROR IN READ-WRITE BITS OF CCR/
      127065  117  122  040
      127070  111  116  040
      127073  122  105  101
      127076  104  055  127
      127101  122  111  124
      127104  105  040  102
      127107  111  124  123
      127112  040  117  106
      127115  040  103  103
      127120  122  000
21212 127122  106  117  122  EM5:    .ASCIZ  /FORCE MISS WRITES TO CACHE/
      127125  103  105  040
      127130  115  111  123
      127133  123  040  127
      127136  122  111  124
      127141  105  123  040
      127144  124  117  040
      127147  103  101  103
      127152  110  105  000
21213 127155  106  117  122  EM6:    .ASCIZ  /FORCE MISS WRITE INVALIDATES CACHE/
      127160  103  105  040
      127163  115  111  123
      127166  123  040  127
      127171  122  111  124
      127174  105  040  111
      127177  116  126  101
      127202  114  111  104
      127205  101  124  105
      127210  123  040  103
      127213  101  103  110
      127216  105  000
21214 127220  125  116  105  EM7:    .ASCIZ  /UNEXPECTED PARITY INTERRUPT/
      127223  130  120  105
      127226  103  124  105
      127231  104  040  120
      127234  101  122  111
      127237  124  131  040
```

GLOBAL ERROR MESSAGES

```
                127242  111     116     124
                127245  105     122     122
                127250  125     120     124
                127253  000
 21215          127254  124     101     107  EM10:   .ASCIZ  /TAG PARITY ERROR/
                127257  040     120     101
                127262  122     111     124
                127265  131     040     105
                127270  122     122     117
                127273  122     000
 21216          127275  104     101     124  EM11:   .ASCIZ  /DATA PARITY ERROR/
                127300  101     040     120
                127303  101     122     111
                127306  124     131     040
                127311  105     122     122
                127314  117     122     000
 21217          127317  114     117     127  EM12:   .ASCIZ  /LOW BYTE PARITY ERROR/
                127322  040     102     131
                127325  124     105     040
                127330  120     101     122
                127333  111     124     131
                127336  040     105     122
                127341  122     117     122
                127344  000
 21218          127345  110     111     107  EM13:   .ASCIZ  /HIGH BYTE PARITY ERROR/
                127350  110     040     102
                127353  131     124     105
                127356  040     120     101
                127361  122     111     124
                127364  131     040     105
                127367  122     122     117
                127372  122     000
 21219          127374  105     122     122  EM14:   .ASCIZ  /ERROR IN DATA PATH/
                127377  117     122     040
                127402  111     116     040
                127405  104     101     124
                127410  101     040     120
                127413  101     124     110
                127416  000
 21220          127417  106     117     122  EM15:   .ASCIZ  /FORCE MISS READS FROM CACHE/
                127422  103     105     040
                127425  115     111     123
                127430  123     040     122
                127433  105     101     104
                127436  123     040     106
                127441  122     117     115
                127444  040     103     101
                127447  103     110     105
                127452  000
 21221          127453  106     117     122  EM16:   .ASCIZ  /FORCE MISS READS FROM CACHE AND MISS/
                127456  103     105     040
                127461  115     111     123
                127464  123     040     122
                127467  105     101     104
                127472  123     040     106
                127475  122     117     115
                127500  040     103     101
```

GLOBAL ERROR MESSAGES

```
                127503  103     110     105
                127506  040     101     116
                127511  104     040     115
                127514  111     123     123
                127517  000
 21222          127520  105     122     122     EM17:   .ASCIZ  \ERROR IN RECORDING HITS IN HIT/MISS\
                127523  117     122     040
                127526  111     116     040
                127531  122     105     103
                127534  117     122     104
                127537  111     116     107
                127542  040     110     111
                127545  124     123     040
                127550  111     116     040
                127553  110     111     124
                127556  057     115     111
                127561  123     123     000
 21223          127564  127     122     111     EM20:   .ASCIZ  /WRITE BYTE ALLOCATES CACHE/
                127567  124     105     040
                127572  102     131     124
                127575  105     040     101
                127600  114     114     117
                127603  103     101     124
                127606  105     123     040
                127611  103     101     103
                127614  110     105     000
 21224          127617  127     122     111     EM21:   .ASCIZ  /WRITE BYTE HIT DOES NOT RECORD HIT/
                127622  124     105     040
                127625  102     131     124
                127630  105     040     110
                127633  111     124     040
                127636  104     117     105
                127641  123     040     116
                127644  117     124     040
                127647  122     105     103
                127652  117     122     104
                127655  040     110     111
                127660  124     000
 21225          127662  102     131     124     EM22:   .ASCIZ  /BYTES REVERSED ON WRITE CYCLES/
                127665  105     123     040
                 27670  122     105     126
                127673  105     122     123
                127676  105     104     040
                127701  117     116     040
                127704  127     122     111
                127707  124     105     040
                127712  103     131     103
                127715  114     105     123
                127720  000
 21226          127721  103     117     116     EM23:   .ASCIZ  /CONDITIONAL BYPASS DOESN'T INVALIDATE CACHE/
                127724  104     111     124
                127727  111     117     116
                127732  101     114     040
                127735  102     131     120
                127740  101     123     123
                127743  040     104     117
                127746  105     123     116
```

GLOBAL ERROR MESSAGES

```
                127751  047     124     040
                127754  111     116     126
                127757  101     114     111
                127762  104     101     124
                127765  105     040     103
                127770  101     103     110
                127773  105     000
 21227          127775  110     111     124     EM24:   .ASCIZ  /HITS RECORDED AFTER FLUSHING CACHE/
                130000  123     040     122
                130003  105     103     117
                130006  122     104     105
                130011  104     040     101
                130014  106     124     105
                130017  122     040     106
                130022  114     125     123
                130025  110     111     116
                130030  107     040     103
                130033  101     103     110
                130036  105     000
 21228          130040  102     131     120     EM25:   .ASCIZ  /BYPASS DOESN'T INVALIDATE CACHE/
                130043  101     123     123
                130046  040     104     117
                130051  105     123     116
                130054  047     124     040
                130057  111     116     126
                130062  101     114     111
                130065  104     101     124
                130070  105     040     103
                130073  101     103     110
                130076  105     000
 21229          130100  115     123     105     EM26:   .ASCIZ  /MSER DOES NOT CLEAR ON WRITE REFERENCE/
                130103  122     040     104
                130106  117     105     123
                130111  040     116     117
                130114  124     040     103
                130117  114     105     101
                130122  122     040     117
                130125  116     040     127
                130130  122     111     124
                130133  105     040     122
                130136  105     106     105
                130141  122     105     116
                130144  103     105     000
 21230          130147  120     101     122     EM27:   .ASCIZ  /PARITY ERROR DON'T CAUSE A MISS/
                130152  111     124     131
                130155  040     105     122
                130160  122     117     122
                130163  040     104     117
                130166  116     047     124
                130171  040     103     101
                130174  125     123     105
                130177  040     101     040
                130202  115     111     123
                130205  123     000
 21231          130207  120     101     122     EM30:   .ASCIZ  /PARITY ERROR DON'T SET MSER WITH CCR<7>=0/
                130212  111     124     131
                130215  040     105     122
```

F 16

SEQ 040?

GLOBAL ERROR MESSAGES

```
                130220  122     117     122
                130223  040     104     117
                130226  116     047     124
                130231  040     123     105
                130234  124     040     115
                130237  123     105     122
                130242  040     127     111
                130245  124     110     040
                130250  103     103     122
                130253  074     067     076
                130256  075     060     000
21232           130261  120     101     122     EM31:   .ASCIZ  /PARITY ERROR IGNORED/
                130264  111     124     131
                130267  040     105     122
                130272  122     117     122
                130275  040     111     107
                130300  116     117     122
                130303  105     104     000
21233           130306  120     101     122     EM32:   .ASCIZ  /PARITY ERROR IGNORED ON LOW BYTE/
                130311  111     124     131
                130314  040     105     122
                130317  122     117     122
                130322  040     111     107
                130325  116     117     122
                130330  105     104     040
                130333  117     116     040
                130336  114     117     127
                130341  040     102     131
                130344  124     105     000
21234           130347  120     101     122     EM33:   .ASCIZ  /PARITY ERROR IGNORED ON HIGH BYTE/
                130352  111     124     131
                130355  040     105     122
                130360  122     117     122
                130363  040     111     107
                130366  116     117     122
                130371  105     104     040
                130374  117     116     040
                130377  110     111     107
                130402  110     040     102
                130405  131     124     105
                130410  000
21235           130411  120     101     122     EM34:   .ASCIZ  /PARITY ABORT LOGIC DOESN'T WORK/
                130414  111     124     131
                130417  040     101     102
                130422  117     122     124
                130425  040     114     117
                130430  107     111     103
                130433  040     104     117
                130436  105     123     116
                130441  047     124     040
                130444  127     117     122
                130447  113     000
21236           130451  115     123     105     EM35:   .ASCIZ  /MSER NOT SET PROPERLY/
                130454  122     040     116
                130457  117     124     040
                130462  123     105     124
                130465  040     120     122
```

GLOBAL ERROR MESSAGES

```
                130470  117  120  105
                130473  122  114  131
                130476  000
21237           130477  120  101  122  EM36:   .ASCIZ  /PARITY INTERRUPT LOGIC DOESN'T WORK/
                130502  111  124  131
                130505  040  111  116
                130510  124  105  122
                130513  122  125  120
                130516  124  040  114
                130521  117  107  111
                130524  103  040  104
                130527  117  105  123
                130532  116  047  124
                130535  040  127  117
                130540  122  113  000
21238           130543  116  130  115  EM37:   .ASCIZ  /NXM AND PARITY ABORT DIN'T HAPPEN/
                130546  040  101  116
                130551  104  040  120
                130554  101  122  111
                130557  124  131  040
                130562  101  102  117
                130565  122  124  040
                130570  104  111  116
                130573  047  124  040
                130576  110  101  120
                130601  120  105  116
                130604  000
21239           130605  120  101  122  EM40:   .ASCIZ  /PARITY ABORT NOT BLOCKED BY NXM TRAP/
                130610  111  124  131
                130613  040  101  102
                130616  117  122  124
                130621  040  116  117
                130624  124  040  102
                130627  114  117  103
                130632  113  105  104
                130635  040  102  131
                130640  040  116  130
                130643  115  040  124
                130646  122  101  120
                130651  000
21240           130652  115  125  114  EM41:   .ASCIZ  /MULTI-PROCESSOR HOOK INSTRUCTION DOESN'T CAUSE MISS/
                130655  124  111  055
                130660  120  122  117
                130663  103  105  123
                130666  123  117  122
                130671  040  110  117
                130674  117  113  040
                130677  111  116  123
                130702  124  122  125
                130705  103  124  111
                130710  117  116  040
                130713  104  117  105
                130716  123  116  047
                130721  124  040  103
                130724  101  125  123
                130727  105  040  115
                130732  111  123  123
```

H16

GLOBAL ERROR MESSAGES

```
                130735      000
 21241          130736      105     122     122  EM42:   .ASCIZ  /ERROR IN PARITY LOGIC/
                130741      117     122     040
                130744      111     116     040
                130747      120     101     122
                130752      111     124     131
                130755      040     114     117
                130760      107     111     103
                130763      000
 21242          130764      105     122     122  EM43:   .ASCIZ  /ERROR IN CACHE DATA RAMS/
                130767      117     122     040
                130772      111     116     040
                130775      103     101     103
                131000      110     105     040
                131003      104     101     124
                131006      101     040     122
                131011      101     115     123
                131014      000
 21243          131015      105     122     122  EM44:   .ASCIZ  /ERROR IN NXM IN STANDALONE MODE/
                131020      117     122     040
                131023      111     116     040
                131026      116     130     115
                131031      040     111     116
                131034      040     123     124
                131037      101     116     104
                131042      101     114     117
                131045      116     105     040
                131050      115     117     104
                131053      105     000
 21244          131055      105     122     122  EM45:   .ASCIZ  \ERROR IN RECORDING HITS THROUGH HIT/MISS REGISTER\
                131060      117     122     040
                131063      111     116     040
                131066      122     105     103
                131071      117     122     104
                131074      111     116     107
                131077      040     110     111
                131102      124     123     040
                131105      124     110     122
                131110      117     125     107
                131113      110     040     110
                131116      111     124     057
                131121      115     111     123
                131124      123     040     122
                131127      105     107     111
                131132      123     124     105
                131135      122     000
 21245          131137      110     111     124  EM46:   .ASCIZ  /HIT RECORDED FOR A LOCATION THAT SHOULD NOT BE IN CACHE/
                131142      040     122     105
                131145      103     117     122
                131150      104     105     104
                131153      040     106     117
                131156      122     040     101
                131161      040     114     117
                131164      103     101     124
                131167      111     117     116
                131172      040     124     110
                131175      101     124     040
```

GLOBAL ERROR MESSAGES

```
                131200  123  110  117
                131203  125  114  104
                131206  040  116  117
                131211  124  040  102
                131214  105  040  111
                131217  116  040  103
                131222  101  103  110
                131225  105  000
 21246          131227  115  111  123  EM47:   .ASCIZ  /MISS RECORDED FOR A LOCATION THAT SHOULD BE IN CACHE/
                131232  123  040  122
                131235  105  103  117
                131240  122  104  105
                131243  104  040  106
                131?46  117  122  040
                131251  101  040  114
                131254  117  103  101
                131257  124  111  117
                131262  116  040  124
                131265  110  101  124
                131270  040  123  110
                131273  117  125  114
                131276  104  040  102
                131301  105  040  111
                131304  116  040  103
                131307  101  103  110
                131312  105  000
 21247          131314  105  122  122  EM50:   .ASCIZ  /ERROF IN TAG STORE/
                131317  117  122  040
                131322  111  116  040
                131325  124  101  107
                131330  040  123  124
                131333  117  122  105
                131336  000
 21248          131337  105  122  122  EM51:   .ASCIZ  /ERROR PCR READ-WRITF BITS/
                131342  117  122  040
                131345  120  103  122
                131350  040  122  105
                131353  101  104  055
                131356  127  122  111
                131361  124  105  040
                131364  102  111  124
                131367  123  000
 21249          131371  105  122  122  EM52:   .ASCIZ  /ERROR IN BCSR READ-WRITE BITS/
                131374  117  122  040
                131377  111  116  040
                131402  102  103  123
                131405  122  040  122
                131410  105  101  104
                131413  055  127  122
                131416  111  124  105
                131421  040  102  111
                131-24  124  123  000
 21250          131427  122  105  123  EM53:   .ASCIZ  /RESET DOESN'T CLEAR BCSR<4>/
                131432  105  124  040
                131435  104  117  105
                131440  123  116  047
                131443  124  040  103
```

J16

GLOBAL ERROR MESSAGES

```
                131446      114     105     101
                131451      122     040     102
                131454      103     123     122
                131457      074     064     076
                131462      000
21251           131463      103     110     105     EM54:   .ASCIZ  /CHECKSUM ERROR IN 16 BIT ROM /
                131466      103     113     123
                131471      125     115     040
                131474      105     122     122
                131477      117     122     040
                131502      111     116     040
                131505      061     066     055
                131510      102     111     124
                131513      040     122     117
                131516      115     040     000
21252           131521      103     110     105     EM55:   .ASCIZ  /CHECKSUM ERROR IN 8 BIT ROM/
                131524      103     113     123
                131527      125     115     040
                131532      105     122     122
                131535      117     122     040
                131540      111     116     040
                131543      070     055     102
                131546      111     124     040
                131551      122     117     115
                131554      000
21253           131555      124     111     115     EM56:   .ASCIZ  /TIMEOUT READING LKS/
                131560      105     117     125
                131563      124     040     122
                131566      105     101     104
                131571      111     116     107
                131574      040     114     113
                131577      123     000
21254           131601      114     113     123     EM57:   .ASCIZ  /LKS<07> DOES NOT BECOME 1/
                131604      074     060     067
                131607      076     040     104
                131612      117     105     123
                131615      040     116     117
                131620      124     040     102
                131623      105     103     117
                131626      115     105     040
                131631      061     000
21255           131633      127     122     111     EM60:   .ASCIZ  /WRITE REFERENCE DOESN'T CLEAR LKS<07>/
                131636      124     105     040
                131641      122     105     106
                131644      105     122     105
                131647      116     103     105
                131652      040     104     117
                131655      105     123     116
                131660      047     124     040
                131663      103     114     105
                131666      101     122     040
                131671      114     113     123
                131674      074     060     067
                131677      076     000
21256           131701      111     114     114     EM61:   .ASCIZ  /ILLEGAL LKS INTERRUPTS/
                131704      105     107     101
                131707      114     040     114
```

GLOBAL ERROR MESSAGES

```
                131712  113  123  040
                131715  111  116  124
                131720  105  122  122
                131723  125  120  124
                131726  123  000
21257           131730  120  122  117  EM62:   .ASCIZ  /PROCESSOR INTERRUPTS DON'T CLEAR LKS<07>/
                131733  103  105  123
                131736  123  117  122
                131741  040  111  116
                131744  124  105  122
                131747  122  125  120
                131752  124  123  040
                131755  104  117  116
                131760  047  124  040
                131763  103  114  105
                131766  101  122  040
                131771  114  113  123
                131774  074  060  067
                131777  076  000
21258           132001  114  113  123  EM63:   .ASCIZ  /LKS READY DOESN'T GO LOW/
                132004  040  122  105
                132007  101  104  131
                132012  040  104  117
                132015  105  123  116
                132020  047  124  040
                132023  107  117  040
                132026  114  117  127
                132031  000
21259           132032  127  122  117  EM64:   .ASCIZ  /WRONG NUMBER OF LKS INTERRUPTS/
                132035  116  107  040
                132040  116  125  115
                132043  102  105  122
                132046  040  117  106
                132051  040  114  113
                132054  123  040  111
                132057  116  124  105
                132062  122  122  125
                132065  120  124  123
                132070  000
21260           132071  114  113  123  EM65:   .ASCIZ  /LKS INTERRUPTS HAPPEN AT WRONG PRIORITY/
                132074  040  111  116
                132077  124  105  122
                132102  122  125  120
                132105  124  123  040
                132110  110  101  120
                132113  120  105  116
                132116  040  101  124
                132121  040  127  122
                132124  117  116  107
                132127  040  120  122
                132132  111  117  122
                132135  111  124  131
                132140  000
21261           132141  102  103  123  EM66:   .ASCIZ  /BCSR<12> DOES NOT DISABLES LKS/
                132144  122  074  061
                132147  062  076  040
                132152  104  117  105
```

GLOBAL ERROR MESSAGES

```
                132155  123  040  116
                132160  117  124  040
                132163  104  111  123
                132166  101  102  114
                132171  105  123  040
                132174  114  113  123
                132177  000
 21262          132200  102  103  123  EM67:   .ASCIZ  /BCSR<13> DOESN'T SET LKS<06>/
                132203  122  074  061
                132206  063  076  040
                132211  104  117  105
                132214  123  116  047
                132217  124  040  123
                132222  105  124  040
                132225  114  113  123
                132230  074  060  066
                132233  076  000
 21263          132235  122  105  123  EM70:   .ASCIZ  /RESET DOESN'T SET LKS<7>/
                132240  105  124  040
                132243  104  117  105
                132246  123  116  047
                132251  124  040  123
                132254  105  124  040
                132257  114  113  123
                132262  074  067  076
                132265  000
 21264          132266  122  105  123  EM71:   .ASCIZ  /RESET DOESN'T CLEAR LKS<06>/
                132271  105  124  040
                132274  104  117  105
                132277  123  116  047
                132302  124  040  103
                132305  114  105  101
                132310  122  040  114
                132313  113  123  074
                132316  060  066  076
                132321  000
 21265          132322  124  111  115  EM72:   .ASCIZ  /TIMEOUT READING SLU REGISTERS/
                132325  105  117  125
                132330  124  040  122
                132333  105  101  104
                132336  111  116  107
                132341  040  123  114
                132344  125  040  122
                132347  105  107  111
                132352  123  124  105
                132355  122  123  000
 21266          132360  105  122  122  EM73:   .ASCIZ  /ERROR IN XMIT READY/
                132363  117  122  040
                132366  111  116  040
                132371  130  115  111
                132374  124  040  122
                132377  105  101  104
                132402  131  000
 21267          132404  122  103  123  EM74:   .ASCIZ  /RCSR<7> DOESN'T BECOME 1/
                132407  122  074  067
                132412  076  040  104
                132415  117  105  123
```

GLOBAL ERROR MESSAGES

```
        132420  116  047  124
        132423  040  102  105
        132426  103  117  115
        132431  105  040  061
        132434  000
21268   132435  127  122  117  EM75:   .ASCIZ  /WRONG CHARACTER RECEIVED/
        132440  116  107  040
        132443  103  110  101
        132446  122  101  103
        132451  124  105  122
        132454  040  122  105
        132457  103  105  111
        132462  126  105  104
        132465  000
21269   132466  122  103  123  EM76:   .ASCIZ  /RCSR<07> NOT CLEARED AFTER READING RBUF/
        132471  122  074  060
        132474  067  076  040
        132477  116  117  124
        132502  040  103  114
        132505  105  101  122
        132510  105  104  040
        132513  101  106  124
        132516  105  122  040
        132521  122  105  101
        132524  104  111  116
        132527  107  040  122
        132532  102  125  106
        132535  000
21270   132536  130  103  123  EM77:   .ASCIZ  /XCSR<07> NOT SET ON RESET/
        132541  122  074  060
        132544  067  076  040
        132547  116  117  124
        132552  040  123  105
        132555  124  040  117
        132560  116  040  122
        132563  105  123  105
        132566  124  000
21271   132570  122  103  123  EM100:  .ASCIZ  /RCSR<07> NOT CLEARED ON RESET/
        132573  122  074  060
        132576  067  076  040
        132601  116  117  124
        132604  040  103  114
        132607  105  101  122
        132612  105  104  040
        132615  117  116  040
        132620  122  105  123
        132623  105  124  000
21272   132626  123  114  125  EM101:  .ASCIZ  /SLU INTERRUPTS HAPPEN AT 4/
        132631  040  111  116
        132634  124  105  122
        132637  122  125  120
        132642  124  123  040
        132645  110  101  120
        132650  120  105  116
        132653  040  101  124
        132656  040  064  000
21273   132661  122  105  123  EM102:  .ASCIZ  /RESET DOES NOT CLEAR PROPER BITS IN SLU REGISTERS/
```

GLOBAL ERROR MESSAGES

```
                132664      105     124     040
                132667      104     117     105
                132672      123     040     116
                132675      117     124     040
                132700      103     114     105
                132703      101     122     040
                132706      120     122     117
                132711      120     105     122
                132714      040     102     111
                132717      124     123     040
                132722      111     116     040
                132725      123     114     125
                132730      040     122     105
                132733      107     111     123
                132736      124     105     122
                132741      123     000
21274           132743      124     122     101   EM103:  .ASCIZ  /TRANSMIT INTERRUPT DOES NOT CLEAR XCSR<07>/
                132746      116     123     115
                132751      111     124     040
                132754      111     116     124
                132757      105     122     122
                132762      125     120     124
                132765      040     104     117
                132770      105     123     040
                132773      116     117     124
                132776      040     103     114
                133001      105     101     122
                133004      040     130     103
                133007      123     122     074
                133012      060     067     076
                133015      000
21275           133016      122     105     103   EM104:  .ASCIZ  /RECEIVE INTERRUPTS DON'T CLEAR RCSR<07>/
                133021      105     111     126
                133024      105     040     111
                133027      116     124     105
                133032      122     122     125
                133035      120     124     123
                133040      040     104     117
                133043      116     047     124
                133046      040     103     114
                133051      105     101     122
                133054      040     122     103
                133057      123     122     074
                133062      060     067     076
                133065      000
21276           133066      102     122     105   EM105:  .ASCIZ  /BREAK CONDITION DOES NOT SET RBUF PROPERLY/
                133071      101     113     040
                133074      103     117     116
                133077      104     111     124
                133102      111     117     116
                133105      040     104     117
                133110      105     123     040
                133113      116     117     124
                133116      040     123     105
                133121      124     040     122
                133124      102     125     106
                133127      040     120     122
```

GLOBAL ERROR MESSAGES

```
                133132  117     120     105
                133135  122     114     131
                133140  000
 21277          133141  122     102     125     EM106:  .ASCIZ  /RBUF <15 11> WASN'T CLEARED ON NEXT CHARACTER/
                133144  106     040     074
                133147  061     065     055
                133152  061     061     076
                133155  040     127     101
                133160  123     116     047
                133163  124     040     103
                133166  114     105     101
                133171  122     105     104
                133174  040     117     116
                133177  040     116     105
                133202  130     124     040
                133205  103     110     101
                133210  122     101     103
                133213  124     105     122
                133216  000
 21278          133217  123     114     125     EM107:  .ASCIZ  /SLU INTERRUPTS DON'T HAPPEN/
                133222  040     111     116
                133225  124     105     122
                133230  122     125     120
                133233  124     123     040
                133236  104     117     116
                133241  047     124     040
                133244  110     101     120
                133247  120     105     116
                133252  000
 21279          133253  105     122     122     EM110:  .ASCIZ  /ERROR IN WRITING TO SLU REGISTERS/
                133256  117     122     040
                133261  111     116     040
                133264  127     122     111
                133267  124     111     116
                133272  107     040     124
                133275  117     040     123
                133300  114     125     040
                133303  122     105     107
                133306  111     123     124
                133311  105     122     123
                133314  000
 21280          133315  106     111     122     EM111:  .ASCIZ  /FIRST CHARACTER WAS NOT OVERRUN BY THE SECOND/
                133320  123     124     040
                133323  103     110     101
                133326  122     101     103
                133331  124     105     122
                133334  040     127     101
                133337  123     040     116
                133342  117     124     040
                133345  117     126     105
                133350  122     122     125
                133353  116     040     102
                133356  131     040     124
                133361  110     105     040
                133364  123     105     103
                133367  117     116     104
                133372  000
```

GLOBAL ERROR MESSAGES

```
 21281 133373     117     126     105  EM112:  .ASCIZ  /OVERRUN CONDITION DOES NOT SET PROPER BITS IN RBUF/
       133376     122     122     125
       133401     116     040     103
       133404     117     116     104
       133407     111     124     111
       133412     117     116     040
       133415     104     117     105
       133420     123     040     116
       133423     117     124     040
       133426     123     105     124
       133431     040     120     122
       133434     117     120     105
       133437     122     040     102
       133442     111     124     123
       133445     040     111     116
       133450     040     122     102
       133453     125     106     000
 21282 133456     117     126     105  EM113:  .ASCIZ  /OVERRUN BITS WERE NOT CLEARED ON THE NEXT CHARACTER/
       133461     122     122     125
       133464     116     040     102
       133467     111     124     123
       133472     040     127     105
       133475     122     105     040
       133500     116     117     124
       133503     040     103     114
       133506     105     101     122
       133511     105     104     040
       133514     117     116     040
       133517     124     110     105
       133522     040     116     105
       133525     130     124     040
       133530     103     110     101
       133533     122     101     103
       133536     124     105     122
       133541     000
 21283 133542     105     122     122  EM114:  .ASCIZ  \ERROR ON XCSR<2>\
       133545     117     122     040
       133550     117     116     040
       133553     130     103     123
       133556     122     074     062
       133561     076     000
 21284 133563     105     122     122  EM115:  .ASCIZ  /ERROR IN TAG STORE FROM STANDALONE MODE/
       133566     117     122     040
       133571     111     116     040
       133574     124     101     107
       133577     040     123     124
       133602     117     122     105
       133605     040     106     122
       133610     117     115     040
       133613     123     124     101
       133616     116     104     101
       133621     114     117     116
       133624     105     040     115
       133627     117     104     105
       133632     000
 21285 133633     104     115     101  EM116:  .ASCIZ  /DMA TAG PARITY DOES NOT GENERATE PROPER RESPONSE/
       133636     040     124     101
```

```
                133641      107     040     120
                133644      101     122     111
                133647      124     131     040
                133652      104     117     105
                133655      123     040     116
                133660      117     124     040
                133663      107     105     116
                133666      105     122     101
                133671      124     105     040
                133674      120     122     117
                133677      120     105     122
                133702      040     122     105
                133705      123     120     117
                133710      116     123     105
                133713      000
 21286          133714      115     123     105  EM117:  .ASCIZ  /MSER<13>NOT SET IN STANDALONE MODE/
                133717      122     074     061
                133722      063     076     116
                133725      117     124     040
                133730      123     105     124
                133733      040     111     116
                133736      040     123     124
                133741      101     116     104
                133744      101     114     117
                133747      116     105     040
                133752      115     117     104
                133755      105     000
 21287          133757      104     115     101  EM120:  .ASCIZ  /DMA WRITE HITS DON'T INVALIDATE CACHE/
                133762      040     127     122
                133765      111     124     105
                133770      040     110     111
                133773      124     123     040
                133776      104     117     116
                134001      047     124     040
                134004      111     116     126
                134007      101     114     111
                134012      104     101     124
                134015      105     040     103
                134020      101     103     110
                134023      105     000
 21288          134025      111     116     040  EM121:  .ASCIZ  /IN BLOCK MODE ON WRITE DMA HITS NOT EVERYTHING IS INVALIDATED/
                134030      102     114     117
                134033      103     113     040
                134036      115     117     104
                134041      105     040     117
                134044      116     040     127
                134047      122     111     124
                134052      105     040     104
                134055      115     101     040
                134060      110     111     124
                134063      123     040     116
                134066      117     124     040
                134071      105     126     105
                134074      122     131     124
                134077      110     111     116
                134102      107     040     111
                134105      123     040     111
```

F1

GLOBAL ERROR MESSAGES

```
                134110      116      126      101
                134113      114      111      104
                134116      101      124      105
                134121      104      000
 21289          134123      122      105      101  EM122:  .ASCIZ  /READ DMA HIT IS WRONG/
                134126      104      040      104
                134131      115      101      040
                134134      110      111      124
                134137      040      111      123
                134142      040      127      122
                134145      117      116      107
                134150      000
 21290          134151      105      122      122  EM123:  .ASCIZ  /ERROR IN Q22BE DMA CYCLES/
                134154      117      122      040
                134157      111      116      040
                134162      121      062      062
                134165      102      105      040
                134170      104      115      101
                134173      040      103      131
                134176      103      114      105
                134201      123      000
 21291          134202      120      111      122  EM124:  .ASCIZ  /PIRQ INTERRUPTS DON'T TAKE PRIORITY OVER Q BUS INTERRUPTS/
                134206      121      040      111
                134211      116      124      105
                134214      122      122      125
                134217      120      124      123
                134222      040      104      117
                134225      116      047      124
                134230      040      124      101
                134233      113      105      040
                134236      120      122      111
                134241      117      122      111
                134244      124      131      040
                134247      117      126      105
                134252      122      040      121
                134255      040      102      125
                134260      123      040      111
                134263      116      124      105
                134266      122      122      125
                134271      120      124      123
                134274      000
 21292          134275      116      117      040  EM125:  .ASCIZ  /NO POWER DOWN TRAP TO 24 OCCUR/
                134300      120      117      127
                134303      105      122      040
                134306      104      117      127
                134311      116      040      124
                134314      122      101      120
                134317      040      124      117
                134322      040      062      064
                134325      040      117      103
                134330      103      125      122
                134333      000
 21293          134334      105      122      122  EM126:  .ASCIZ  /ERROR DOING Q22BE INTERRUPTS/
                134337      117      122      040
                134342      104      117      111
                134345      116      107      040
                134350      121      062      062
```

GLOBAL ERROR MESSAGES

```
                134353  102  105  040
                134356  111  116  124
                134361  105  122  122
                134364  125  120  124
                134367  123  000
 21294          134371  105  122  122  EM127:  .ASCIZ  /ERROR IN OPERATION OF PMG COUNTER/
                134374  117  122  040
                134377  111  116  040
                134402  117  120  105
                134405  122  101  124
                134410  111  117  116
                134413  040  117  106
                134416  040  120  115
                134421  107  040  103
                134424  117  125  116
                134427  124  105  122
                134432  000
 21295          134433  125  116  105  EM130:  .ASCIZ  /UNEXPECTED TRAP TO 4/
                134436  130  120  105
                134441  103  124  105
                134444  104  040  124
                134447  122  101  120
                134452  040  124  117
                134455  040  064  000
 21296          134460  105  122  122  EM131:  .ASCIZ  /ERROR WRITING TO LKS<6>/
                134463  117  122  040
                134466  127  122  111
                134471  124  111  116
                134474  107  040  124
                134477  117  040  114
                134502  113  123  074
                134505  066  076  000
 21297          134510  105  122  122  EM132:  .ASCIZ  /ERROR IN MAINTENANCE REGISTER/
                134513  117  122  040
                134516  111  116  040
                134521  115  101  111
                134524  116  124  105
                134527  116  101  116
                134532  103  105  040
                134535  122  105  107
                134540  111  123  124
                134543  105  122  000
 21298          134546  105  105  122  EM133:  .ASCIZ  /EEROM TYPE ERROR/
                134551  117  115  040
                134554  124  131  120
                134557  105  040  105
                134562  122  122  117
                134565  122  000
 21299          134567  122  105  101  EM134:  .ASCIZ  /READ OR WRITE EEROM ERROR/
                134572  104  040  117
                134575  122  040  127
                134600  122  111  124
                134603  105  040  105
                134606  105  122  117
                134611  115  040  105
                134614  122  122  117
                134617  122  000
```

GLOBAL ERROR MESSAGES

```
 21300
 21301 134621  040  124  105  DH1:    .ASCII  / TEST  ERROR/<15><12>
       134624  123  124  011
       134627  105  122  122
       134632  117  122  015
       134635  012
 21302 134636  040  040  043          .ASCIZ  /  #     PC/
       134641  011  040  120
       134644  103  000
 21303 134646  040  124  105  DH4:    .ASCII  / TEST  ERROR   EXPCTED RECEIVED/<15><12>
       134651  123  124  011
       134654  105  122  122
       134657  117  122  011
       134662  105  130  120
       134665  103  124  105
       134670  104  011  122
       134673  105  103  105
       134676  111  126  105
       134701  104  015  012
 21304 134704  040  040  043          .ASCIZ  /  #     PC      DATA     DATA/
       134707  011  040  120
       134712  103  011  040
       134715  104  101  124
       134720  101  040  040
       134723  040  040  040
       134726  104  101  124
       134731  101  000
 21305 134733  040  124  105  DH5:    .ASCII  / TEST  ERROR   HITMIS  DATA IN DATA IN/<15><12>
       134736  123  124  011
       134741  105  122  122
       134744  117  122  011
       134747  110  111  124
       134752  115  111  123
       134755  011  104  101
       134760  124  101  040
       134763  111  116  011
       134766  104  101  124
       134771  101  040  111
       134774  116  015  012
 21306 134777  040  040  043          .ASCIZ  /  #     PC      REG.   CACHE   MEMORY/
       135002  011  040  120
       135005  103  011  040
       135010  122  105  107
       135013  056  011  103
       135016  101  103  110
       135021  105  011  115
       135024  105  115  117
       135027  122  131  000
 21307 135032  040  124  105  DH7:    .ASCII  / TEST  ERROR   ADDRESS MSER/<15><12>
       135035  123  124  011
       135040  105  122  122
       135043  117  122  011
       135046  101  104  104
       135051  122  105  123
       135054  123  011  115
       135057  123  105  122
       135062  015  012
```

GLOBAL ERROR MESSAGES

```
 21308 135064  040  040  043          .ASCIZ  /  #     PC     ACCESSED/
       135067  011  040  120
       135072  103  011  101
       135075  103  103  105
       135100  123  123  105
       135103  104  000
 21309 135105  040  124  105  DH24:   .ASCII  / TEST  ERROR   NUMBER/<15><12>
       135110  123  124  011
       135113  105  122  122
       135116  117  122  011
       135121  116  125  115
       135124  102  105  122
       135127  015  012
 21310 135131  040  040  043          .ASCI7  /  #     PC     OF HITS/
       135134  011  040  120
       135137  103  011  117
       135142  106  040  110
       135145  111  124  123
       135150  000
 21311 135151  105  122  122  DH27:   .ASCII  \ERROR  ERROR   MSER    HIT/MISS\<15><12>
       135154  117  122  011
       135157  105  122  122
       135162  117  122  011
       135165  115  123  105
       135170  122  011  110
       135173  111  124  057
       135176  115  111  123
       135201  123  015  012
 21312 135204  040  040  043          .ASCIZ  /  #     PC/
       135207  011  040  120
       135212  103  000
 21313 135214  040  124  105  DH41:   .ASCII  / TEST  ERROR   INSTRUCTION/<15><12>
       135217  123  124  011
       135222  105  122  122
       135225  117  122  011
       135230  111  116  123
       135233  124  122  125
       135236  103  124  111
       135241  117  116  015
       135244  012
 21314 135245  040  040  043          .ASCIZ  /  #     PC      OPCODE/
       135250  011  040  120
       135253  103  011  040
       135256  117  120  103
       135261  117  104  105
       135264  000
 21315 135265  040  124  105  DH43:   .ASCII  / TEST  ERROR   EXPECTD RECEIVD CACHE/<15><12>
       135270  123  124  011
       135273  105  122  122
       135276  117  122  011
       135301  105  130  120
       135304  105  103  124
       135307  104  011  122
       135312  105  103  105
       135315  111  126  104
       135320  011  103  101
       135323  103  110  105
```

GLOBAL ERROR MESSAGES

```
                135326      015     012
 1316           135330      040     040     043             .ASCIZ  /  #     PC      DATA    DATA    LOCATION/
                135333      011     040     120
                135336      103     011     040
                135341      104     101     124
                135344      101     011     040
                135347      104     101     124
                135352      101     011     114
                135355      117     103     101
                135360      124     111     117
                135363      116     000
21317           135365      040     124     105     DH47:   .ASCII  / TEST  ERROR   ADDRESS ADDRESS/<15><12>
                135370      123     124     011
                135373      105     122     122
                135376      117     122     011
                135401      101     104     104
                135404      122     105     123
                135407      123     011     101
                135412      104     104     122
                135415      105     123     123
                135420      015     012
21318           135422      040     040     043             .ASCIZ  /  #     PC      <21-16>  <15-0>/
                135425      011     040     120
                135430      103     011     074
                135433      062     061     055
                135436      061     066     076
                135441      011     040     074
                135444      061     065     055
                135447      060     076     000
21319           135452      040     124     105     DH65:   .ASCII  / TEST  ERROR   PRIORITY/<15><12>
                135455      123     124     011
                135460      105     122     122
                135463      117     122     011
                135466      120     122     111
                135471      117     122     111
                135474      124     131     015
                135477      012
21320           135500      040     040     043             .ASCIZ  /  #     PC      LEVEL/
                135503      011     040     120
                135506      103     011     114
                135511      105     126     105
                135514      114     000
21321           135516      040     124     105     DH72:   .ASCII  / TEST  ERROR   ADDRESS/<15><12>
                135521      123     124     011
                135524      105     122     122
                135527      117     122     011
                135532      101     104     104
                135535      122     105     123
                135540      123     015     012
21322           135543      040     040     043             .ASCIZ  /  #     PC      FAILED/
                135546      011     040     120
                135551      103     011     106
                135554      101     111     114
                135557      105     104     000
21323           135562      040     124     105     DH105:  .ASCII  / TEST  ERROR   RBUF/<15><12>
                135565      123     124     011
                135570      105     122     122
```

GLOBAL ERROR MESSAGES

```
                135573  117  122  011
                135576  122  102  125
                135601  106  015  012
 21324          135604  040  040  043          .ASCIZ  /  #     PC/
                135607  011  040  120
                135612  103  000
 21325          135614  040  124  105  DH115:  .ASCII  / TEST  ERROR   MSER    ADDRESS/<15><12>
                135617  123  124  011
                135622  105  122  122
                135625  117  122  011
                135630  115  123  105
                135633  122  011  101
                135636  104  104  122
                135641  105  123  123
                135644  015  012
 21326          135646  040  040  043          .ASCIZ  /  #     PC             ACCESSED/
                135651  011  040  120
                135654  103  040  011
                135657  011  101  103
                135662  103  105  123
                135665  123  105  104
                135670  000
 21327          135671  040  124  105  DH134:  .ASCII  / TEST  ERROR    # of              DATA  PCR   ADDRESS/<15><12>
                135674  123  124  011
                135677  105  122  122
                135702  117  122  011
                135705  040  043  040
                135710  157  146  011
                135713  011  040  040
                135716  040  104  101
                135721  124  101  040
                135724  040  120  103
                135727  122  040  040
                135732  040  101  104
                135735  104  122  105
                135740  123  123  015
                135743  012
 21328          135744  040  040  043          .ASCIZ  /  #     PC     ERRORS  PATTERN    READ        ACCESSED/
                135747  011  040  120
                135752  103  040  011
                135755  105  122  122
                135760  117  122  123
                135763  040  040  120
                135766  101  124  124
                135771  105  122  116
                135774  040  040  040
                135777  040  122  105
                136002  101  104  040
                136005  040  040  040
                136010  040  040  040
                136013  040  101  103
                136016  103  105  123
                136021  123  105  104
                136024  000
 21329
 21330                               .EVEN
 21331          136026  001162  001116  000000  DT1:    .WORD   $TMP1,$ERRPC,0
```

GLOBAL ERROR MESSAGES

```
21332 136034  001162  001116  000001  DT4:    .WORD   $TMP1,$ERRPC,R1,CCR,0
      136042  177746  000000
21333 136046  001162  001116  000002  DT5:    .WORD   $TMP1,$ERRPC,R2,R1,$GDDAT,0
      136054  000001  001124  000000
21334 136062  001162  001116  001122  DT7:    .WORD   $TMP1,$ERRPC,$BDADR,MSER,0
      136070  177744  000000
21335 136074  001162  001116  001124  DT14:   .WORD   $TMP1,$ERRPC,$GDDAT,TSTLOC,0
      136102  003162  000000
21336 136106  001162  001116  001124  DT17:   .WORD   $TMP1,$ERRPC,$GDDAT,RECDAT,0
      136114  114150  000000
21337 136120  001162  001116  000003  DT24:   .WORD   $TMP1,$ERRPC,R3,0
      136126  000000
21338 136130  001162  001116  177744  DT27:   .WORD   $TMP1,$ERRPC,MSER,R3,0
      136136  000003  000000
21339 136142  001162  001116  001124  DT35:   .WORD   $TMP1,$ERRPC,$GDDAT,MSER,0
      136150  177744  000000
21340 136154  001162  001116  001126  DT41:   .WORD   $TMP1,$ERRPC,$BDDAT,0
      136162  000000
21341 136164  001162  001116  000001  DT43:   .WORD   $TMP1,$ERRPC,R1,RECDAT,$BDADR,0
      136172  114150  001122  000000
21342 136200  001162  001116  172354  DT47:   .WORD   $TMP1,$ERRPC,KIPAR6,$BDADR,0
      136206  001122  000000
21343 136212  001162  001116  000001  DT50:   .WORD   $TMP1,$ERRPC,R1,$BDADR,0
      136220  001122  000000
21344 136224  001162  001116  001124  DT51:   .WORD   $TMP1,$ERRPC,$GDDAT,PCR,0
      136232  177522  000000
21345 136236  001162  001116  001124  DT52:   .WORD   $TMP1,$ERRPC,$GDDAT,BCSR,0
      136244  177520  000000
21346 136250  001162  001116  001124  DT64:   .WORD   $TMP1,$ERRPC,$GDDAT,LKSFL,0
      136256  002722  000000
21347 136262  001162  001116  001124  DT65:   .WORD   $TMP1,$ERRPC,$GDDAT,0
      136270  000000
21348 136272  001162  001116  001124  DT75:   .WORD   $TMP1,$ERRPC,$GDDAT,$BDDAT,0
      136300  001126  000000
21349 136304  001162  001116  177562  DT105:  .WORD   $TMP1,$ERRPC,RBUF,0
      136312  000000
21350 136314  001162  001116  001126  DT115:  .WORD   $TMP1,$ERRPC,$BDDAT,KIPAR6,$BDADR,0
      136322  172354  001122  000000
21351 136330  001162  001122  000000  DT130:  .WORD   $TMP1,$BDADR,0
21352 136336  001162  001116  003022  DT134:  .WORD   $TMP1,$ERRPC,ERRCNT,R3,$BDDAT,R4,$BDADR,0
      136344  000003  001126  000004
      136352  001122  000000
```

MODIFIED ERROR MESSAGE TYPEOUT ROUTINE

```
21354                                   .SBTTL  MODIFIED ERROR MESSAGE TYPEOUT ROUTINE
21355                                   ;;*************************************************************************
21356                                   ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
21357                                   ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
21358                                   ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
21359                                   ;*
21360                                   ;*THE ONLY DIFFERENCE BETWEEN THIS ROUTINE AND THE ORIGINAL "$ERRTYP" FROM
21361                                   ;*SYSMAC IS THAT YOU CAN PASS INFORMATION IN GENERAL PURPOSE REGISTERS TO THIS
21362                                   ;*ROUTINE. THE GENERAL PURPOSE REGISTERS USED ARE TO BE SPECIFIED IN DT*
21363                                   ;*FORMAT. R0 SHOULD NOT BE USED.
21364
21365 136356                           ERTYPE:
21366 136356  005037  001162                   CLR     $TMP1           ;;JUST CLEAR IT
21367 136362  113737  001102  001162           MOVB    $TSTNM,$TMP1    ;;STORE TEST NUMBER
21368 136370  104401  001175                   TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
21369 136374  010046                           MOV     R0,-(SP)        ;;SAVE R0
21370 136376  005000                           CLR     R0              ;;PICKUP THE ITEM INDEX
21371 136400  153700  001114                   BISB    @#$ITEMP,R0
21372 136404  001004                           BNE     1$              ;;IF ITEM NUMBER IS ZERO, JUST
21373                                                                   ;;TYPE THE PC OF THE ERROR
21374 136406  013746  001116                   MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
21375                                                                   ;;ERROR ADDRESS
21376 136412  104402                           TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
21377 136414  000426                           BR      6$              ;;GET OUT
21378 136416  005300                   1$:     DEC     R0              ;;ADJUST THE INDEX SO THAT IT WILL
21379 136420  006300                           ASL     R0              ;;      WORK FOR THE ERROR TABLE
21380 136422  006300                           ASL     R0
21381 136424  006300                           ASL     R0
21382 136426  062700  001324                   ADD     #$ERRTB,R0      ;;FORM TABLE POINTER
21383 136432  012037  136442                   MOV     (R0)+,2$        ;;PICKUP "ERROR MESSAGE" POINTER
21384 136436  001404                           BEQ     3$              ;;SKIP TYPEOUT IF NO POINTER
21385 136440  104401                           TYPE
21386 136442  000000                   2$:     .WORD   0               ;;ERROR MESSAGE POINTER GOES HERE
21387 136444  104401  001175                   TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
21388 136450  012037  136460           3$:     MOV     (R0)+,4$        ;;PICKUP "DATA HEADER" POINTER
21389 136454  001404                           BEQ     5$              ;;SKIP TYPEOUT IF 0
21390 136456  104401                           TYPE                    ;;TYPE THE "DATA HEADER"
21391 136460  000000                   4$:     .WORD   0               ;;"DATA HEADER" POINTER GOES HERE
21392 136462  104401  001175                   TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
21393 136466  011000                   5$:     MOV     (R0),R0         ;;PICKUP "DATA TABLE" POINTER
21394 136470  001004                           BNE     7$              ;;GO TYPE THE DATA
21395 136472  012600                   6$:     MOV     (SP)+,R0        ;;RESTORE R0
21396 136474  104401  001175                   TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
21397 136500  000207                           RTS     PC              ;;RETURN
21398 136502                           7$:
21399 136502  021027  000005                   CMP     (R0),#5         ;;GENERAL PURPOSE REGISTER?
21400 136506  101021                           BHI     9$              ;;IF NOT, GO TYPE DATA
21401 136510  042737  000700  136544           BIC     #BIT8!BIT7!BIT6,8$      ;;CLEAR BITS FOR SOURCE REGISTER
21402 136516  011037  001160                   MOV     (R0),$TMP0      ;;SAVE (R0)
21403 136522  000337  001160                   SWAB    $TMP0           ;;GET REGISTER NUMBER TO HIGH BYTE
21404 136526  006237  001160                   ASR     $TMP0           ;;GET REGISTER NUMBER TO BITS 8-6
21405 136532  006237  001160                   ASR     $TMP0
21406 136536  053737  001160  136544           BIS     $TMP0,8$        ;;SET BITS IN MOV INSTRUCTION
21407                                                                   ;;ACCORDING TO REGISTER NUMBER
21408 136544  010046                   8$:     MOV     R0,-(SP)        ;;MOVE CONTEXT OF REGISTER TO STACK
21409 136546  005720                           TST     (R0)+           ;;ADVANCE POINTER
21410 136550  000401                           BR      10$             ;;GO TYPE
```

MODIFIED ERROR MESSAGE TYPEOUT ROUTINE

```
21411 136552  013046                  9$:     MOV     @(R0)+, (SP)    ;;IF NOT GPR, SAVE @(R0)+ FOR TYPEOUT
21412 136554  104402                  10$:    TYPOC                   ;;GO TYPE -OCTAL ASCI (ALL DIGITS)
21413 136556  005710                          TST     (R0)            ;;IS THERE ANOTHER NUMBER?
21414 136560  001744                          BEQ     6$              ;;BR IF NO
21415 136562  104401  136570                  TYPE    ,11$            ;;TYPE TWO(2) SPACES
21416 136566  000745                          BR      7$
21417 136570     040     040     000  11$:    .ASCIZ  /  /            ;;TWO(2) SPACES
21418                                         .EVEN
21419
21420                                 .SBTTL  GLOBAL SUBROUTINES SECTION
21421
21422                                 ;++
21423                                 ; THE GLOBAL SUBROUTINES SECTION CONTAINS THE SUBROUTINES
21424                                 ; THAT ARE USED IN MORE THAN ONE TEST.
21425                                 ;--
21426
21427                                 ;++
21428                                 ; FUNCTIONAL DESCRIPTION:
21429                                 ;    SUBROUTINE TO INITIALIZE ALL THE MMU REGISTERS
21430
21431
21432                                 ; INPUTS: NONE
21433
21434                                 ; OUTPUTS: NONE
21435
21436                                 ; SUBORDINATE ROUTINES USED: LOAD PARS
21437                                 ;                            LOAD PDRS
21438
21439                                 ; FUNCTIONAL SIDE EFFECTS: NONE
21440
21441                                 ; CALLING SEQUENCE:     JSR     PC,INITMM
21442
21443 136574  012701  172240  INITMM: MOV     #172240,R1              ;BASE ADDRESS OF SIPARS
21444 136600  004737  136736          JSR     PC,     LDPARS
21445 136604  012701  172260          MOV     #172260,R1              ;BASE ADDRESS OF SDPARS
21446 136610  004737  136736          JSR     PC,     LDPARS
21447 136614  012701  172340          MOV     #172340,R1              ;BASE ADDRESS OF KIPARS
21448 136620  004737  136736          JSR     PC,     LDPARS
21449 136624  012701  172360          MOV     #172360,R1              ;BASE ADDRESS OF KDPARS
21450 136630  004737  136736          JSR     PC,     LDPARS
21451 136634  012701  177640          MOV     #177640,R1              ;BASE ADDRESS OF UIPARS
21452 136640  004737  136736          JSR     PC,     LDPARS
21453 136644  012701  177660          MOV     #177660,R1              ;BASE ADDRESS OF UDPARS
21454 136650  004737  136736          JSR     PC,     LDPARS
21455 136654  012701  177600          MOV     #177600,R1              ;BASE ADDRESS OF UIPDRS
21456 136660  004737  136766          JSR     PC,     LDPDRS
21457 136664  012701  177620          MOV     #177620,R1              ;BASE ADDRESS OF UDPDRS
21458 136670  004737  136766          JSR     PC,     LDPDRS
21459 136674  012701  172300          MOV     #172300,R1              ;BASE ADDRESS OF KIPDRS
21460 136700  004737  136766          JSR     PC,     LDPDRS
21461 136704  012701  172320          MOV     #172320,R1              ;BASE ADDRESS OF KDPDRS
21462 136710  004737  136766          JSR     PC,     LDPDRS
21463 136714  012701  172200          MOV     #172200,R1              ;BASE ADDRESS OF SIPDRS
21464 136720  004737  136766          JSR     PC,     LDPDRS
21465 136724  012701  172220          MOV     #172220,R1              ;BASE ADDRESS OF SDPDRS
21466 136730  004737  136766          JSR     PC,     LDPDRS
21467 136734  000207                  RTS     PC                      ;RETURN
```

GLOBAL SUBROUTINES SECTION

```
21469                                   ;**
21470                                   ; FUNCTIONAL DESCRIPTION:
21471                                   ;    SUBROUTINE TO INITIALIZE ALL THE MMU PAGE ADDRESS REGISTERS (PARS).
21472                                   ;       THIS ROUTINE WILL INITIALIZE 8 PARS STARTING AT A BASE ADDRESS
21473                                   ;       SUPPLIED BY THE CALLING ROUTINE. PARS 0 5 WILL BE MAPPED FROM
21474                                   ;       ADDRESS 0 TO ADDRESS 137777 (0-24K). PAR 6 WILL BE MAPPED FROM
21475                                   ;       ADDRESS 200000 TO 217777 AND PAR 7 WILL BE MAPPED TO THE I/O
21476                                   ;       PAGE.
21477
21478                                   ; INPUTS:
21479                                   ;       R1 CONTAINS THE BASE ADDRESS OF THE NEXT 8 PARS TO BE INITIALIZED
21480
21481                                   ; OUTPUTS: NONE
21482
21483                                   ; SUBORDINATE ROUTINES USED: NONE
21484
21485                                   ; FUNCTIONAL SIDE EFFECTS: NONE
21486
21487                                   ; CALLING SEQUENCE:     JSR     PC,LDPARS
21488
21489 136736  012702  000006          LDPARS: MOV     #6,     R2              ;LET LOOP COUNTER COUNT FIRST 6 PARS
21490 136742  005003                          CLR     R3                      ;INITIALIZE INDEX VALUE
21491 136744  010321                  1$:     MOV     R3,     (R1)+           ;LOAD PARS
21492 136746  062703  000200                  ADD     #200,   R3              ;INDEX IN 4K INCREMENTS
21493 136752  077204                          SOB     R2,     1$              ;LOAD FIRST SIX PARS
21494 136754  012721  002000                  MOV     #2000,  (R1)+           ;LET PAR6 MAP TO 200000
21495 136760  012711  177600                  MOV     #177600,(R1)            ;LET PAR7 MAP TO I/O PAGE
21496 136764  000207                          RTS     PC                      ;RETURN
```

GLOBAL SUBROUTINES SECTION

```
21498                                   ;++
21499                                   ; FUNCTIONAL DESCRIPTION:
21500                                   ;    SUBROUTINE TO INITIALIZE ALL THE MMU PAGE DECRIPTOR REGISTERS (PDRS).
21501                                   ;       THIS ROUTINE WILL INITIALIZE 8 PDRS STARTING AT A BASE ADDRESS
21502                                   ;       SUPPLIED BY THE CALLING ROUTINE. PDRS 0 5 WILL BE INITIALIZED TO
21503                                   ;       4K READ/WRITE BYPASS AND PDRS 6 AND 7 WILL BE INITIALIZED TO
21504                                   ;       4K READ/WRITE NO BYPASS.
21505                                   ;       NOTE:   THERE IS NO NEED TO BYPASS ON I/O PAGE REFERENCES BECAUSE
21506                                   ;               THE CACHE DOES NOT ALLOCATE ANY OF THESE REFERENCES.
21507
21508                                   ; INPUTS:
21509                                   ;       R1 CONTAINS THE BASE ADDRESS OF THE NEXT 8 PDRS TO BE INITIALIZED
21510
21511                                   ; OUTPUTS: NONE
21512
21513                                   ; SUBORDINATE ROUTINES USED: NONE
21514
21515                                   ; FUNCTIONAL SIDE EFFECTS: NONE
21516
21517                                   ; CALLING SEQUENCE:     JSR     PC,LDPARS
21518
21519 136766  012702  000006          LDPDRS: MOV     #6,     R2              ;LET LOOP COUNTER COUNT FIRST 6 PARS
21520 136772  012721  177406          1$:     MOV     #177406,(R1)+           ;LOAD PDRS WITH 4K READ/WRITE BYPASS
21521 136776  077203                          SOB     R2,     1$              ;LOAD FIRST SIX PDRS
21522 137000  012721  077406                  MOV     #77406, (R1)+           ;LET PAR6 BE 4K READ/WRITE NO BYPASS
21523 137004  012711  077406                  MOV     #77406, (R1)            ;LET PAR7 BE 4K READ/WRITE NO BYPASS ALSO
21524 137010  000207                          RTS     PC                      ;RETURN
```

```
21526                                   ;++
21527                                   ; FUNCTIONAL DESCRIPTION:
21528                                   ;    SUBROUTINE TO HANDLE PARITY ERROR ABORTS FROM THE RAM STORE RAM TESTS.
21529                                   ;
21530                                   ; INPUTS:
21531                                   ;       MEMORY SYSTEM ERROR REGISTER CONTAINS BITS INDICATING FAILURE
21532                                   ;
21533                                   ; OUTPUTS: NONE
21534                                   ;
21535                                   ; SUBORDINATE ROUTINES USED: NONE
21536                                   ;
21537                                   ; FUNCTIONAL SIDE EFFECTS: NONE
21538                                   ;
21539                                   ; CALLING SEQUENCE: CALLED BY PARITY ABORT
21540                                   ;       MOV     @#114,  SLOC00  ;SAVE CONTENTS OF PARITY ABORT VECTOR
21541                                   ;       MOV     #DSPAR, @#114   ;LET VECTOR POINT TO PARITY ABORT ROUTINE
21542                                   ;
21543                                   ;       (CACHE PARITY ERROR OCCURS)
21544                                   ;
21545 137012  011637  001122            RAMPAR: MOV     (SP),$BDADR             ;STOR ADDESS TRAPPED
21546 137016  032737  000100  177744            BIT     #BIT06, MSER            ;IF LOW BYTE PARITY ERROR
21547 137024  001401                            BEQ     1$                      ;THEN
21548 137026  104007                            ERROR   +7                      ;ERROR
21549 137030  032737  000200  177744    1$:     BIT     #BIT07, MSER            ;IF HIGH BYTE PARITY ERROR
21550 137036  001401                            BEQ     2$                      ;THEN
21551 137040  104007                            ERROR   +7                      ;ERROR
21552 137042  032737  000040  177744    2$:     BIT     #BIT05, MSER            ;IF TAG PARITY ERROR
21553 137050  001401                            BEQ     3$                      ;THEN
21554 137052  104007                            ERROR   +7                      ;ERROR
21555 137054  005037  177744            3$:     CLR     MSER                    ;INITIALIZE MSER AFTER ERROR
21556 137060  000002                            RTI                             ;RETURN
21557 137062  005237  002722            LKSINT: INC     LKSFL                   ;INCREMENT FLAG
21558 137066  000002                            RTI
21559                                   
21560                                           .SBTTL Q22BE SIZE ROUTINE
21561                                   ;THIS ROUTINE WILL AUTOSIZE FOR UP TO TWO Q22 BUS EXERCISERS.  IF NONE
21562                                   ;FOUND LOCATIONS CSR1 AND CSR12 WILL BE LEFT ZEROES.  THIS ROUTINE WILL
21563                                   ;ONLY RUN IN NOT UFD MODE.
21564                                   
21565 137070  032737  001000  177750    Q22SIZ: BIT     #BIT09,MAIREG                   ;UNIBUS SYSTEM?
21566 137076  001401                            BEQ     1$                              ;IF NOT, ADVANCE TO ROUTINE
21567 137100  000207                            RTS     PC                              ;OTHERWISE, RETURN
21568                                   ;
21569                                   ; PREPARE TO DO SIZING
21570                                   ;
21571 137102  013701  000004            1$:     MOV     ERRVEC,R1                       ;STORE TIMEOUT VECTOR
21572 137106  012737  137262  000004            MOV     #7$,ERRVEC                      ;POINT NEW TO PROGRAM
21573 137114  012737  000340  000006            MOV     #340,ERRVEC+2                   ;AT PRIORITY 7
21574 137122  005037  001160                    CLR     $TMP0                           ;CLEAR Q22BE COUNTER
21575 137126  012702  170000                    MOV     #170000,R2                      ;FIRST POSSIBLE ADDRESS
21576 137132  012703  000510                    MOV     #510,R3                         ;VECTOR FOR IT
21577 137136  000404                            BR      3$                              ;TRY THOSE VALUES
21578                                   ;
21579                                   ; NOW DO ACTUAL SIZING
21580                                   ;
21581 137140  062702  000020            2$:     ADD     #20,R2                          ;GET CSR FOR NEXT Q22BE
21582 137144  062703  000004                    ADD     #4,R3                           ;GET VECTOR FOR NEXT ONE
```

Q22BE SIZE ROUTINE

```
21583 137150  005712                    3$:     TST     (R2)                    ;TRY TO ACCESS CSR
21584                                   ;
21585                                   ; IF NO TIMEOUT, STORE EXISTING ADDRESSES TO REGISTERS
21586                                   ;
21587 137152  005737  001160                    TST     $TMP0                   ;FIRST Q22BE FOUND?
21588 137156  001010                            BNE     4$                      ;IF SECOND, BRANCH
21589 137160  012705  002664                    MOV     #CSR1,R5                ;START WITH CSR1 FOR 1ST
21590 137164  010237  002702                    MOV     R2,SIMGOA               ;SIMULTANEOUS GO
21591 137170  062737  000016  002702            ADD     #16,SIMGOA              ;ADDRESS
21592 137176  000402                            BR      5$                      ;BRANCH TO INITIALISE
21593 137200  012705  002704            4$:     MOV     #CSR12,R5               ;START WITH CSR12 FOR 2ND
21594 137204  012704  000004            5$:     MOV     #4,R4                   ;INITIALISE 5 REGISTERS
21595 137210  010215                            MOV     R2,(R5)                 ;INITIALISE CSR1
21596 137212  011565  000002            6$:     MOV     (R5),2(R5)              ;STORE TO NEXT ONE
21597 137216  005725                            TST     (R5)+                   ;GET NEXT ADDRESS
21598 137220  062715  000002                    ADD     #2,(R5)                 ;GET ADDRESS, POINT NEXT
21599 137224  077406                            SOB     R4,6$                   ;DO FOR NEXT 4 REGISTERS
21600 137226  010365  000002                    MOV     R3,2(R5)                ;STORE INTERRUPT VEC'OR
21601 137232  010365  000004                    MOV     R3,4(R5)                ;AND PRIORITY
21602 137236  062765  000002  000004            ADD     #2,4(R5)
21603 137244  005237  001160                    INC     $TMP0                   ;COUNT Q22BE'S
21604 137250  022737  000002  001160            CMP     #2,$TMP0                ;TWO FOUND?
21605 137256  001406                            BEQ     9$                      ;IF SO, STOP SIZING
21606 137260  000402                            BR      8$                      ;OTHERWISE, CONTINUE SIZING
21607                                   ;
21608                                   ; ON TIMEOUT TRY TO LOOK AT NEXT ADDRESS RANGE
21609                                   ;
21610 137262  005726                    7$:     TST     (SP)+                   ;RESTORE STAC\ FRO"
21611 137264  005726                            TST     (SP)+                   ;TIMEOUT
21612 137266  022702  170160            8$:     CMP     #170160,R2              ;AT THE LAST POSSIBLE?
21613 137272  '01322                           BNE     2$                      ;IF NOT, BRANCH
21614 137274    5737  002664            9$:     TST     CSR1                    ;1 FOUND?
21615 137300  ..1402                           BEQ     10$                     ;IF NONE, BRANCH
21616 137302  104401  137314                    TYPE    ,ONOQ22                 ;TYPE FOUND
21617 137306  010137  000004            10$:    MOV     R1,ERRVEC               ;RESTORE TIMEOUT VECTOR
21618 137312  000207                            RTS     PC                      ;RETURN
21619
21620 137314      012     015     121   ONOQ22: .ASCIZ  <12><15>/Q22BE USED DURING TESTING/
      137317      062     062     102
      137322      105     040     125
      137325      123     105     104
      137330      040     104     125
      137333      122     111     116
      137336      107     040     124
      137341      105     123     124
      137344      111     116     107
      137347      000
21621                                           .EVEN
21622                                           .SBTTL Q22BE INTERRUPT INITIALISE ROUTINE
21623                                           ;THIS ROUTINE WILL INITIALISE Q22BE TO INTERRUPT AT A PRIORITY AT (R0)+
21624                                           ;AT THE STARTING ADDRESS IN R3. THE TEST HAVE TO SET ACTUAL DONE BIT
21625                                           ;BY CLEARING GO.
21626
21627 137350  005013                    Q22INT: CLR     (R3)                    ;CLEAR TRANSFER TYPE IN CSR1
21628 137352  052710  000001                    BIS     #BIT00,(R0)             ;ZERO DONE
21629 137356  011063  000002                    MOV     (R0),2(R3)              ;SET PRIORITY IN CSR2
21630 137362  042710  000001                    BIC     #BIT00,(R0)             ;PREPARE TO SET DONE
```

F2

Q22BE INTERRUPT INITIALISE ROUTINE

```
21631 137366  000207                          RTS     PC
21632
21633                                 .SBTTL DMATRN DATO CYCLE THRU Q22BE
21634                                 ;THIS ROUTINE PERFORMS DATO FROM A LOCATION TEMP THRU THE FIRST
21635                                 ;FOUND Q22BE STARTING AT LOCATION @CSR1. R0 HAS 0 IF ONLY 1 TRANSFER IS
21636                                 ;TO BE PERFORMED. OTHERWISE 16 BLOCK MODE TRANSFERS ARE TO BE PERFORMED.
21637                                 ;IN THE LATTER CASE ADDRESS AND WORD COUNT HAS TO BE LOADED BEFORE.
21638
21639 137370  012777  012525  043276  DMATRN: MOV     #12525,@DATA                    ;DATA USED
21640 137376  005700                          TST     R0                              ;DO 1 WORD?
21641 137400  001404                          BEQ     1$                              ;IF YES, BRANCH
21642 137402  012777  001001  043256          MOV     #BIT09!BIT00,@CSR2              ;BLOCK MODE, GO
21643 137410  000414                          BR      2$                              ;BRANCH TO DO IT
21644 137412  012777  001601  043244  1$:     MOV     #1601,@CSR1                     ;RESET LATENCY COUNT,DATO
21645 137420  012777  002740  043242          MOV     #TEMP,@BA                       ;LOAD DMA ADDRESS
21646 137426  012777  177777  043236          MOV     #177777,@WC                     ;DO 1 WORD
21647 137434  012777  000001  043224          MOV     #BIT00,@CSR2                    ;DO IT
21648 137442  105777  043220          2$:     TSTB    @CSR2                           ;DMA DONE?
21649 137446  100375                          BPL     2$                              ;WAIT TILL DONE
21650 137450  000207                  3$:     RTS     PC                              ;RETURN FROM SUBROUTINE
21651
21652                                 .SBTTL DMARD DATI THRU Q22BE
21653                                 ;THIS ROUTINE PERFORMS DATI CYCLE THRU Q22BE IN EITHER BLOCK MODE OR A SINGLE
21654                                 ;TRANSFER MODE. MEMORY LOCATION USED IS TEMP. R0 IS ZERO FOR SINGLE TRANSFER
21655
21656 137452  012777  002740  043210  DMARD:  MOV     #TEMP,@BA                       ;LOAD DMA ADDRESS
21657 137460  005700                          TST     R0                              ;DO 1 WORD?
21658 137462  001412                          BEQ     1$                              ;IF YES, BRANCH
21659 137464  012777  001507  043172          MOV     #1507,@CSR1                     ;16 DATIB
21660 137472  012777  177770  043172          MOV     #177770,@WC                     ;DO 8 WORD
21661 137500  012777  001001  043160          MOV     #BIT09!BIT00,@CSR2              ;BLOCK MODE GO
21662 137506  000411                          BR      2$                              ;GO CHECK
21663 137510  012777  001407  043146  1$:     MOV     #1407,@CSR1                     ;RESET LATENCY COUNT,DATI
21664                                                                                 ;LOAD NEW DATA TO DATA R.
21665 137516  012777  177777  043146          MOV     #177777,@WC                     ;DO 1 WORD
21666 137524  012777  000001  043134          MOV     #BIT00,@CSR2                    ;DO IT
21667 137532  105777  043130          2$:     TSTB    @CSR2                           ;DMA DONE?
21668 137536  100375                          BPL     2$                              ;WAIT TILL DONE
21669 137540  000207                  3$:     RTS     PC                              ;RETURN FROM SUBROUTINE
21670
21671
21672
21673 137542  011637  001122          TOUT:   MOV     (SP),$BDADR             ;STORE TRAPPED PC
21674 137546  104130                          ERROR   +130                    ;UNEXPECTED TRAP
21675 137550  000002                          RTI
21676
21678                                 ;
21679                                 ;MMU GLOBAL SUBROUTINES
21680                                 ;
21681                                 ;
21682                                 ;ROUTINE TO INITIALIZE MEMORY MANAGEMENT
21683                                 ;
21684 137552  010046          MMU:    MOV     R0,-(SP)                ;SAVE CONTENTS OF REGISTERS
21685 137554  010146                  MOV     R1,-(SP)                ;
21686 137556  010246                  MOV     R2,-(SP)                ;
21687 137560  012700  177600          MOV     #177600,R0              ;
21688 137564  004737  137652          JSR     PC,PDR                  ;INIT I AND D USER PDR'S
```

DMARD DATI THRU Q22BE

```
21689 137570  004737  137674                  JSR     PC,PAR                  ;INIT I USER PAR'S
21690 137574  004737  137674                  JSR     PC,PAR                  ;INIT D USER PAR'S
21691 137600  012700  172200                  MOV     #172200,R0
21692 137604  004737  137652                  JSR     PC,PDR                  ;INIT I AND D SUP PDR'S
21693 137610  004737  137674                  JSR     PC,PAR                  ;INIT I SUP PAR'S
21694 137614  004737  137674                  JSR     PC,PAR                  ;INIT D SUP PAR'S
21695 137620  004737  137652                  JSR     PC,PDR                  ;INIT I AND D KER PDR'S
21696 137624  004737  137674                  JSR     PC,PAR                  ;INIT I KER PAR'S
21697 137630  004737  137674                  JSR     PC,PAR                  ;INIT D KER PAR'S
21698 137634  012737  000027  172516          MOV     #27,@#172516            ;INIT MMR3
21699 137642  012602                          MOV     (SP)+,R2                ;RESTORE REGISTERS
21700 137644  012601                          MOV     (SP)+,R1                ;
21701 137646  012600                          MOV     (SP)+,R0                ;
21702 137650  000207                          RTS     PC                      ;RETURN
21703
21704                                 ;
21705                                 ;ROUTINE TO INITIALIZE PDR'S
21706                                 ;
21706 137652  005002                  PDR:    CLR     R2                      ;INIT CNTR
21707 137654  012720  077406          PDR1:   MOV     #77406,(R0)+            ;INIT PDR
21708 137660  062702  000001                  ADD     #1,R2                   ;INCREMENT CNTR
21709 137664  022702  000020                  CMP     #16.,R2                 ;ARE WE DONE?
21710 137670  001371                          BNE     PDR1                    ;BRANCH IF NOT
21711 137672  000207                          RTS     PC                      ;RETURN
21712
21713                                 ;
21714                                 ;ROUTINE TO INITIALIZE PAR'S
21715                                 ;
21715 137674  005001                  PAR:    CLR     R1                      ;SETUP TO INIT PAR
21716 137676  010120                  PAR1:   MOV     R1,(R0)+                ;INIT PAR
21717 137700  062701  000200                  ADD     #200,R1                 ;GET READY FOR NEXT PAR
21718 137704  022701  001600                  CMP     #1600,R1                ;REACHED A PAR7
21719 137710  001372                          BNE     PAR1                    ;BRANCH IF NOT
21720 137712  012720  177600                  MOV     #177600,(R0)+           ;INIT PAR7
21721 137716  000207                          RTS     PC                      ;RETURN
21722
21723                                 ;
21724                                 ;TIME OUT ROUTINE
21725                                 ;
21725 137720  005205                  ADDTRP: INC     R5                      ;INCREMENT TIME OUT FLAG
21726 137722  000002                          RTI                             ;RETURN
21727
21728                                 ;
21729                                 ;MMU TRAP ROUTINE
21730                                 ;
21730 137724  023727  003030  000001  MMUTRP: CMP     FLAG,#1                 ;ARE WE EXPECTING AN ABORT
21731 137732  001401                          BEQ     1$                      ;YES GO ON
21732 137734  104002                          ERROR   +2              ;NO GO TO ERROR
21733 137736  010046                  1$:     MOV     R0,-(SP)                ;SAVE CONTENTS OF REG 0
21734 137740  013700  177776                  MOV     @#177776,R0             ;SAVE A COPY OF PSW
21735 137744  072027  177764                  ASH     #-14,R0                 ;LOOK AT BITS<15:14>
21736 137750  020027  000002                  CMP     R0,#2                   ;WAS PS<15:14>=10
21737 137754  001001                          BNE     OK                      ;NO GO ON
21738 137756  000411                          BR      NOTOK                   ;YES CHANGE BITS TO 00
21739 137760  013700  177776          OK:     MOV     @#177776,R0             ;SAVE A COPY OF PSW
21740 137764  072027  000002                  ASH     #2,R0                   ;LOOK AT BITS<13:12>
21741 137770  072027  177764                  ASH     #-14,R0                 ;
21742 137774  020027  000002                  CMP     R0,#2                   ;WAS PS<13:12>=10
21743 140000  001002                          BNE     OK1                     ;NO GO ON
21744 140002  005066  000004          NOTOK:  CLR     4(SP)                   ;CLEAR ILLEGAL MODE FFROM OLD PSW
21745 140006  013737  177572  003042  OK1:    MOV     @#177572,SAVMR0         ;SAVE A COPY OF MMR0
```

H2

SEQ 0434

DMARD DATI THRU Q22BE

```
 21746 140014  013737  177574  003044          MOV     @#177574,SAVMR1         ;SAVE A COPY OF MMR1
 21747 140022  013737  177576  003046          MOV     @#177576,SAVMR2         ;SAVE A COPY OF MMR2
 21748 140030  005037  177572                  CLR     @#177572                ;CLEAR ABORT BITS AND TURN MMU OFF
 21749 140034  005037  003030                  CLR     FLAG                    ;CLEAR MMU ABORT FLAG
 21750 140040  012600                          MOV     (SP)+,R0                ;RESTORE ORIGINAL CONTENTS OF REG 0
 21751 140042  000002                          RTI                             ;RETURN
```

```
21755                                   ;FPP COMMON SUBROUTINES
21756 140044  012600                    WLDTRP: MOV     (SP)+,R0                ;SAVE PC
21757 140046  012605                            MOV     (SP)+,R5                ;SAVE STATUS AND RESTORE STACK
21758 140050  104003                            ERROR   +3
21759 140052  000110                            JMP     (R0)                    ;GO BACK INLINE
21760                                   ;
21761                                   ;
21762                                   ;
21763 140054  000000                    TRPFLG: .WORD   0
21764 140056  000207                    ERRFP:  RTS     R7
21765 140060  000207                    ERR:    RTS     R7
21766                                   ;
21767                                   ;
21768                                   ;
21769                                   ;
21770                                   ;
21771                                   ;SUBROUTINE DATA VERFICATION -
21772                                   ;
21773                                   ; CALLED BY     JSR     R7,DATVER
21774                                   ;
21775                                   ;INPUT:         (R4)=EXPECTED DATA
21776                                   ;               (R1)=RECEIVED DATA
21777                                   ;
21778                                   ;THIS ROUTINE VERIFIES THAT THE 4 CONSECTIVE WORDS STARTING WITH (R4) ARE
21779                                   ;EQUAL TO THE FOUR WORDS ADDRESSED BY (R1). THE CONTENTS OF R4, AND R1 ARE NOT
21780                                   ;DISTURBED.
21781                                   ;LOCATION "COUNT" , IF NOT EQUAL TO 0 SIGNIFIES DATA ERROR
21782                                   ;IF THE STATUS IS FLOATING MODE, THE LAST TWO BYTES OF RECEIEVED
21783                                   ;ARE SIMPLY CHECKED FOR ZEROS
21784                                   ;
21785
21786 140062  010446                    DATVFR: MOV     R4,-(SP)                ;SAVE R4
21787 140064  010146                            MOV     R1,-(SP)                        ;SAVE R1
21788 140066  012737  000003  003120            MOV     #3,COUNT                ;SET UP ITERATION COUNT
21789 140074  000137  140112                    JMP     DAT1                    ;
21790                                   ;
21791 140100  010446                    DATVER: MOV     R4,-(SP)                ;SAVE R4
21792 140102  010146                            MOV     R1,-(SP)                ;SAVE R1
21793 140104  012737  000005  003120            MOV     #5,COUNT                ;SET UP ITERATION COUNT
21794 140112  005337  003120            DAT1:   DEC     COUNT
21795 140116  001402                            BEQ     2$                      ;BRANCH IF DONE
21796 140120  022421                            CMP     (R4)+,(R1)+                     ;
21797 140122  001773                            BEQ     DAT1
21798 140124  012601                    2$:     MOV     (SP)+,R1                ;RESTORE R1
21799 140126  012604                            MOV     (SP)+,R4                ;RESTORE R4
21800 140130  000207                            RTS     R7                      ;GO BACK TO CALLING ROUTINE
21801                                                                           ;IF DATA ERROR, COUNT NE 0
```

J2

DMARD DATI THRU Q22BE

```
21803                                   ;
21804                                   ;$$$
21805                                   ;
21806                                   ; SUBROUTINE - DETERMINE FLOATING POINT ACCELERATOR (DETFPA)
21807                                   ;
21808                                   ; THIS SUBROUTINE IS CALLED IF AN ERROR IS DETECTED DURING EXECUTION OF THE
21809                                   ; FLOATING POINT TESTS.
21810                                   ; IT DETERMINES WHEATHER OR NOT THE FLOATING POINT ACCELERATOR CHIP OPTION
21811                                   ; IS PRESENT ON THE CPU BOARD AND PRINTS THE APPROPRIATE ERROR MESSAGE.
21812                                   ; THIS DETERMINATION IS MADE BASED ON THE "FPA AVAILABLE" FLAG, BIT 8
21813                                   ; OF THE MAINTENANCE REGISTER AT LOCATION 17777750. IF THE FPA BIT IS SET
21814                                   ; THEN THE FLOATING POINT ACCELERATOR CHIP IS INSTALLED ON THE CPU BOARD AND
21815                                   ; AN ERROR MESSAGE IS PRINTED WHICH STATES THAT THE FLOATING POINT ERROR IS
21816                                   ; DUE TO THIS CHIP. OTHERWISE, THE J11 IS BLAMED FOR THE FLOATING POINT ERROR.
21817                                   ;
21818                                   ;$$$
21819                                   ;
21820                                   ; CALLED BY: CALL       @#DETFPA ;$$$
21821                                   ;
21822                                   ; INPUTS: NONE                   ;$$$
21823                                   ;
21824                                   ; OUTPUTS: ERROR MESSAGES        ;$$$
21825                                   ;
21826 140132  032737  000400  177750  DETFPA: BIT     #400,@#MAIREG   ;IS THE FPA HERE? $$$
21827 140140  001007                          BNE     FPAOPT          ;YES, BRANCH FPAOPT $$$
21828 140142  000240                          NOP                     ;DEBUG AID. $$$
21829 140144  032737  000400  177750          BIT     #400,@#MAIREG   ;IF NOT, $$$
21830 140152  001405                          BEQ     NOFPA           ;BRANCH TO NOFPA $$$
21831 140154  000240                          NOP                     ;DEBUG AID. $$$
21832 140156  000000                          HALT                    ; $$$
21833                                   ;
21834 140160  104401  140174          FPAOPT: TYPE    ,FPAFLT         ; $$$
21835 140164  000402                          BR      EXTFPA          ; $$$
21836 140166  104401  140257          NOFPA:  TYPE    ,J11FLT         ; $$$
21837 140172  000207                  EXTFPA: RTS     PC              ; $$$
21838                                   ;
21839 140174      105     122     122  FPAFLT: .ASCIZ  /ERROR DETECTED IN FLOATING POINT ACCELERATOR CHIP./ ; $$$
      140177      117     122     040
      140202      104     105     124
      140205      105     103     124
      140210      105     104     040
      140213      111     116     040
      140216      106     114     117
      140221      101     124     111
      140224      116     107     040
      140227      120     117     111
      140232      116     124     040
      140235      101     103     103
      140240      105     114     105
      140243      122     101     124
      140246      117     122     040
      140251      103     110     111
      140254      120     056     000
21840                                   ;
21841 140257      105     122     122  J11FLT: .ASCIZ  /ERROR DETECTED IN J11 FLOATING POINT PROCESSOR./ ; $$$
      140262      117     122     040
      140265      104     105     124
```

K2

SEQ 0437

DMARD DATI THRU Q22BE

```
        140270      105     103     124
        140273      105     104     040
        140276      111     116     040
        140301      112     061     061
        140304      040     106     114
        140307      117     101     124
        140312      111     116     107
        140315      040     120     117
        140320      111     116     124
        140323      040     120     122
        140326      117     103     105
        140331      123     123     117
        140334      122     056     000
21842                                   .EVEN                   ; ###
21843                           ;
```

```
21846
21847
21848                                   .SBTTL  END OF PASS ROUTINE
                                        ;;*****************************************************************
                                        ;*INCREMENT THE PASS NUMBER ($PASS)
                                        ;*INDICATE END-OF PROGRAM AFTER 1 PASSES THRU THE PROGRAM
                                        ;*TYPE "END PASS #XXXXX" (WHERE XXXXX IS A DECIMAL NUMBER)
                                        ;*IF THERES A MONITOR GO TO IT
                                        ;*IF THERE ISN'T JUMP TO LOOP
140340                                  $EOP:
140340  032737  000100  000052                BIT #BIT06,@#52
140346  001030                                BNE $GET42
140350  004537  126756                        jsr r5,vireop
140354  005037  001102                        CLR     $TSTNM          ;;ZERO THE TEST NUMBER
140360  005037  001164                        CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
140364  005237  001206                        INC     $PASS           ;;INCREMENT THE PASS NUMBER
140370  042737  100000  001206              BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
140376  005327                              DEC     (PC)+           ;;LOOP?
140400  000001                  $EOPCT: .WORD   1
140402  003022                              BGT     $DOAGN          ;;YES
140404  012737                              MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
140406  000001                  $ENDCT: .WORD   1
140410  140400                              $EOPCT
140412  104401  140457                      TYPE    ,$ENDMG         ;;TYPE "END PASS #"
140416  013746  001206                      MOV     $PASS, (SP)     ;;SAVE $PASS FOR TYPEOUT
140422  104405                              TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
140424  104401  140454                      TYPE    ,$ENULL         ;;TYPE A NULL CHARACTER
140430  013700  000042          $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
140434  001405                              BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
140436  000005                              RESET                   ;;CLEAR THE WORLD
140440  004710                  $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
140442  000240                              NOP                     ;;SAVE ROOM
140444  000240                              NOP                     ;;FOR
140446  000240                              NOP                     ;;ACT11
140450                          $DOAGN:
140450  000137                              JMP     @(PC)+          ;;RETURN
140452  004740                  $RTNAD: .WORD   LOOP
140454     377     377     000  $ENULL: .BYTE   -1,-1,0             ;;NULL CHARACTER STRING
140457     015     012     105  $ENDMG: .ASCIZ  <15><12>/END PASS #/
140462     116     104     040
140465     120     101     123
140470     123     040     043
140473     000
21849                                   .SBTTL  SCOPE HANDLER ROUTINE
                                        ;;*****************************************************************
                                        ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
                                        ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
                                        ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
                                        ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
                                        ;*SW14=1        LOOP ON TEST
                                        ;*SW11=1        INHIBIT ITERATIONS
                                        ;*SW09=1        LOOP ON ERROR
                                        ;*SW08=1        LOOP ON TEST IN SWR<5:0>
                                        ;*CALL
                                        ;*      SCOPE           ;;SCOPE=IOT
140474                                  $SCOPE:
140474  104407                              CKSWR                   ;;TEST  FOR CHANGE IN SOFT-SWR
```

M2

SCOPE HANDLER ROUTINE

```
140476  052737  001000  177520          BIS #1000,BCSR ;ENABLE
140504  032777  040000  040426  1$:     BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
140512  001117                          BNE     $OVER           ;;YES IF SW14=1
                                ;#####START OF CODE FOR THE XOR TESTER#####
140514  000416                  $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
                                                                ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
140516  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
140522  012737  140542  000004          MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
140530  005737  177060                  TST     @#177060        ;;TIME OUT ON XOR?
140534  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
140540  000466                          BR      $SVLAD          ;;GO TO THE NEXT TEST
140542  022626                  5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
140544  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
140550  000426                          BR      7$              ;;LOOP ON THE PRESENT TEST
140552                          6$:;#####END OF CODE FOR THE XOR TESTER#####
140552  032777  000400  040360          BIT     #BIT08,@SWR     ;;LOOP ON SPEC. TEST?
140560  001407                          BEQ     2$              ;;BR IF NO
140562  017746  040352                  MOV     @SWR,-(SP)      ;;SET DESIRED TEST NUM. FROM SWR
140566  042716  000300                  BIC     #$SWRMK,(SP)    ;;STRIP AWAY UNDESIRED BITS
140572  122637  001102                  CMPB    (SP)+,$TSTNM    ;;ON THE RIGHT TEST?
140576  001465                          BEQ     $OVER           ;;BR IF YES
140600  105737  001103          2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
140604  001421                          BEQ     3$              ;;BR IF NO
140606  123737  001115  001103          CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
140614  101015                          BHI     3$              ;;BR IF NO
140616  032777  001000  040314          BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
140624  001404                          BEQ     4$              ;;BR IF NO
140626  013737  001110  001106  7$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
140634  000446                          BR      $OVER
140636  105037  001103          4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
140642  005037  001164                  CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
140646  000415                          BR      1$              ;;ESCAPE TO THE NEXT TEST
140650  032777  004000  040262  3$:     BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
140656  001011                          BNE     1$              ;;BR IF YES
140660  005737  001206                  TST     $PASS           ;;IF FIRST PASS OF PROGRAM
140664  001406                          BEQ     1$              ;;      INHIBIT ITERATIONS
140666  005237  001104                  INC     $ICNT           ;;INCREMENT ITERATION COUNT
140672  023737  001164  001104          CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
140700  002024                          BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
140702  012737  000001  001104  1$:     MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
140710  013737  140774  001164          MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
140716  105237  001102          $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
140722  113737  001102  001204          MOVB    $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
140730  011637  001106                  MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
140734  011637  001110                  MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
140740  005037  001166                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
140744  112737  000001  001115          MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
140752  013777  001102  040162  $OVER:  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
140760  013716  001106                  MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
140764  042737  001000  177520          BIC #1000,BCSR ;DISABLE
140772  000002                          RTI
140774  000001                  $MXCNT: 1                       ;;MAX. NUMBER OF ITERATIONS
21850                           .SBTTL  ERROR HANDLER ROUTINE
                                ;;**************************************************************
                                ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
                                ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
                                ;*AND GO TO ERTYPE ON ERROR
```

N2

SEQ 0440

ERROR HANDLER ROUTINE

```
                                ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
                                ;*SW15=1        HALT ON ERROR
                                ;*SW13=1        INHIBIT ERROR TYPEOUTS
                                ;*SW10=1        BELL ON ERROR
                                ;*SW09=1        LOOP ON ERROR
                                ;*CALL
                                ;*      ERROR   +N      ;;ERROR=EMT AND N=ERROR ITEM NUMBER
140776
140776  005737  004120          $ERROR:
                                        TST     UQUIET          ;;TEST FOR USER-QUIET MODE
141002  001403                          BEQ     9$              ;;BRANCH IF FIELD-SERVICE MODE
141004  005000                          CLR     R0              ;;IN CASE R0 HAS A #3 IN IT (↑C)
141006  004737  141220                  JSR     PC,ABORT        ;;TEST FOR ABORT CONDITION
141012                          9$:
141012  104407                          CKSWR                   ;;TEST FOR CHANGE IN SOFT SWR
141014  052737  001000  177520          BIS     #1000,BCSR              ;ENABLE HALT ON BREAK
141022  105237  001103          7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
141026  001775                          BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
141030  013777  001102  040104          MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
141036  032777  002000  040074          BIT     #BIT10,@SWR     ;;BELL ON ERROR?
141044  001402                          BEQ     1$              ;;NO - SKIP
141046  104401  001170                  TYPE    ,$BELL          ;;RING BELL
141052  005237  001112          1$:     INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
141056  011637  001116                  MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
141062  162737  000002  001116          SUB     #2,$ERRPC
141070  117737  040022  001114          MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
141076  032777  020000  040034          BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
141104  001004                          BNE     20$             ;;SKIP TYPEOUTS
141106  004737  136356                  JSR     PC,ERTYPE       ;;GO TO USER ERROR ROUTINE
141112  104401  001175                  TYPE    ,$CRLF
141116                          20$:
141116  122737  000001  001220          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
141124  001007                          BNE     2$              ;;NO,SKIP APT ERROR REPORT
141126  113737  001114  141140          MOVB    $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
141134  004737  141376                  JSR     PC,$ATY4        ;;REPORT FATAL ERROR TO APT
141140     000                  21$:    .BYTE   0
141141     000                          .BYTE   0
141142  000777                  22$:    BR      22$             ;;APT ERROR LOOP
141144  005777  037770          2$:     TST     @SWR            ;;HALT ON ERROR
141150  100002                          BPL     3$              ;;SKIP IF CONTINUE
141152  000000                          HALT                    ;;HALT ON ERROR!
141154  104407                          CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
141156  032777  001000  037754  3$:     BIT     #BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
141164  001402                          BEQ     4$              ;;BR IF NO
141166  013716  001110                  MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
141172  005737  001166          4$:     TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
141176  001402                          BEQ     5$              ;;BR IF NONE
141200  013716  001166                  MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
141204                          5$:
141204  022737  140440  000042          CMP     #$ENDAD,@#42    ;;ACT-11 AUTO-ACCEPT?
141212  001001                          BNE     6$              ;;BRANCH IF NO
141214  000000                          HALT                    ;;YES
141216                          6$:
141216  000002                          RTI                     ;;RETURN
                                .SBTTL  ABORT ROUTINE FOR LCP/ORION UFD MODE
141220  005737  004116          ABORT:  TST     UFDFLG          ;TEST FOR USER FRIENDLY MODE
141224  001454                          BEQ     NOABRT          ;IF NOT UFD THEN CONTINUE NORMAL OPERATION
141226  020027  000032                  CMP     R0,#32          ;IS IT A ↑Z ?
```

ABORT ROUTINE FOR LCP/ORION UFD MODE

```
141232  001443                          BEQ     ABORTZ          ;JUST GO BACK TO CHAIN IF IT IS (NO ERROR)
141234  020027  000003                   CMP     R0,#3           ;IS IS A ↑C ?
141240  001404                          BEQ     ABORTC          ;BR TO LOAD ↑C ON XXDP+ STACK (NO ERROR)
141242  005737  004120                   TST     UQUIET          ;TEST FOR USER-QUIET MODE
141246  001443                          BEQ     NOABRT          ;IF FIELD-SERVICE MODE, CONTINUE NORMAL OPERATION
                                                                ; BECAUSE FIELD-SERVICE MODE DOES NOT QUIT ON ERROR
141250  000422                          BR      ABORTE          ;SET DRSERR THEN LEAVE
141252  013737  004112  000030  ABORTC: MOV     SAV30,30        ;RESTORE EMT LOCATION (30)
141260  013737  004114  000032          MOV     SAV32,32        ;RESTORE EMT PRIORITY LOCATION (32)
141266  104043                          EMT     +43             ;GET XXDP STACK LOC. INTO R0 FROM MONITOR
141270  005720                  1$:     TST     (R0)+           ;FIND END OF STACK
141272  001376                          BNE     1$
141274  112760  000057  177777          MOVB    #'/,-1(R0)      ;LOAD SLASH OVER ZERO
141302  112720  000136                  MOVB    #'↑,(R0)+       ;LOAD UPARROW
141306  112720  000103                  MOVB    #'C,(R0)+       ;LOAD C
141312  105010                          CLRB    (R0)            ;MAKE NEW END TO STACK
141314  000412                          BR      ABORTZ          ;NOW LEAVE
141316  013737  004112  000030  ABORTE: MOV     SAV30,30        ;RESTORE EMT LOCATION (30)
141324  013737  004114  000032          MOV     SAV32,32        ;RESTORE EMT PRIORITY LOCATION (32)
141332  104042                          EMT     +42             ;GET DCA LOCATION INTO R0 FROM MONITOR
141334  012760  177777  000042          MOV     #-1,42(R0)      ;SET A -1 INTO LOCATION DRSERR IN MONITOR
141342  013700  000042  ABORTZ: MOV     @#42,R0         ;AND PUT THE MONITOR RETURN ADDRESS IN R0
141346  005037  000042          CLR     @#42            ;CLEAR MONITOR RETURN FLAG
141352  000137  140440          JMP     $ENDAD          ;RETURN TO MONITOR-DO NOT PUSH STACK HERE
141356  000207          NOABRT: RTS     PC              ;IF NOTUFD RETURN TO MAINLINE
 21851                  .SBTTL  APT COMMUNICATIONS ROUTINE
                        ;;**************************************************************
141360  112737  000001  141624  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
141366  112737  000001  141622  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
141374  000403                          BR      $ATYC
141376  112737  000001  141624  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
141404                          $ATYC:
141404  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
141406  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
141410  105737  141622                  TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
141414  001450                          BEQ     5$              ;;IF NOT:  BR
141416  122737  000001  001220          CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
141424  001031                          BNE     3$              ;;IF NOT:  BR
141426  132737  000100  001221          BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
141434  001425                          BEQ     3$              ;;IF NOT:  BR
141436  017600  000004                  MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
141442  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
141450  005737  001200          1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
141454  001375                          BNE     1$              ;;IF NOT:  WAIT
141456  010037  001214                  MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
141462  105720                  2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
141464  001376                          BNE     2$
141466  163700  001214                  SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
141472  006200                          ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
141474  010037  001216                  MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
141500  012737  000004  001200          MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
141506  000413                          BR      5$
141510  017637  000004  141534  3$:     MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
141516  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
141524  013746  177776                  MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
141530  004737  141626                  JSR     PC,$TYPE        ;;CALL TYPE MACRO
141534  000000                  4$:     .WORD   0
```

APT COMMUNICATIONS ROUTINE

```
141536                                  5$:
141536  105737  141624                 10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
141542  001416                                  BEQ     12$             ;;IF NOT:  BR
141544  005737  001220                          TST     $ENV            ;;RUNNING UNDER APT?
141550  001413                                  BEQ     12$             ;;IF NOT:  BR
141552  005737  001200                 11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
141556  001375                                  BNE     11$             ;;IF NOT:  WAIT
141560  017637  000004  001202                 MOV     @4(SP),$FATAL   ;;GET ERROR #
141566  062766  000002  000004                 ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
141574  005237  001200                          INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
141600  105037  141624                 12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
141604  105037  141623                         CLRB    $LFLG           ;;CLEAR LOG FLAG
141610  105037  141622                         CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
141614  012601                                  MOV     (SP)+,R1        ;;POP STACK INTO R1
141616  012600                                  MOV     (SP)+,R0        ;;POP STACK INTO R0
141620  000207                                  RTS     PC              ;;RETURN
141622     000                  $MFLG:  .BYTE   0               ;;MESSG. FLAG
141623     000                  $LFLG:  .BYTE   0               ;;LOG FLAG
141624     000                  $FFLG:  .BYTE   0               ;;FATAL FLAG
                                        .EVEN
          000200                APTSIZE=200
          000001                APTENV=001
          000100                APTSPOOL=100
          000040                APTCSUP=040
21852                           .SBTTL  TYPE ROUTINE
                                ;;**************************************************************
                                ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
                                ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
                                ;*NOTE1:         $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
                                ;*NOTE2:         $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
                                ;*NOTE3:         $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
                                ;*
                                ;*CALL:
                                ;*1) USING A TRAP INSTRUCTION
                                ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
                                ;*OR
                                ;*      TYPE
                                ;*      MESADR
                                ;*
141626  105737  001157          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
141632  100002                          BPL     1$              ;;BR IF YES
141634  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
141636  000430                          BR      3$              ;;LEAVE
141640  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
141642  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
141646  122737  000001  001220          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
141654  001011                          BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
141656  132737  000100  001221          BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
141664  001405                          BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
141666  010037  141676                  MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
141672  004737  141366                  JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
141676  000000                  61$:    .WORD   0               ;;MESSAGE ADDRESS
141700  132737  000040  001221  62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
141706  001003                          BNE     60$             ;;YES,SKIP TYPE OUT
141710  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
141712  001005                          BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
141714  005726                          TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
```

D3

SEQ 0443

TYPE ROUTINE

```
141716  012600                  60$:    MOV     (SP)+,R0        ;;RESTORE R0
141720  062716  000002            3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
141724  000002                          RTI                     ;;RETURN
141726  122716  000011            4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
141732  001430                          BEQ     8$
141734  122716  000200                  CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
141740  001006                          BNE     5$
141742  005726                          TST     (SP)+           ;;POP  <CR><LF> EQUIV
141744  104401                          TYPE                    ;;TYPE A CR AND LF
141746  001175                          $CRLF
141750  105037  142156                  CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
141754  000755                          BR      2$              ;;GET NEXT CHARACTER
141756  004737  142040            5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
141762  123726  001156            6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
141766  001350                          BNE     2$              ;;IF NO GO GET NEXT CHAR.
141770  013746  001154                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
                                                                ;;AND THE NULL CHAR.
141774  105366  000001            7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
142000  002770                          BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
142002  004737  142040                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
142006  105337  142156                  DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
142012  000770                          BR      7$              ;;LOOP
                                ;HORIZONTAL TAB PROCESSOR
142014  112716  000040            8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
142020  004737  142040            9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
142024  132737  000007  142156            BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
142032  001372                          BNE     9$              ;;TAB STOP
142034  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
142036  000724                          BR      2$              ;;GET NEXT CHARACTER
142040                          $TYPEC:
142040  105777  037100                  TSTB    @$TKS           ;;CHAR IN KYBD BUFFER?          ;MJD001
142044  100022                          BPL     10$             ;;BR IF NOT                     ;MJD001
142046  017746  037074                  MOV     @$TKB,-(SP)     ;;GET CHAR                      ;MJD001
142052  042716  177600                  BIC     #177600,(SP)    ;;STRIP EXTRANEOUS BITS         ;MJD001
142056  122716  000023                  CMPB    #$XOFF,(SP)     ;;WAS CHAR XOFF                 ;MJD001
142062  001012                          BNE     102$            ;;BR IF NOT                     ;MJD001
142064                          101$:                                                           ;MJD001
142064  105777  037054                  TSTB    @$TKS           ;;WAIT FOR CHAR                 ;MJD001
142070  100375                          BPL     101$                                            ;MJD001
142072  117716  037050                  MOVB    @$TKB,(SP)      ;;GET CHAR                      ;MJD001
142076  042716  177600                  BIC     #177600,(SP)    ;;STRIP IT                      ;MJD001
142102  122716  000021                  CMPB    #$XON,(SP)      ;;WAS IT XON?                   ;MJD001
142106  001366                          BNE     101$            ;;BR IF NOT                     ;MJD001
142110                          102$:                                                           ;MJD001
142110  005726                          TST     (SP)+           ;;FIX STACK                     ;MJD001
142112                          10$:                                                            ;MJD001
142112  105777  037032                  TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
142116  100375                          BPL     10$                                             ;MJD001
142120  116677  000002  037024          MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
142126  122766  000015  000002          CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
142134  001003                          BNE     1$              ;;BRANCH IF NO
142136  105037  142156                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
142142  000406                          BR      $TYPEX          ;;EXIT
142144  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
142152  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
142154  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
142156  000000                  $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
```

TYPE ROUTINE

```
142160  000207                  $TYPEX: RTS     PC
 21853
                                .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
                                ;;******************************************************************
                                ;*THIS ROUTINE IS USED TO CHANGE A 16 BIT BINARY NUMBER TO A 6 DIGIT
                                ;*OCTAL (ASCII) NUMBER AND TYPE IT.
                                ;*$TYPOS-  ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
                                ;*CALL:
                                ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                                ;*      TYPOS                   ;;CALL FOR TYPEOUT
                                ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
                                ;*      .BYTE   M               ;;M=1 OR 0
                                ;*                                      ;;1=TYPE LEADING ZEROS
                                ;*                                      ;;0=SUPPRESS LEADING ZEROS
                                ;*
                                ;*$TYPON ---ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
                                ;*$TYPOS OR $TYPOC
                                ;*CALL:
                                ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                                ;*      TYPON                   ;;CALL FOR TYPEOUT
                                ;*
                                ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
                                ;*CALL:
                                ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                                ;*      TYPOC                   ;;CALL FOR TYPEOUT
142162  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
142166  116637  000001  142405          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
142174  112637  142407                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
142200  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
142204  000406                          BR      $TYPON
142206  112737  000001  142405  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
142214  112737  000006  142407          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
142222  112737  000005  142404  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
142230  010346                          MOV     R3,-(SP)        ;;SAVE R3
142232  010446                          MOV     R4,-(SP)        ;;SAVE R4
142234  010546                          MOV     R5,-(SP)        ;;SAVE R5
142236  113704  142407                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
142242  005404                          NEG     R4
142244  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
142250  110437  142406                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
142254  113704  142405                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
142260  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
142264  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
142266  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
142270  000404                          BR      3$              ;;GO DO MSB
142272  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
142274  006105                          ROL     R5
142276  006105                          ROL     R5
142300  010503                          MOV     R5,R3
142302  006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
142304  105337  142406                  DECB    $OMODE          ;;TYPE THIS DIGIT?
142310  100016                          BPL     7$              ;;BR IF NO
142312  042703  177770                  BIC     #177770,R3      ;;GET RID OF JUNK
142316  001002                          BNE     4$              ;;TEST FOR 0
142320  005704                          TST     R4              ;;SUPPRESS THIS 0?
142322  001403                          BEQ     5$              ;;BR IF YES
142324  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
142326  052703  000060                  BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
```

F3

SEQ 0445

BINARY TO OCTAL (ASCII) AND TYPE

```
142332  052703  000040          5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
142336  110337  142402                  MOVB    R3,8$           ;;SAVE FOR TYPING
142342  104401  142402                  TYPE    ,8$             ;;GO TYPE THIS DIGIT
142346  105337  142404          7$:     DECB    $OCNT           ;;COUNT BY 1
142352  003347                          BGT     2$              ;;BR IF MORE TO DO
142354  002402                          BLT     6$              ;;BR IF DONE
142356  005204                          INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
142360  000744                          BR      2$              ;;GO DO THE LAST DIGIT
142362  012605                  6$:     MOV     (SP)+,R5        ;;RESTORE R5
142364  012604                          MOV     (SP)+,R4        ;;RESTORE R4
142366  012603                          MOV     (SP)+,R3        ;;RESTORE R3
142370  016666  000002  000004          MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
142376  012616                          MOV     (SP)+,(SP)
142400  000002                          RTI                     ;;RETURN
142402     000                  8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
142403     000                          .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
142404     000                  $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
142405     000                  $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
142406  000000                  $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
 21854                          .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
                                ;;**************************************************************
                                ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
                                ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
                                ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
                                ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
                                ;*REPLACED WITH SPACES.
                                ;*CALL:
                                ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
                                ;*      TYPDS                   ;;GO TO THE ROUTINE
142410                          $TYPDS:
142410  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
142412  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
142414  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
142416  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
142420  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
142422  012746  020200                  MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
142426  016605  000020                  MOV     20(SP),R5       ;;GET THE INPUT NUMBER
142432  100004                          BPL     1$              ;;BR IF INPUT IS POS.
142434  005405                          NEG     R5              ;;MAKE THE BINARY NUMBER POS.
142436  112766  000055  000001          MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
142444  005000                  1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
142446  012703  142624                  MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
142452  112723  000040                  MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
142456  005002                  2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
142460  016001  142614                  MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
142464  160105                  3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
142466  002402                          BLT     4$              ;;BR IF DONE
142470  005202                          INC     R2              ;;INCREASE THE BCD DIGIT BY 1
142472  000774                          BR      3$
142474  060105                  4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
142476  005702                          TST     R2              ;;CHECK IF BCD DIGIT=0
142500  001002                          BNE     5$              ;;FALL THROUGH IF 0
142502  105716                          TSTB    (SP)            ;;STILL DOING LEADING 0'S?
142504  100407                          BMI     7$              ;;BR IF YES
142506  106316                  5$:     ASLB    (SP)            ;;MSD?
142510  103003                          BCC     6$              ;;BR IF NO
142512  116663  000001  177777          MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
```

CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

```
142520  052702  000060          6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
142524  052702  000040          7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
142530  110223                          MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
142532  005720                          TST     (R0)+           ;;JUST INCREMENTING
142534  020027  000010                  CMP     R0,#10          ;;CHECK THE TABLE INDEX
142540  002746                          BLT     2$              ;;GO DO THE NEXT DIGIT
142542  003002                          BGT     8$              ;;GO TO EXIT
142544  010502                          MOV     R5,R2           ;;GET THE LSD
142546  000764                          BR      6$              ;;GO CHANGE TO ASCII
142550  105726                  8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
142552  100003                          BPL     9$              ;;BR IF NO
142554  116663  177777  177776        MOVB    -1(SP),-2(R3)   ;;YES- SET THE SIGN FOR TYPING
142562  105013                  9$:     CLRB    (R3)            ;;SET THE TERMINATOR
142564  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
142566  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
142570  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
142572  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
142574  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
142576  104401  142624                  TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
142602  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
142610  012616                          MOV     (SP)+,(SP)
142612  000002                          RTI                     ;;RETURN TO USER
142614  023420                  $DTBL:  10000.
142616  001750                          1000.
142620  000144                          100.
142622  000012                          10.
142624                          $DBLK:  .BLKW   4
 21855                          .SBTTL  TTY INPUT ROUTINE
                                ;;**************************************************************
                                .ENABL  LSB
                                ;;**************************************************************
                                ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
                                ;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
                                ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
                                ;*WHEN OPERATING IN TTY FLAG MODE.
142634  022737  000176  001140  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED?
142642  001074                          BNE     15$             ;;BRANCH IF NO
142644  105777  036274                  TSTB    @$TKS           ;;CHAR THERE?
142650  100071                          BPL     15$             ;;IF NO, DON'T WAIT AROUND
142652  117746  036270                  MOVB    @$TKB,-(SP)     ;;SAVE THE CHAR
142656  042716  177600                  BIC     #^C177,(SP)     ;;STRIP-OFF THE ASCII
142662  022726  000007                  CMP     #7,(SP)+        ;;IS IT A CONTROL G?
142666  001062                          BNE     15$             ;;NO, RETURN TO USER
142670  123727  001134  000001          CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
142676  001456                          BEQ     15$             ;;BRANCH IF YES
142700  104401  143371                  TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
142704  104401  143376          $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
142710  013746  000176                  MOV     SWREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
142714  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
142716  104401  143407                  TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
142722  005046                  19$:    CLR     -(SP)           ;;CLEAR COUNTER
142724  005046                          CLR     -(SP)           ;;THE NEW SWR
142726  105777  036212          7$:     TSTB    @$TKS           ;;CHAR THERE?
142732  100375                          BPL     7$              ;;IF NOT TRY AGAIN
142734  117746  036206                  MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
142740  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
142744  021627  000025          9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
```

TTY INPUT ROUTINE

```
142750  001005                          BNE     10$             ;;BRANCH IF NOT
142752  104401  143364                  TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (↑U)
142756  062706  000006          20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
142762  000757                          BR      19$             ;;LET'S TRY IT AGAIN
142764  021627  000015          10$:    CMP     (SP),#15        ;;IS IT A <CR>?
142770  001022                          BNE     16$             ;;BRANCH IF NO
142772  005766  000004                  TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
142776  001403                          BEQ     11$             ;;BRANCH IF YES
143000  016677  000002  036132          MOV     2(SP),@SWR      ;;SAVE NEW SWR
143006  062706  000006          11$:    ADD     #6,SP           ;;CLEAR UP STACK
143012  104401  001175          14$:    TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
143016  123727  001135  000001          CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
143024  001003                          BNE     15$             ;;BRANCH IF NOT
143026  012777  000100  036110          MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
143034  000002          15$:    RTI                             ;;RETURN
143036  004737  142040          16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
143042  021627  000060                  CMP     (SP),#60        ;;CHAR < 0?
143046  002420                          BLT     18$             ;;BRANCH IF YES
143050  021627  000067                  CMP     (SP),#67        ;;CHAR > 7?
143054  003015                          BGT     18$             ;;BRANCH IF YES
143056  042726  000060                  BIC     #60,(SP)+       ;;STRIP-OFF ASCII
143062  005766  000002                  TST     2(SP)           ;;IS THIS THE FIRST CHAR
143066  001403                          BEQ     17$             ;;BRANCH IF YES
143070  006316                          ASL     (SP)            ;;NO, SHIFT PRESENT
143072  006316                          ASL     (SP)            ;;   CHAR OVER TO MAKE
143074  006316                          ASL     (SP)            ;;   ROOM FOR NEW ONE.
143076  005266  000002          17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
143102  056616  177776                  BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
143106  000707                          BR      7$              ;;GET THE NEXT ONE
143110  104401  001174          18$:    TYPE    ,$QUES          ;;TYPE ?<CR><LF>
143114  000720                          BR      20$             ;;SIMULATE CONTROL-U
                        .DSABL  LSB
                        ;;*****************************************************************
                        ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
                        ;*CALL:
                        ;*      RDCHR                           ;;INPUT A SINGLE CHARACTER FROM THE TTY
                        ;*      RETURN HERE                     ;;CHARACTER IS ON THE STACK
                        ;*                                      ;;WITH PARITY BIT STRIPPED OFF
                        ;
143116  011646          $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
143120  016666  000004  000002          MOV     4(SP),2(SP)     ;;SAVE THE PS
143126  105777  036012          1$:     TSTB    @$TKS           ;;WAIT FOR
143132  100375                          BPL     1$              ;;A CHARACTER
143134  117766  036006  000004          MOVB    @$TKB,4(SP)     ;;READ THE TTY
143142  042766  177600  000004          BIC     #↑C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
143150  026627  000004  000023          CMP     4(SP),#23       ;;IS IT A CONTROL-S?
143156  001013                          BNE     3$              ;;BRANCH IF NO
143160  105777  035760          2$:     TSTB    @$TKS           ;;WAIT FOR A CHARACTER
143164  100375                          BPL     2$              ;;LOOP UNTIL ITS THERE
143166  117746  035754                  MOVB    @$TKB,-(SP)     ;;GET CHARACTER
143172  042716  177600                  BIC     #↑C177,(SP)     ;;MAKE IT 7-BIT ASCII
143176  022627  000021                  CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
143202  001366                          BNE     2$              ;;IF NOT DISCARD IT
143204  000750                          BR      1$              ;;YES, RESUME
143206  026627  000004  000021  3$:     CMP     4(SP),#$XON     ;;IS IT A RANDOM XON?                   ;RAN001
143214  001744                          BEQ     1$              ;;BRANCH IF YES                         ;RAN001
143216  026627  000004  000140          CMP     4(SP),#140      ;;IS IT UPPER CASE?
```

TTY INPUT ROUTINE

```
143224  002407                          BLT     4$                      ;;BRANCH IF YES
143226  026627  000004  000175          CMP     4(SP),#175              ;;IS IT A SPECIAL CHAR?
143234  003003                          BGT     4$                      ;;BRANCH IF YES
143236  042766  000040  000004          BIC     #40,4(SP)               ;;MAKE IT UPPER CASE
143244  000002                  4$:     RTI                             ;;GO BACK TO USER
                                ;;**************************************************************
                                ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
                                ;*CALL:
                                ;*      RDLIN                           ;;INPUT A STRING FROM THE TTY
                                ;*      RETURN HERE                     ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
                                ;*                                      ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
143246  010346                  $RDLIN: MOV     R3,-(SP)                ;;SAVE R3
143250  012703  143354          1$:     MOV     #$TTYIN,R3              ;;GET ADDRESS
143254  022703  143364          2$:     CMP     #$TTYIN+8.,R3           ;;BUFFER FULL?
143260  101405                          BLOS    4$                      ;;BR IF YES
143262  104410                          RDCHR                           ;;GO READ ONE CHARACTER FROM THE TTY
143264  112613                          MOVB    (SP)+,(R3)              ;;GET CHARACTER
143266  122713  000177          10$:    CMPB    #177,(R3)               ;;IS IT A RUBOUT
143272  001003                          BNE     3$                      ;;SKIP IF NOT
143274  104401  001174          4$:     TYPE    ,$QUES                  ;;TYPE A '?'
143300  000763                          BR      1$                      ;;CLEAR THE BUFFER AND LOOP
143302  111337  143352          3$:     MOVB    (R3),9$                 ;;ECHO THE CHARACTER
143306  104401  143352                  TYPE    ,9$
143312  122723  000015                  CMPB    #15,(R3)+               ;;CHECK FOR RETURN
143316  001356                          BNE     2$                      ;;LOOP IF NOT RETURN
143320  105063  177777                  CLRB    -1(R3)                  ;;CLEAR RETURN (THE 15)
143324  104401  001176                  TYPE    ,$LF                    ;;TYPE A LINE FEED
143330  012603                          MOV     (SP)+,R3                ;;RESTORE R3
143332  011646                          MOV     (SP),-(SP)              ;;ADJUST THE STACK AND PUT ADDRESS OF THE
143334  016666  000004  000002          MOV     4(SP),2(SP)             ;;      FIRST ASCII CHARACTER ON IT
143342  012766  143354  000004          MOV     #$TTYIN,4(SP)
143350  000002                          RTI                             ;;RETURN
143352     000                  9$:     .BYTE   0                       ;;STORAGE FOR ASCII CHAR. TO TYPE
143353     000                          .BYTE   0                       ;;TERMINATOR
143354                          $TTYIN: .BLKB   8.                      ;;RESERVE 8 BYTES FOR TTY INPUT
143364     136     125     015  $CNTLU: .ASCIZ  /^U/<15><12>            ;;CONTROL "U"
143367     012     000
143371     136     107     015  $CNTLG: .ASCIZ  /^G/<15><12>            ;;CONTROL "G"
143374     012     000
143376     015     012     123  $MSWR:  .ASCIZ  <15><12>/SWR = /
143401     127     122     040
143404     075     040     000
143407     040     040     116  $MNEW:  .ASCIZ  /  NEW = /
143412     105     127     040
143415     075     040     000
21856                           .SBTTL  READ AN OCTAL NUMBER FROM THE TTY
                                ;;**************************************************************
                                ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
                                ;*CHANGE IT TO BINARY.
                                ;*CALL:
                                ;*      RDOCT                           ;;READ AN OCTAL NUMBER
                                ;*      RETURN HERE                     ;;LOW ORDER BITS ARE ON TOP OF THE STACK
                                ;*                                      ;;HIGH ORDER BITS ARE IN $HIOCT
143420  011646                  $RDOCT: MOV     (SP),-(SP)              ;;PROVIDE SPACE FOR THE
143422  016666  000004  000002          MOV     4(SP),2(SP)             ;;INPUT NUMBER
143430  010046                          MOV     R0,-(SP)                ;;PUSH R0 ON STACK
143432  010146                          MOV     R1,-(SP)                ;;PUSH R1 ON STACK
```

J3

SEQ 0449

READ AN OCTAL NUMBER FROM THE TTY

```
143434  010246                          MOV     R2, (SP)        ;;PUSH R2 ON STACK
143436  104411                  1$:     RDLIN                   ;;READ AN ASCIZ LINE
143440  012600                          MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
143442  005001                          CLR     R1              ;;CLEAR DATA WORD
143444  005002                          CLR     R2
143446  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
143450  001412                          BEQ     3$              ;;IF ZERO GET OUT
143452  006301                          ASL     R1              ;;*2
143454  006102                          ROL     R2
143456  006301                          ASL     R1              ;;*4
143460  006102                          ROL     R2
143462  006301                          ASL     R1              ;;*8
143464  006102                          ROL     R2
143466  042716  177770                  BIC     #^C7,(SP)       ;;STRIP THE ASCII JUNK
143472  062601                          ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
143474  000764                          BR      2$              ;;LOOP
143476  005726                  3$:     TST     (SP)+           ;;CLEAN TERMINATOR FROM STACK
143500  010166  000012                  MOV     R1,12(SP)       ;;SAVE THE RESULT
143504  010237  143520                  MOV     R2,$HIOCT
143510  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
143512  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
143514  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
143516  000002                          RTI                     ;;RETURN
143520  000000                  $HIOCT: .WORD   0               ;;HIGH ORDER BITS GO HERE
 21857                          .SBTTL  TRAP DECODER
                                ;;******************************************************************
                                ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
                                ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
                                ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
                                ;*GO TO THAT ROUTINE.
143522  010046                  $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
143524  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
143530  005740                          TST     -(R0)           ;;BACKUP BY 2
143532  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
143534  006300                          ASL     R0              ;;POSITION FOR INDEXING
143536  016000  143556                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
143542  000200                          RTS     R0              ;;GO TO ROUTINE
                                ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
143544  011646                  $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
143546  016666  000004  000002          MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
143554  000002                          RTI                     ;;RESTORE THE PSW
                                .SBTTL  TRAP TABLE
                                ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
                                ;*BY THE "TRAP" INSTRUCTION.
                                ;       ROUTINE
                                ;       -------
143556  143544                  $TRPAD: .WORD   $TRAP2
143560  141626                          $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
143562  142206                          $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
143564  142162                          $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
143566  142222                          $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
143570  142410                          $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
143572  142704                          $GTSWR  ;;CALL=GTSWR    TRAP+6(104406)  GET SOFT-SWR SETTING
143574  142634                          $CKSWR  ;;CALL=CKSWR    TRAP+7(104407)  TEST FOR CHANGE IN SOFT-SWR
143576  143116                          $RDCHR  ;;CALL=RDCHR    TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
143600  143246                          $RDLIN  ;;CALL=RDLIN    TRAP+11(104411) TTY TYPEIN STRING ROUTINE
143602  143420                          $RDOCT  ;;CALL=RDOCT    TRAP+12(104412) READ AN OCTAL NUMBER FROM TTY
```

POWER DOWN AND UP ROUTINES

```
21858                                   .SBTTL  POWER DOWN AND UP ROUTINES
                                        ;;*****************************************************************
                                        ;POWER DOWN ROUTINE
143604  012737  143744  000024    $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
143612  012737  000340  000026            MOV     #340,@#PWRVEC+2 ;;PRIO:7
143620  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
143622  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
143624  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
143626  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
143630  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
143632  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
143634  017746  035300                  MOV     @SWR,-(SP)      ;;PUSH @SWR ON STACK
143640  010637  143750                  MOV     SP,$SAVR6       ;;SAVE SP
143644  012737  143656  000024          MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
143652  000000                          HALT
143654  000776                          BR      .-2             ;;HANG UP
                                        ;;*****************************************************************
                                        ;POWER UP ROUTINE
143656  012737  143744  000024    $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
143664  013706  143750                  MOV     $SAVR6,SP       ;;GET SP
143670  005037  143750                  CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
143674  005237  143750          1$:     INC     $SAVR6          ;;WAIT FOR THE INC
143700  001375                          BNE     1$              ;;OF  WORD
143702  012677  035232                  MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
143706  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
143710  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
143712  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
143714  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
143716  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
143720  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
143722  012737  143604  000024          MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
143730  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
143736  104401                          TYPE                    ;REPORT THE POWER FAILURE
143740  143752                  $PWRMG: .WORD   $POWER          ;;POWER FAIL MESSAGE POINTER
143742  000002                          RTI
143744  000000                  $ILLUP: HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
143746  000776                          BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
143750  000000                  $SAVR6: 0                       ;;PUT THE SP HERE
143752     015     012     120  $POWER: .ASCIZ  <15><12>"POWER"
143755     117     127     105
143760     122     000
                                        .EVEN
21860           000001                  .END
```

L3

Symbol table

```
ABASE = 000000        AMADR3= 000000        BIT14  = 040000        DATBO   124574        D4      052760
ABOEXT  111314        AMADR4= 000000        BIT15  = 100000        DATI    125124        D5      052774
ABORT   141220        AMAMS1= 000000        BIT2   = 000004        DATVER  140100        D6      053004
ABORTC  141252        AMAMS2= 000000        BIT3   = 000010        DATVFR  140062        D7      053010
ABORTE  141316        AMAMS3= 000000        BIT4   = 000020        DAT1    140112        EEPAS   003034
ABORTI  107634        AMAMS4= 000000        BIT5   = 000040        DCOUNT  114144        EMTOA   030464
ABORTR  107676        AMSGAD= 000000        BIT6   = 000100        DDISP = 177570        EMTOB   030472
ABORTZ  141342        AMSGLG= 000000        BIT7   = 000200        DELAY   116754        EMTSAV  004024
ABORT0  044016        AMSGTY= 000000        BIT8   = 000400        DETFPA  140132        EMTVEC= 000030
ABORT7  044204        AMTYP1= 000000        BIT9   = 001000        DH1     134621        EM1     127002
ABROUT  111146        AMTYP2= 000000        BN0      043746        DH105   135562        EM10    127254
ACDW1 = 000000        AMTYP3= 000000        BN1      044136        DH115   135614        EM100   132570
ACDW2 = 000000        AMTYP4= 000000        BPTOA    030660        DH134   135671        EM101   132626
ACPUOP= 000000        APASS = 000000        BPTOB    030666        DH24    135105        EM102   132661
ACTCHS  002724        APRIOR= 000000        BPTVEC=  000014        DH27    135151        EM103   132743
AC0   =%000000        APTCSU= 000040        BTER     051362        DH4     134646        EM104   133016
AC1   =%000001        APTENV= 000001        BTEXP    003100        DH41    135214        EM105   133066
AC2   =%000002        APTSIZ= 000200        BTGO     051162        DH43    135265        EM106   133141
AC3   =%000003        APTSPO= 000100        BTRES    003110        DH47    135365        EM107   133217
AC4   =%000004        ASWREG= 000000        BTTST    051662        DH5     134733        EM11    127275
AC5   =%000005        ATESTN= 000000        BTTSTE   051710        DH65    135452        EM110   133253
AC6   =%000006        AUNIT = 000000        BYPARR   106114        DH7     135032        EM111   133315
AC7   =%000007        AUSWR = 000000        CCHPAS   003032        DH72    135516        EM112   133373
ADDLSB  114154        AVECT1= 000000        CCR    = 177746        DISPLA  001142        EM113   133456
ADDT    052264        AVECT2= 000000        CCRTBL   111106        DISPRE  000174        EM114   133542
ADDTRP  137720        A126    025246        CHECK    015342        DMAPAR  124040        EM115   133563
ADDW0 = 000000        BA      002670        CHECK1   015420        DMARD   137452        EM116   133633
ADDW1 = 000000        BACDAT  105606        CHECK2   015476        DMATRN  137370        EM117   133714
ADDW10= 000000        BA2     002710        CHECK7   053606        DPAREN  124100        EM12    127317
ADDW11= 000000        BCR    = 177524        CHEC10   054032        DSWR   = 177570        EM120   133757
ADDW12= 000000        BCSR   = 177520        CHEC26   057214        DT1     136026        EM121   134025
ADDW13= 000000        BDR    = 177524        CHEC27   057556        DT105   136304        EM122   134123
ADDW14= 000000        BFA     052316        CHEC30   060122        DT115   136314        EM123   134151
ADDW15= 000000        BFAC1   052060        CHEC32   060554        DT130   136330        EM124   134203
ADDW2 = 000000        BFAC2   052104        CHEC33   061050        DT134   136336        EM125   134275
ADDW3 = 000000        BFAC3   052130        CHEK7    053612        DT14    136074        EM126   134334
ADDW4 = 000000        BFAC4   052154        CHK10    054050        DT17    136106        EM127   134371
ADDW5 = 000000        BFAC5   052202        CHK7     053620        DT24    136120        EM13    127345
ADDW6 = 000000        BFAE    052356        CH10     054036        DT27    136130        EM130   134433
ADDW7 = 000000        BFB     052354        CKSWR  = 104407        DT35    136142        EM131   134460
ADDW8 = 000000        BGNTLP  113472        CMPRTN   066176        DT4     136034        EM132   134510
ADDW9 = 000000        BIT0   = 000001        COUNT    003120        DT41    136154        EM133   134546
ADEVCT= 000000        BIT00  = 000001        CPEREG=  177766        DT43    136164        EM134   134567
ADEVM = 000000        BIT01  = 000002        CR     = 000015        DT47    136200        EM14    127374
AENV  = 000000        BIT02  = 000004        CRLF   = 000200        DT5     136046        EM15    127417
AENVM = 000000        BIT03  = 000010        CSR1     002664        DT50    136212        EM16    127453
AFATAL= 000000        BIT04  = 000020        CSR12    002704        DT51    136224        EM17    127520
ALLCTR  003152        BIT05  = 000040        CSR2     002666        DT52    136236        EM2     127036
ALLEND  125304        BIT06  = 000100        CSR22    002706        DT64    136250        EM20    127564
ALR0TS  021532        BIT07  = 000200        CURADD   114170        DT65    136262        EM21    127617
ALR1TS  021610        BIT08  = 000400        CURDAT   114162        DT7     136062        EM22    127662
ALR2TS  021666        BIT09  = 001000        C121A    023262        DT75    136272        EM23    127721
ALR3TS  021744        BIT1   = 000002        C121B    023302        DVDSUB  070236        EM24    127775
ALR4TS  022022        BIT10  = 002000        C121C    023322        DVFSUB  067244        EM25    130040
ALR5TS  022100        BIT11  = 004000        DAPABO   107350        D1      052726        EM26    130100
AMADR1= 000000        BIT12  = 010000        DATA     002674        D2      052740        EM27    130147
AMADR2= 000000        BIT13  = 020000        DATA2    002714        D3      052742        EM3     127050
```

M3

Symbol table

EM30	130207	EXBAD2	115266	GPR4TS	005374	KIPAR4=	172350	MBTCC	015614		
EM31	130261	EXITST	114134	GPR5TS	005452	KIPAR5=	172352	MBTD	032760		
EM32	130306	EXPBDT	111126	GPR6TS	005530	KIPAR6=	172354	MBTE	051712		
EM33	130347	EXPDAT	114146	GTSWR =	104406	KIPAR7=	172356	MBTF	032770		
EM34	130411	EXPIR1	034344	HITMIS=	177752	KIPDR0=	172300	MBTO	033002		
EM35	130451	EXPTBL	105634	HMPARR	105404	KIPDR1=	172302	MBTOA	033032		
EM36	130477	EXPWDT	111116	HOP10	067506	KIPDR2=	172304	MBTOB	033034		
EM37	130543	EXTFPA	140172	HOP11	070500	KIPDR3=	172306	MBTOC	033046		
EM4	127062	FINNOP	021252	HOP12	071536	KIPDR4=	172310	MBTOD	033072		
EM40	130605	FIN1	052444	HOP13	072564	KIPDR5=	172312	MBTOE	033074		
EM41	130652	FIN10	054114	HOP14	073660	KIPDR6=	172314	MBTOF	033106		
EM42	130736	FIN11	054220	HOP15	075354	KIPDR7=	172316	MBT1	051140		
EM43	130764	FIN116	022742	HOP16	075670	KMCR =	177734	MBT2	051142		
EM44	131015	FIN117	023056	HOP17	076352	LASTCH	116110	MBT2A	051154		
EM45	131055	FIN120	023200	HOP18	100612	LCDSUB	101102	MBT8	051370		
EM46	131137	FIN121	023574	HOP19	101230	LCFSUB	100464	MBT8A	051424		
EM47	131227	FIN122	024044	HOP20	102650	LDPARS	136736	MBT8B	051452		
EM5	127122	FIN125	025206	HOP21	103734	LDPDRS	136766	MBT8C	051500		
EM50	131314	FIN126	026246	HOP22	104376	LEDS	123016	MBT8D	051526		
EM51	131337	FIN127	026706	HOP44	066324	LF =	000012	MBT8E	051556		
EM52	131371	FIN13	054434	HT =	000011	LKS =	177546	MBT8F	051606		
EM53	131427	FIN130	027576	ILAOA	030760	LKSFL	002722	MBT8FG	051622		
EM54	131463	FIN14	054604	ILBOB	030766	LKSINT	137062	MBT8I	051646		
EM55	131521	FIN15	055022	ILL	053230	LOOP	004740	MB66	011770		
EM56	131555	FIN16	055212	ILLBOA	031054	LOOPIN	003154	MCB44	011634		
EM57	131601	FIN17	055424	ILLBOB	031062	LOST	052424	MCLRD	076416		
EM6	127155	FIN2	052544	ILLOP1	053030	LOWADD	003016	MCLRI	076504		
EM60	131633	FIN20	055622	ILLOP2	053124	LSTADD	114166	MCMPD	066014		
EM61	131701	FIN21	056042	INITMM	136574	LXPSUB	102400	MCTSCC	016066		
EM62	131730	FIN22	056226	INQ22	125446	MACCC	016216	MDA0	010406		
EM63	132001	FIN23	056362	INTERR	110106	MACE	051322	MDCCC	016004		
EM64	132032	FIN24	056562	INTRPC	110052	MAC0	051212	MDDSUB	075054		
EM65	132071	FIN26	057316	IOTOA	030366	MAC0A	051216	MDFSUB	073350		
EM66	132141	FIN27	057660	IOTOB	030374	MAC1	051226	MDIVD	067506		
EM67	132200	FIN30	060224	IOTVEC=	000020	MAC2	051242	MDIVF	066324		
EM7	127220	FIN31	060364	IOXXX	033520	MAC3	051256	MDM0	010362		
EM70	132235	FIN32	060656	J11FLT	140257	MAC4	051272	MDM27	010472		
EM71	132266	FIN33	061152	KDPAR0=	172360	MAC5	051306	MDS0	010436		
EM72	132322	FIN4	053134	KDPAR1=	172362	MADCC	016142	MEMK	123434		
EM73	132360	FIN5	053240	KDPAR2=	172364	MAIREG=	177750	MEMQ	123464		
EM74	132404	FIN6	053440	KDPAR3=	172366	MALCC	016770	MEMT0	030420		
EM75	132435	FIN7	053664	KDPAR4=	172370	MARCC	017050	MET	032442		
EM76	132466	FLAG	003030	KDPAR5=	172372	MASK	003160	META	032500		
EM77	132536	FLO	003062	KDPAR6=	172374	MA11	011010	METB	032510		
ENDHRT	105576	FLOAT	003052	KDPAR7=	172376	MA55	011714	METD	032520		
ENDLUP	114110	FMPARR	105214	KDPDR0=	172320	MBB11	011106	METF	032530		
ENDMOV	114172	FPAFLT	140174	KDPDR1=	172322	MBB22	011370	METO	032550		
ENDTAG	115314	FPAOPT	140160	KDPDR2=	172324	MBCCC	015664	METOA	032614		
ENDTLP	114060	FPVEC =	000244	KDPDR3=	172326	MBC00	010612	METOB	032624		
ERR	140060	FRSTST	004742	KDPDR4=	172330	MBC11	011166	METOC	032636		
ERRCNT	003022	FSTADD	114164	KDPDR5=	172332	MBC22	011456	METOD	032662		
ERRFP	140056	FWDSEQ	114152	KDPDR6=	172334	MBI00	010532	METOE	032672		
ERROR =	104000	GOODAD	003020	KDPDR7=	172336	MBPT0	030614	METOF	032704		
ERROUT	104536	GPR0TS	005104	KIPAR0=	172340	MBSCC	015736	MFA	051714		
ERRVEC=	000004	GPR1TS	005162	KIPAR1=	172342	MBT	032724	MFACU	051712		
ERTYPE	136356	GPR2TS	005240	KIPAR2=	172344	MBTA	032746	MFSRCM	061154		
EXBAD	114124	GPR3TS	005316	KIPAR3=	172346	MBTB	032750	MIALL	033320		

Symbol table

```
MIALLA  033344        MJR77A  014776        MLDC2   077644        MRTB    031550        MST6    010240
MIALLB  033346        MJSI    033400        MLDDM2  061240        MRTE    031560        MST7    010306
MIALLD  033356        MJSIA   033432        MLDDM3  061342        MRTF    031570        MSW37   012614
MIALLF  033366        MJSIB   033434        MLDDM4  061476        MRTO    031602        MSXP    104076
MIL     033120        MJSIC   033446        MLDDM5  061616        MRTOA   031632        MSXT    017302
MILA    033144        MJSID   033472        MLDDM6  061722        MRTOB   031634        MSXTCC  017140
MILAO   030712        MJSIE   033474        MLDDM7  062016        MRTOC   031646        MS11    011046
MILB    033146        MJSIF   033506        MLDM27  062120        MRTOD   031672        MS22    011320
MILD    033156        MJSR    013716        MLDSUB  072322        MRTOE   031674        MS33    011550
MILF    033166        MJSRA   014034        MLFSUB  071274        MRTOF   031706        MS77    012040
MILLBO  031012        MJSRB   014200        MLS1    076550        MRTS    015002        MTP     031720
MILLO   030224        MJSRC   014346        MLS2    076662        MSB     064216        MTPA    032122
MILLOA  030270        MJSR1   014036        MLS3    076774        MSBCC   016416        MTPAA   032176
MILLOB  030276        MJSR1A  014050        MLS4    077124        MSBCCC  016466        MTPAE   032226
MILO    033200        MJSR1B  014074        MLS5    077240        MSCD    102650        MTPAH   032174
MILOA   033232        MJSR2   013754        MLS6    077370        MSCF    103734        MTPAL   032164
MILOB   033234        MJSR2A  013766        MLS7    077510        MSDF    075670        MTPB    031752
MILOC   033246        MJSR2B  014012        MLXP    101230        MSER  = 177744        MTPF    031772
MILOD   033272        MJSR3   014202        MMARK   017554        MSFD    075354        MTPO    032004
MILOE   033274        MJSR3A  014214        MMODD   073660        MSFDI   076352        MTPOA   032034
MILOF   033306        MJSR3B  014240        MMODF   072564        MSOB    017500        MTPOB   032036
MIOT    032246        MJSR4   014116        MMRL5   012452        MSPAA   007640        MTPOC   032050
MIOTA   032270        MJSR4A  014130        MMR0  = 177572        MSPAU   027576        MTPOD   032074
MIOTB   032272        MJSR4B  014154        MMR1  = 177574        MSPB    005732        MTPOE   032076
MIOTD   032302        MJSR5   014350        MMR2  = 177576        MSPBB   007724        MTPOF   032110
MIOTF   032312        MJSR5A  014362        MMR3  = 172516        MSPC    005756        MTPQ    031762
MIOTO   030322        MJSR5B  014406        MMU     137552        MSPD    006006        MTPR    031750
MITO    032324        MJSR6   014264        MMULD   071536        MSPEO   006044        MTRPO   030516
MITOA   032354        MJSR6A  014276        MMULF   070500        MSPF    006106        MTRY    027740
MITOB   032356        MJSR6B  014322        MMUTRP  137724        MSPG    006150        MTRYA   030014
MITOC   032370        MJSR7   014432        MMVCC   015550        MSPH    006206        MTRYB   030034
MITOD   032414        MJSR7A  014444        MMVEC = 000250        MSPI    006252        MTRYM   030066
MITOE   032416        MJSR7E  014470        MM11    010700        MSPJ    006312        MTSO    015550
MITOF   032430        MJU1    012772        MM22    011246        MSPL    006432        MTT     031176
MJ      012720        MJU1A   013004        MNCCCC  016300        MSPM    006500        MTTA    031232
MJP     013556        MJU2    012734        MNGOP   063406        MSPN    006556        MTTB    031234
MJP17   013566        MJU2A   012746        MNNRM1  062226        MSPO    006624        MTTD    031244
MJP27   013610        MJU2B   012756        MNNRM2  062506        MSPP    006730        MTTE    031254
MJP27A  013622        MJU3    013016        MNNRM3  062660        MSPQ    007026        MTTR    031360
MJP37   013674        MJU3A   013030        MNNRM4  063022        MSPR    007134        MTTRA   031424
MJP37A  013706        MJU3B   013040        MNRM    064360        MSPS    007222        MTTRB   031426
MJP67   013632        MJU4    013112        MODE1   046034        MSPT    007270        MTTRC   031440
MJP67A  013644        MJU4A   013124        MODE2   047414        MSPU    007306        MTTRD   031476
MJP67B  013660        MJU4B   013136        MODGAR  073650        MSPV    007406        MTTRE   031500
MJP77   013662        MJU5    013052        MRLB1   012200        MSPVO   007352        MTTRF   031512
MJP77E  013716        MJU5A   013064        MRLCC   016552        MSPX    007452        MTTS    031266
MJRA    014474        MJU5B   013076        MRLO    012126        MSPY    007522        MTTSA   031322
MJR27   014530        MJU6    013152        MRL2    012260        MSPZ    007570        MTTSB   031326
MJR27A  014564        MJU6A   013164        MRL3    012336        MSPO    005710        MTTSD   031336
MJR27B  014604        MJU7    013204        MRL4    012412        MSTB3   007774        MTTSE   031346
MJR37   014666        MJU7E   013216        MRL6    012514        MSTO    031102        MTTSQ   031324
MJR37A  014722        MJ2     012770        MRL7    012552        MSTOE   031156        MUVAD   064500
MJR6A   014662        MJ5     013150        MRRB1   012672        MSTOEE  031164        MXDF1   063142
MJR6B   014664        MJ7     013200        MRRCC   016632        MST4    010046        MXOR    017420
MJR67   014606        MLCD    100612        MRRO    012650        MST4B   010104        MXRCC   017210
MJR67A  014642        MLCF    077760        MRT     031524        MST5    010144        M2      004766
MJR77   014742        MLDC    065376        MRTA    031546        MST5B   010176        M3      005002
```

Symbol table

```
M4          005022      POLY    = 120001    SDPAR5= 172272    SW01    = 000002    TAB42   003724
M5          005042      PROCNT  023126      SDPAR6= 172274    SW02    = 000004    TAB43   003734
M6          005062      PR0     = 000000    SDPAR7= 172276    SW03    = 000010    TAB45   003744
NEWADD      003026      PR1     = 000040    SDPDR0= 172220    SW04    = 000020    TAB46   003754
NEWDAT      114160      PR2     = 000100    SDPDR1= 172222    SW05    = 000040    TAB47   003764
NOABRT      141356      PR3     = 000140    SDPDR2= 172224    SW06    = 000100    TAB47A  003774
NOFPA       140166      PR4     = 000200    SDPDR3= 172226    SW07    = 000200    TAB48   004004
NOTOK       140002      PR5     = 000240    SDPDR4= 172230    SW08    = 000400    TAB49   004014
NULL    =   000000      PR6     = 000300    SDPDR5= 172232    SW09    = 001000    TAB5    003274
NXMFIN      047636      PR7     = 000340    SDPDR6= 172234    SW1     = 000002    TAB5A   003304
NXMPAR      111350      PS      = 177776    SDPDR7= 172236    SW10    = 002000    TAB6    003314
NXMTRP      047244      PSW     = 177776    SEQ     003072    SW11    = 004000    TAB6A   003324
ODDXX       033620      PSWBTS  005612      SFDSUB  075550    SW12    = 010000    TAB7    003334
OK          137760      PWRVEC= 000024      SIMGOA  002702    SW13    = 020000    TAB8    003344
OKAY7       043056      Q22EN   003002      SIPAR0= 172240    SW14    = 040000    TAB9    003354
OKAY7A      043066      Q22INT  137350      SIPAR1= 172242    SW15    = 100000    TAPAB0  107514
OKA7        043044      Q22SIZ  137070      SIPAR2= 172244    SW2     = 000004    TA114   021170
OK1         140006      RAMPAR  137012      SIPAR3= 172246    SW3     = 000010    TA116   022644
OK7         043026      RBUF    = 177562    SIPAR4= 172250    SW4     = 000020    TBITVE= 000014
ONOQ22      137314      RCSR    = 177560    SIPAR5= 172252    SW5     = 000040    TB114   021200
PAR         137674      RDCHR   = 104410    SIPAR6= 172254    SW6     = 000100    TC114   021210
PARAD1      045664      RDLIN   = 104411    SIPAR7= 172256    SW7     = 000200    TD114   021220
PARAD2      047276      RDOCT   = 104412    SIPDR0= 172200    SW8     = 000400    TEMP    002740
PARVA1      045716      RECDAT  114150      SIPDR1= 172202    SW9     = 001000    TE102   017760
PARVA2      047330      RECDST  003142      SIPDR2= 172204    SXPSUB  104256    TE103   020002
PARVA3      047446      RECFEC  003122      SIPDR3= 172206    TAB1    003234    TE104   020024
PAR1        137676      RECST   003132      SIPDR4= 172210    TAB10   003364    TE105   020046
PCR     =   177522      RESVEC= 000010      SIPDR5= 172212    TAB11   003374    TE106   020070
PDR         137652      RET1    015266      SIPDR6= 172214    TAB11A  003404    TE107   020112
PDR1        137654      RET2    015354      SIPDR7= 172216    TAB12   003414    TE110   020134
PHY1        046002      RET3    015432      SIXBIT  115460    TAB13   003424    TE111   020156
PIR     =   177772      RITEDA  114156      SLEND   122620    TAB13B  003434    TE112   020200
PIRQ    =   177772      RTSE    015054      SLOC00  003012    TAB14   003444    TE113   020422
PIRQNX      034222      RTS1    015022      SLOC01  003014    TAB15   003454    TE113A  020642
PIRQT       125736      RTS6    015032      SPAU1   027614    TAB16   003464    TE114   021014
PIRQVE= 000240      RXXX    034040      SPAU2   027636    TAB17   003474    TE115   022164
PIRRTN      034522      R6      =%000006    SPAU3   027656    TAB18   003504    TE115A  022374
PIRTBL      034324      R7      =%000007    SPAU4   027702    TAB2    003244    TE115B  022404
PIRTEX      034224      SAVBR   002730      SPAU5   027716    TAB21   003514    TE115C  022412
PIRXXX      034166      SAVMR0  003042      SPAU6   027734    TAB22   003524    TE115D  022420
PIR1        034224      SAVMR1  003044      SPS     003074    TAB23   003534    TE115F  022426
PIR2        034344      SAVMR2  003046      SPSJ    003076    TAB24   003544    TE116   022430
PIR2EX      034452      SAVPCR  002726      SR0     = 177572  TAB25   003554    TE116A  022704
PIR3        034452      SAVPOS  003156      SR1     = 177574  TAB26   003564    TE116B  022710
PIR3EX      034546      SAVSUP  003036      SR2     = 177576  TAB27   003574    TE116C  022722
PIR4        034546      SAVSWR  003050      SR3     = 172516  TAB28   003604    TE116D  022732
PIR5        034624      SAVUSE  003040      STACK   = 001100  TAB29   003614    TE117   022742
PIR5EX      035034      SAV30   004112      START   004024    TAB29A  003624    TE117A  023054
PIR6        035034      SAV32   004114      STBOT   = 001000  TAB3    003254    TE120   023060
PIR6EX      035136      SCDSUB  103470      STKLMT= 177774    TAB30   003634    TE120A  023134
PITBL1      034430      SCOPE   = 000004    STMOVI  113326    TAB31   003644    TE121   023210
PITBL2      035020      SDFSUB  076120      STMOVT  114450    TAB32   003654    TE122   023574
PI1         034474      SDPAR0= 172260      SUBT    052232    TAB33   003664    TE123   024046
PI2         034506      SDPAR1= 172262      SWR     001140    TAB34   003674    TE124   024266
PI3         034510      SDPAR2= 172264      SWREG   000176    TAB4    003264    TE125   024520
PLF0        043674      SDPAR3= 172266      SW0     = 000001  TAB40   003704    TE125A  024760
PLF1        044066      SDPAR4= 172270      SW00    = 000001  TAB41   003714    TE126   025206
```

Symbol table

```
TE126A  025254        TSFP5   053134        TST13   107044        TS14    047134        T123D   024254
TE126B  025730        TSFP6   053240        TST14   107360        TS15    050020        T123E   024256
TE127   026246        TSFP7   053440        TST15   107524        TS16    051024        T123F   024262
TE127A  026446        TSF10   054054        TST16   107752        TS16A   051016        T124A   024462
TE130   026710        TSF13   054424        TST17   110156        TS1822  046066        T124B   024472
TE130A  027166        TSF14   054574        TST2    104376        TS26D0  057236        T124C   024500
TF114   021230        TSF15   055012        TST20   110402        TS26D1  057246        T124D   024506
TG114   021240        TSF16   055202        TST21   111150        TS26D2  057256        T124E   024510
THRBIT  115726        TSF17   055414        TST22   111360        TS26D3  057266        T124F   024514
TH114   021250        TSF2    052514        TST23   111600        TS26D4  057276        T13FIN  046066
TIMDEL  120602        TSF20   055612        TST24   112136        TS26D5  057306        T14     047160
TIMEOU  053114        TSF21   056032        TST25   112714        TS27D0  057600        T14FIN  047532
TIMOUT  003000        TSF22   056216        TST26   113056        TS27D1  057610        T15     050070
TKVEC =  000060       TSF23   056352        TST27   114172        TS27D2  057620        T15A    050146
TMM16A  050534        TSF24   056552        TST3    104542        TS27D3  057630        T15FIN  050174
TMM16B  050414        TSF31   060354        TST30   115314        TS27D4  057640        UDPAR0= 177660
TMM16C  050444        TSF6    053430        TST31   115466        TS27D5  057650        UDPAR1= 177662
TMM16D  050474        TSF7    053624        TST32   115734        TS30D0  060144        UDPAR2= 177664
TMM16E  050524        TSLOOP  114556        TST33   116254        TS30D1  060154        UDPAR3= 177666
TMM16F  050526        TSMA    043116        TST34   116406        TS30D2  060164        UDPAR4= 177670
TM16A   051120        TSMB    043136        TST35   116772        TS30D3  060174        UDPAR5= 177672
TOUT    137542        TSMC    043146        TST36   117156        TS30D4  060204        UDPAR6= 177674
TPVEC =  000064       TSMMU0  015056        TST37   117464        TS30D5  060214        UDPAR7= 177676
TRAPVE=  000034       TSMMU1  035152        TST4    104642        TS32D0  060576        UDPDR0= 177620
TRPFLG  140054        TSMMU2  035236        TST40   117632        TS32D1  060606        UDPDR1= 177622
TRPOA   030562        TSMMU3  036436        TST41   120134        TS32D3  060626        UDPDR2= 177624
TRPOB   030570        TSMMU4  036736        TST42   120302        TS32D4  060636        UDPDR3= 177626
TRTVEC=  000014       TSMMU5  037666        TST43   120612        TS32D5  060646        UDPDR4= 177630
TRYMA   030126        TSMMU6  040020        TST44   120652        TS33D0  061072        UDPDR5= 177632
TRYMB   030160        TSMMU7  042524        TST45   120750        TS33D1  061102        UDPDR6= 177634
TRYMC   030206        TSMMU8  043152        TST46   121032        TS33D2  061112        UDPDR7= 177636
TSD31   060252        TSMMU9  043354        TST47   121246        TS33D3  061122        UFDFLG  004116
TSEND   115276        TSMM10  044254        TST5    105226        TS33D4  061132        UFDSET= 000001
TSFP1   052356        TSMM11  044636        TST50   121300        TS33D5  061142        UIPAR0= 177640
TSFP10  053664        TSMM12  045074        TST51   121644        TS6DA   053410        UIPAR1= 177642
TSFP11  054114        TSMM13  045462        TST52   122112        TS6DAT  053420        UIPAR2= 177644
TSFP12  054220        TSMM14  046530        TST53   122402        TS7     042774        UIPAR3= 177646
TSFP13  054270        TSMM15  047654        TST54   122620        TS7DA1  053634        UIPAR4= 177650
TSFP14  054434        TSMM16  050174        TST55   123142        TS7DA2  053644        UIPAR5= 177652
TSFP15  054606        TSMM6A  040126        TST56   123516        TS7DA4  053654        UIPAR6= 177654
TSFP16  055024        TSMM6B  040564        TST57   123764        TS7FIN  043150        UIPAR7= 177656
TSFP17  055214        TSMM6C  041324        TST6    105412        TS9FIN  044252        UIPDR0= 177600
TSFP2   052446        TSMM6D  042024        TST60   124136        TYPDS =  104405       UIPDR1= 177602
TSFP20  055426        TSM16A  050270        TST61   124336        TYPE  =  104401       UIPDR2= 177604
TSFP21  055624        TSM16B  050316        TST62   125304        TYPOC =  104402       UIPDR3= 177606
TSFP22  056044        TSM16C  050342        TST63   125562        TYPON =  104404       UIPDR4= 177610
TSFP23  056230        TSM16D  050406        TST64   125746        TYPOS =  104403       UIPDR5= 177612
TSFP24  056364        TSM7    043104        TST65   126046        T10FIN  044636        UIPDR6= 177614
TSFP25  056564        TSM9    043510        TST66   126230        T11FIN  045074        UIPDR7= 177616
TSFP26  056722        TSTADD  003024        TST7    105652        T114    021146        UNXPIR  034450
TSFP27  057320        TSTEND  116770        TS10    044554        T116    022612        UQUIET  004120
TSFP3   052544        TSTLOC  003162        TS10D1  054064        T12FIN  045460        VIREOP  126756
TSFP30  057662        TSTLUP  113410        TS10D2  054074        T122A   023672        VIR1    045750
TSFP31  060226        TST1    004740        TS10D4  054104        T122B   024024        VIR2    047362
TSFP32  060366        TST10   106126        TS11    044754        T123A   024230        VIR3    047500
TSFP33  060660        TST11   106370        TS11D1  054210        T123B   024240        VMKOR   004122
TSFP4   052660        TST12   106644        TS12    045412        T123C   024246        VQBE1   002676
```

Symbol table

```
VQBE2   002716          $DDW10  001310          $ETEND  001324          $MSGTY  001200          $TIMES  001164
VQPR1   002700          $DDW11  001312          $FATAL  001202          $MSWR   143376          $TKB    001146
VQPR2   002720          $DDW12  001314          $FFLG   141624          $MTYP1  001231          $TKS    001144
WC      002672          $DDW13  001316          $FILLC  001156          $MTYP2  001235          $TMP0   001160
WC2     002712          $DDW14  001320          $FILLS  001155          $MTYP3  001241          $TMP1   001162
WLDTRP  140044          $DDW15  001322          $GDADR  001120          $MTYP4  001245          $TN   = 000067
XBUF  = 177566          $DDW2   001270          $GDDAT  001124          $MXCNT  140774          $TPB    001152
XCBIT   016710          $DDW3   001272          $GET42  140430          $NULL   001154          $TPFLG  001157
XCSR  = 177564          $DDW4   001274          $GTSWR  142704          $NWTST= 000000          $TPS    001150
XME100  017714          $DDW5   001276          $HD   = 000001          $OCNT   142404          $TRAP   143522
XME101  017736          $DDW6   001300          $HIBTS  000232          $OMODE  142406          $TRAP2  143544
$APTHD  000232          $DDW7   001302          $HIOCT  143520          $OVER   140752          $TRP  = 000013
$ATYC   141404          $DDW8   001304          $ICNT   001104          $PASS   001206          $TRPAD  143556
$ATY1   141360          $DDW9   001306          $ILLUP  143744          $PASTM  000240          $TSTM   000236
$ATY3   141366          $DEVCT  001210          $INTAG  001135          $POWER  143752          $TSTNM  001102
$ATY4   141376          $DEVM   001256          $ITEMB  001114          $PWRDN  143604          $TTYIN  143354
$AUTOB  001134          $DOAGN  140450          $LF     001176          $PWRMG  143740          $TYPDS  142410
$BASE   001254          $DTBL   142614          $LFLG   141623          $PWRUP  143656          $TYPE   141626
$BDADR  001122          $ENDAD  140440          $LPADR  001106          $QUES   001174          $TYPEC  142040
$BDDAT  001126          $ENDCT  140406          $LPERR  001110          $RDCHR  143116          $TYPEX  142160
$BELL   001170          $ENDMG  140457          $MADR1  001232          $RDLIN  143246          $TYPOC  142206
$CDW1   001260          $ENULL  140454          $MADR2  001236          $RDOCT  143420          $TYPON  142222
$CDW2   001262          $ENV    001220          $MADR3  001242          $RDSZ = 000010          $TYPOS  142162
$CHARC  142156          $ENVM   001221          $MADR4  001246          $RTNAD  140452          $UNIT   001212
$CKSWR  142634          $EOP    140340          $MAIL   001200          $SAVR6  143750          $UNITM  000242
$CMTAG  001100          $EOPCT  140400          $MAMS1  001230          $SCOPE  140474          $USWR   001224
$CM3  = 000000          $ERFLG  001103          $MAMS2  001234          $SETUP= 000137          $VECT1  001250
$CM4  = 000002          $ERMAX  001115          $MAMS3  001240          $STUP = 177777          $VECT2  001252
$CNTLG  143371          $ERROR  140776          $MAMS4  001244          $SVLAD  140716          $XOFF = 000023
$CNTLU    5364          $ERRPC  001116          $MBADR  000234          $SVPC = 000232          $XON  = 000021
$CPUOP  001226          $ERRTB  001324          $MFLG   141622          $SWR  = 167400          $XTSTR  140514
$CRLF   001175          $ERTTL  001112          $MNEW   143407          $SWREG  001222          $$GET4= 000000
$DBLK   142624          $ESCAP  001166          $MSGAD  001214          $SWRMK= 000300          $OFILL  142405
$DDW0   001264          $ETABL  001220          $MSGLG  001216          $TESTN  001204          .$X   = 000232
$DDW1   001266

. ABS.  143762    000   (RW,I,GBL,ABS,OVR)
        000000    001   (RW,I,LCL,REL,CON)
Errors detected:  10

*** Assembler statistics


Work  file  reads: 471
Work  file writes: 403
Size of work file: 63968 Words  ( 250 Pages)
Size of core pool: 19684 Words  ( 75 Pages)
Operating  system: RSX-11M/PLUS (Under VAX/VMS)

Elapsed time: 00:06:36.83
COKDAF0,COKDAF0/NL:TOC/-SP=ORION.MLB/ML,COKDAF0.MAC/CS:GBL
```