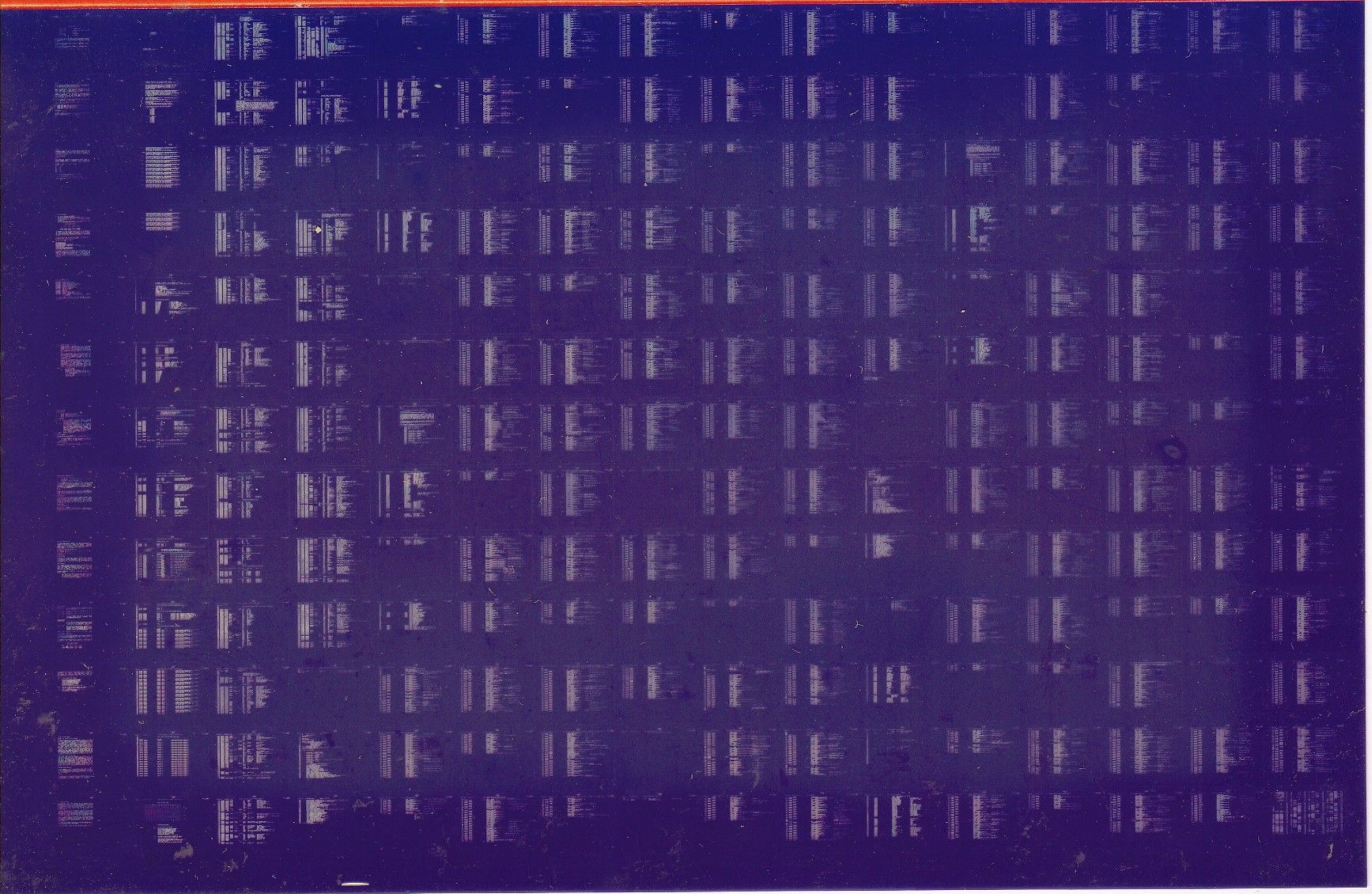
DMC-11

DMC-11 CROM & JUMP TEST
CZDMGD0

AH-8562D-MC

FICHE 1 OF 2

MAR 1978
digital
MADE IN USA



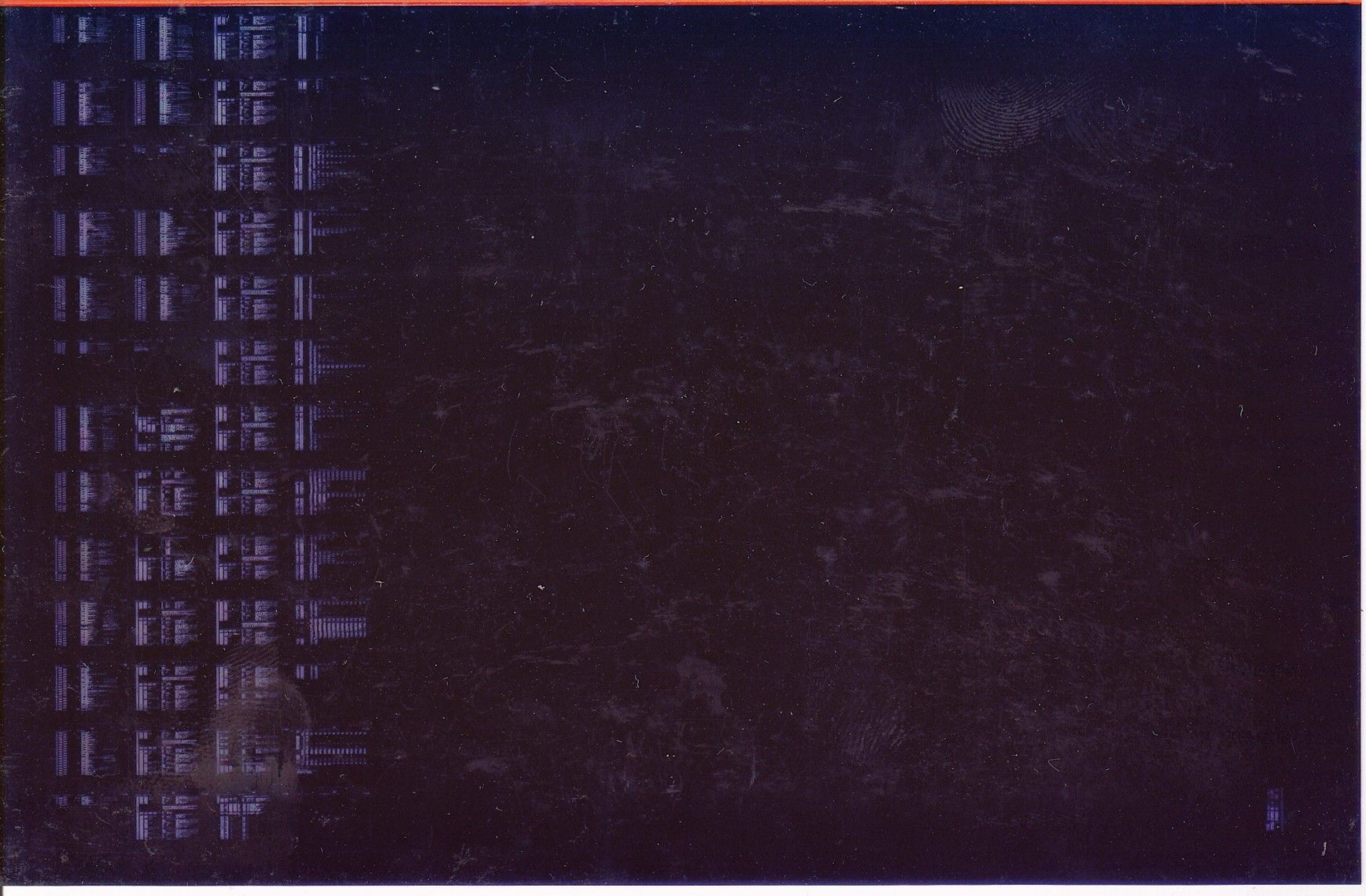
DMC-11

DMC-11 CROM & JUMP TEST
CZDMGD0

AH-8562D-MC

FICHE 2 OF 2

MAR 1978
digital
MADE IN USA



IDENTIFICATION

PRODUCT CODE:	AC-8560D-MC
PRODUCT NAME:	CZDMGDO DMC-11 CROM & JUMP TST
DATE:	FEB 1978
MAINTAINER:	DIAGNOSTICS
AUTHOR:	FAY BASHAW

1. ABSTRACT

The function of the DMC11 diagnostics is to verify that the option operates according to specifications. The diagnostics verfiy that there are no malfunctions and the all operations of the DMC11 are correct in its environment.

Parameters must be set up to alert the diagnostics to the DMC11 configuration. These parameters are contained in the STATUS TABLE and are generated in two ways: 1) Manual Input - the operator answers questions. 2) Autosizing - the program determines the parameters automatically.

CZDMGD tests the DMC11-AR and DMC11-AL micro-processors (M8200-YA M8200-YB). It performs jump tests on the micro-processor and verifies the control ROM of the M8200. This diagnostic will not run on a KMC (M8204), however it is possible to load the KMC CRAM with the DMC micro-code. See test 2 for details.

Currently there are five off line diagnostics that are to be run in sequence to insure that if an error should occur it will be detected at an early stage.

NOTE: Additional diagnostics may be added in the future.

The five diagnostics are:

1. DZDMC [REV] Basic W/R and Micro-processor tests
2. DZDME [REV] DDCMP Line unit tests
3. DZDMF [REV] BITSTUFF Line unit tests
4. CZDMGD [REV] Jump and CROM tests
5. DZDMH [REV] Free-running tests (Heat test tape)

2. REQUIREMENTS

2.1 EQUIPMENT

Any PDP11 family CPU (except an LSI-11) with minimum 8k memory
ASR 33 (or equilivalent)
DMC11-AR (M8200-YA) or an DMC11-AL (M8200-YB)

2.2 STORAGE

Program will use all 8K of memory except where ABL and BOOTSTRAP LOADER reside. Locations 1500 thru 1640; contain the "STATUS TABLE" information which is generated at start of diagnostics by manual input (questions) or automatically (auto-sizing). This area is an overlay area and should not be altered by the operator.

3. LOADING PROCEEDURE

3.1 METHOD

All programs are in absolute format and are loaded using the ABSOLUTE LOADER. NOTE: if the diagnostics are on a media such as DISK ,MAGTAPE,DECTAPE, or CASSETTE; follow instructions for the monitor which has been provided on that specific media.

ABSOLUTE LOADER starting address *500

MEMORY	* SIZE
4k	17
8k	37
12k	57
16k	77
20k	117
24k	137
28k	157

3.1.1 Place address of ABS loader into switch register.
(also place 'HALT' SW up)

3.1.2 Depress 'LOAD ADDRESS' key on console and release.

3.1.3 Depress 'START KEY' on console and release (program should now be loading into CPU)

4. STARTING PROCEEDURE

a. Set switch register to 000200
b. Depress 'LOAD ADDRESS' key and release
c. Set SWR to zero for 'AUTO SIZING' or SWR bit0=1 for manual input (questions) or SWR bit7=1 to use existing parameters set up by a previous start or a previously run DMC11 diagnostic.
d. Depress 'START KEY' and release. The program will type Maindec Name and program name (if this was the first start up of the program) and also the following:

MAP OF DMC11 STATUS

PC	CSR	STAT1	STAT2	STAT3
001500	160010	145310	177777	000000
001510	160020	145320	177777	000000

The program will type 'R' and proceed to run the diagnostic. The above is only an example. This would indicate the status table starting at add. 1500 in the program. In this example the table contains the information and status of two DMC11'S. THE STATUS TABLE MUST BE VERIFIED BY THE USER IF AUTO SIZING IS DONE. For information of status table see section 8.4 for help.

If the diagnostic was started with SW00=1 indicating manual parameter input then the following shows an example of the questions asked and some example answers:

```
HOW MANY DMC11'S TO BE TESTED?1

01
CSR ADDRESS?160010
VECTOR ADDRESS?310
BR PRIORITY LEVEL?  (4,5,6,7)?5
DOES MICRO-PROCESSOR HAVE CRAM?  (Y OR N)N
WHICH LINE UNIT?  IF NONE TYPE "N",  IF  M8201  TYPE  "1",  IF
M8202 TYPE "2"?1
IS THE LOOP BACK CONNECTOR ON?Y
SWITCH PAC#1 (DDCMP LINE#)?377
SWITCH PAC#2 (BM873 BOOT ADD)?377
```

Following the questions the status map is printed out as described above, the information in the map reflects the answers to the questions. If the diagnostic was started with SW00=0 and SW07=0 (AUTO-SIZING) then no questions are asked and only the status-map is printed out. If AUTO-SIZING is used the status information must be verified to be correct (match the hardware). if it does not match the hardware the diagnostic must be restarted with SW00=1 and the questions answered.

4.1 CONTROL SWITCH SETTINGS

SW 15 Set: Halt on error
SW 14 Set: Loop on current test
SW 13 Set: Inhibit error print out
SW 12 Set: Inhibit type out/abell on error.
SW 11 Set: Inhibit iterations. (quick pass)
SW 10 Set: Escape to next test on error
SW 09 Set: Loop with current data
SW 08 Set: Catch error and loop on it
SW 07 Set: Use previous status table.
SW 06 Set: Halt in ROMCLK routine before clocking micro-processor
SW 05 Set: Reserved
SW 04 Set: Reserved
SW 03 Set: Reselect DMC11's desired active
SW 02 Set: Lock on selected test
SW 01 Set: Restart program at selected test
SW 00 Set: Build new status table from questions. (If SW07=0 and SW00=0 a new status table is built by auto-sizing)

Switch 06 and 08-15 are dynamic and can be changed as needed while the diagnostic is running. Switches 00-03 and switch 07 are static, and are used only on starting or restarting the diagnostic.

4.1.2 SWITCH REGISTER OPTIONS (at start up)

SW 01 RESTART PROGRAM AT SELECTED TEST. It is strongly suggested that at least one pass has been made before trying to select a test, the reason being is that the program has to clear areas and set up parameters. When this switch is used the diagnostic will ask TEST NO.? Answer by typing the number of the test desired and carrige return to begin execution at the selected test.

SW 02 LOCK ON SELECTED TEST. This switch when used with SW01 will cause the program to constantly loop on the selected test. Hitting any key on the console will let it advance to the next test and loop until a key is hit again. If SW02=0 when SW01 is used. The program will begin at the selected test and continue normal operations.

SW 03 RESELECT DMC11'S DESIRED ACTIVE. Please note that a message is typed out for setting the switch register equal to DMC11's active. this means if the system has four DMC11s; bits 00,01,02,03 will be set in loc 'DMACTV' from the switch register. Using this switch(SW00) alters that location;therefore if four DMC11s are in the system ***DO NOT*** set switchs greater than SW 03 in the up position. this would be a fatal error. do not select more active DMC11s than there is information on in the status table.

METHOD: A: Load address 200
B: Start with SW 00=1
C: Program will type message
D: Set a switch for each DMC desired active.
EXAMPLE: If you have 4 DMC's but only want to run the first and the last set SWR bits 0 and 3 = 1. PRESS CONTINUE
E: Number (IF VALID) will be in data lights (excluding 11/05)
F: Set with any other switch settings desired.
PRESS CONTINUE.

4.1.3 DYNAMIC SWITCHES

ERROR SWITCHES

1. SW 12 Delete print out/bell on error.
2. SW 13 Delete error printout.
3. SW 15 Halt on the error.
4. SW 08 Goto beginning of the test(on error).
5. SW 10 Goto next test(on error).

SCOPE SWITCHES

1. SW06 Halt in ROMCLK routine before clocking micro-processor instruction. This allows the operator to scope a micro-processor instruction in the static state before it is clocked. Hit continue to resume running.
2. SW09 (if enabled by 'SCOPI') on an error; If an '*' is printed in front of the test no. (ex. *TEST NO. 10) SW09 is incorporated in that test and therefore SW09 is usually the best switch for the scope loop (SW14=0, SW10=0, SW09=1, SW08=0). If SW09 is not enabled; and there is a HARD error (constant); SW08 is best. (SW14=1,0, SW10=0, SW09=0, SW08=1). for intermittemt errors; SW14=1 will loop on test reguardless of error or not error. (SW14=1, SW10=0, SW09=0, SW08=1,0)
3. SW11 Inhibit interations.
4. SW14 Loop on current test.

4.2 STARTING ADDRESS

Starting address is at 000200 there are no other starting addresses for the DMC11 diagnostics. (See Section 4.0)

NOTE: If address 000042 is non-zero the program assumes it is under ACT11 or XXDP control and will act accordingly after all available DMC11's are tested the program will return to 'XXDP' or 'ACT-11'.

5. OPERATING PROCEDURE

When program is initially started messages as described in section 4.0 will be printed, and program will begin running the diagnostic

5.2 PROGRAM AND/OR OPERATOR ACTION

The typical approach should be

1. Halt on error (via SW 15=1) when ever an error occurs.
2. Clear SW 15.
3. Set SW 14: (loop on this test)
4. Set SW 13: (inhibit error print out)

The TEST NUMBER and PC will be typed out and possibily an error message (this depends on the test) to give the operator an idea as to the source of the problem. If it is necessary to know more information concerning the error report; LOOK IN THE LISTING for that TEST NUMBER which was typed out and then NOTE THE PC of the ERROR REPORT this way the EXACT FUNCTION of the test CAN BE DETERMINED.

6. ERRORS

As described previously there will always be a TEST NUMBER and PC typed out at the time of an error (providing SW 13=0 and SW 12=0). in most cases additional information will be supplied in the the error message to give the operator an indication of the error.

6.2 ERROR RECOVERY

If for some reason the DMC11 should 'HANG THE BUS' (gain control of bus so that console manual functions are inhibited) an init or power down/up is necessary for operator to regain control of cpu. If this should happen; look in location 'TSTNO' (address 1226)for the number of the test that was running at the time of the catastrophic error. In this way the operator will have an idea as to what the DMC11 was doing at the time of the error.

7. RESTRICTIONS

7.1 STARTING RESTRICTIONS

See section 4. (PLEASE)
Status table should be verified reguardless of how program was started. Also it is important to use this listing along with the information printed on the TTY to completly isolate problems.

7.2 OPERATING RESTRICTIONS

The first time a DMC11 diagnostic is loaded into core ard run the STATUS TABLE must be set up. This is done by manual input (SW00=1) or by autosizing (SW00=0 and SW07=0). Thereafter however the status table need not be setup by subsequent restarts or even loading the next DMC diagnostic because the STATUS TABLE is overlayed. The current parameters in the STATUS TABLE are used when SW07=1 on start up.

7.3 HARDWARE CONFIGURATION RESTRICTIONS

DMC11(M8200)- Jumper W1 must be in, and switch 7 of E76 must be in the OFF position.

KMC(M8204)- Jumper W1 must be in.

8. MISCELLANEOUS

8.1 EXECUTION TIME

All DMC11 device diagnostics will give an 'END PASS' message (providing no errors and sw12=0) within 4 mins. This is assuming SW11=1 (DELETE ITERATIONS) is set to give the fastest possible execution. The actual execution time depends greatly on the PDP11 CPU configuration and the amount of memory in the system.

8.2 PASS COMPLETE

NOTE: EVERY time the program is started; the tests will run as if SW11 (delete iterations) was up (=1). This is to 'VERIFY NO HARD ERRORS' as soon as possible. Therefore the first pass -EACH TIME PROGRAM IS STARTED- will be a 'QUICK PASS' until all DMC11's in system are tested. When the diagnostic has completed a pass the following is an example of the print out to be expected.

```
END PASS CZDMGD CSR:  175000 VEC:  0300 PASSES:  000001
ERRORS:  000000
```

NOTE: The pass count and error counts are cummulitive for each DMC11 that is running, and are set to zero only when the diagnostic is started. Therefore after an overnight run for example, the total passes and errors for each DMC11 since the diagnostic was started are reflected in PASSES: and ERRORS:.

8.4 KEY LOCATIONS

RETURN (1214) Contains the address where program will return when iteration count is reached or if loop on test is asserted.

NEXT (1216) Contains the address of the next test to be peformed.

TSTNO (1226) Contains the number of the test now being peformed.

RUN (1316) The bit in 'RUN' always points to the DMC11 currently being tested. EXAMPLE: (RUN) 1302/0000000001000000 Means that DMC11 no.06 is the DMC11 now running.

DMCR00-DMCR17
DMST00-DMST17
(1500)-(1640)

These locations contain the information needed to test up to 16 (decimal) DMC11s sequentialy. they contain the CSR,VECTOR and STATUS concerning the configuration of each DMC11.

DMACTV (1306) Each bit set in this location indicates that the associated DMC11 will be tested in turn. EXAMPLE: (DMACTV) 1276/0000000000011111 means that DMC11 no. 00,01,02,03,04 will be tested. EXAMPLE: (DMACTV) 1276/0000000000010001 Means that DMC11 no. 00,04 will be tested.

DMCSR (1404) Contains the CSR of the current DMC11 under test.

8.4A 'STATUS TABLE' (1500-1640)

The table is filled by AUTO SIZING or by the manual parameter input (questions) as described previously. Also if desired by user; the locations may be altered by hand (toggled in) to suit the specific configuration.

The example status map shown below contains information for two DMC11'S. the table can .contain up to 16 DMC11'S. Following the map is a description of the bits for each map entry

MAP OF DMC11 STATUS

PC	CSR	STAT1	STAT2	STAT3
001500	160010	145310	177777	000000
001510	160020	016320	000000	000000

Each map entry contains 4 words which contain the status information for 1 DMC11. The PC shows where in core memory the first of the 4 words is. In the example above the first DMC'S status is in locations, 1500, 1502, 1504, and 1506. The second DMC status is located at 1510, 1512, 1514, and 1516. The information contained in each 4 word entry is defined as follows:

CSR: Contains DMC11 CSR address

STAT1: BITS 00-08 IS DMC11 VECTOR ADDRESS
BIT15=1 MICRO-PROCESSOR HAS CRAM
BIT15=0 MICRO-PROCESSOR HAS CROM
BIT14=1 TURNAROUND CONNECTOR IS ON
BIT14=0 NO TURNAROUND CONNECTOR
BIT13=0 LINE UNIT IS AN M8201
BIT13=1 LINE UNIT IS AN M8202
BIT12=1 NO LINE UNIT
BITS 09-11 IS DMC11 BR PRIORITY LEVEL

STAT2: LOW BYTE IS SWITCH PAC#1 (DDCMP LINE NUMBER)
HIGH BYTE IS SWITCH PAC#2 (BM873 BOOT ADD)

STAT3: BIT0=1 PERFORM FREE RUNNING TESTS ON KMC
BIT1=0 DMC11-AR (LOW SPEED)
BIT1=1 DMC11-AL (HIGH SPEED)

8.5 METHOD OF AUTO SIZING

8.5.1 FINDING THE CONTROL STATUS REGISTER.

The auto-sizing routine finds a DMC11 as follows: It starts at address 160000 and tests all address in increments of 10 up to and including address 167760. If the address does not time out, the following is done, the first CROM address is written to a 125252 then it is read back. If it contains a -1 or 125252 or 626 or 16520 a DMC11 or KMC11 has been found, if not, the address is updated by 10 and the search continues. A -1 indicates a DMC11 with no CROM, a 125252 indicates a KMC11 with CRAM, a 626 indicates a DMC11-AL, and a 16520 indicates a DMC11-AL. Further tests are performed at this point to determine which line unit, if any, is installed, if a loop-back connector is installed and various switch settings on the line unit. THIS IS WHY THE STATUS TABLE MUST BE VERIFIED BY THE USER AND IF ANY OF THE INFORMATION DOES NOT AGREE WITH THE HARDWARE THE DIAGNOSTIC MUST BE RESTARTED AND THE QUESTIONS MUST BE ANSWERED. All DMC11's in the system will be found by the auto-sizer. If it does not find a DMC11 the diagnostic must be restarted and the questions answered.

8.5.2 FINDING THE VECTOR AND BR LEVEL

The vector area (address 300-776) is filled with the instruction IOT and '.+2' (next address). The processor status is started at 7 and the DMC is programmed to interrupt. The PS is lowered by 1 until the DMC interrupts, a delay is made and if no interupt occures at PS level 3 (because of a bad DMC11) the program assumes vector address 300 at BR level 5 and the problem should be fixed in the diagnostic. Once the problem is fixed; the program should be re-setup again to get correct vector. If an interupt occured; the address to which the DMC11 interupted to is picked up and reported as the vector. NOTE: if the vector reported is not the vector set up by you; there is a problem and AUTO SIZING should not be done.

8.6 SOFTWARE SWITCH REGISTER

If the diagnostic is run on an 11/04 or other CPU without a switch register then a software switch register is used to allow user the same switch options as described previously. If the hardware switch register does not exist or if one does and it contains all ones (177777) this software switch register is used.

Control:

To obtain control at any allowable time during execution of the diagnostic the operator types a CTRL G on the console terminal keyboard. As soon as the CTRL G is recognized, by the diagnostic, the following message will be displayed:

SWR=XXXXXX NEW?

Where XXXXXX is the current contents of the software switch register in octal. The software control routine will then await operator action. At which time the operator is required to type one or more of the legal characters: 1) 0 - 7, 2) line feed(<LF>), 3) carriage return(<CR>), or 4) control-U (CTRL U). No check is made for legality. If the input character is not a <LF>, <CR>, or CTRL U it is assumed to be an octal digit.

To change the contents of the SSR the operator simply types the new desired value in octal - leading zeros need not be typed. And terminates the input string with a <CR> or <LF> depending on the program action desired as described below. The input value will be truncated to the last 6 digits typed. At least one digit must be typed on any given input string prior to the terminator before a change to the SSR will occur.

When the input string is terminated with a <CR> the diagnostic will continue execution from the point at which it was interrupted. If a <CR> is the only thing typed the program will continue without changing the SSR. The <LF> differs from the <CR> by restarting the program as if it were restarted at address 200.

If a CTRL U is typed at any point in the input string prior to the terminator the input value will be disregarded and the prompt displayed (SWR = XXXXXX NEW?).

To set the SSR for the starting switches, first load the diagnostic, then hit CTRL G, then start the diagnostic.

DOCUMENT
****
'CZDMGD LST
****

```
6       MAINDEC-11-CZDMG-D   DMC11 CROM AND JUMP TESTS
        COPYRIGHT 1976, DIGITAL EQUIPMENT CORP., MAYNARD, MASS. 01754
        ------------------------------------------------------------

1675    ************************** TEST 1 **************************
        THIS IS A SPECIAL TEST WHICH IS ONLY EXECUTED  ONE TIME,
        THE FIRST PASS AFTER THE DIAGNOSTIC IS LOADED. IT TYPES ON
        THE CONSOLE THE PART NUMBERS OF THE CROMS WHICH THIS
        REVISION SUPPORTS. TO FORCE A TYPE OUT PATCH LOCATION
        ROMNUM: TO A ZERO.

1696    ************************** TEST 2 **************************
        THIS IS A SPECIAL TEST WHICH WILL RUN ON A KMC (DMC WITH
        WRITTABLE CONTROL STORE) TO LOAD THE CRAM WITH THE DDCMP
        MICRO-CODE. FIRST BE SURE BIT1 OF STAT3 IS SET UP AS FOLLOWS
        1=LOCAL HIGH SPEED CODE,  0=REMOTE LOW SPEED CODE  THE STATUS
        OF STAT3 BIT1 DETERMINES WHICH MICRO-CODE WILL
        BE LOADED IN THE KMC. LOOP ON THIS TEST FOR A FEW SECONDS
        TO LOAD THE KMC.

1727    ************************** TEST 3 **************************
        TEST OF BR RIGHT SHIFT
        VERIFY THAT A DEST OF BR RSH (011) OF A MICRO-INSTRUCTION
        SHIFTS THE RESULTING BR DATA RIGHT ONCE.

1768    ************************** TEST 4 **************************
        CROM READ TEST
        THIS TEST READS EACH ROM LOCATION AND COMPARES
        IT TO A SOFTWARE DUPLICATE OF THE CROM. THIS TEST
        ALSO TESTS THE JUMP(I) MICRO-PROCESSOR INSTRUCTION.

1773    IF THIS TEST FAILS CHECK YOUR CROM PART NUMBERS.
        CZDMG-D SUPPORTS THE FOLLOWING PART NUMBERS:

        DMC11-AR  (M8200-YA)
                  23-630A9
                  23-631A9
                  23-632A9
                  23-633A9
                  23-634A9
                  23-635A9
                  23-636A9
                  23-637A9

        DMC11-AL  (M8200-YB)
                  23-622A9
                  23-623A9
                  23-624A9
                  23-625A9
                  23-626A9
                  23-627A9
                  23-628A9
                  23-629A9
```

```
1840    ************************** TEST 5 ****************************
        CROM TEST OF JUMP(I) NEVER MICRO-PROCESSOR INSTRUCTION.
        PERFORM THE JUMP INSTRUCTION
        VERIFY THAT THE JUMP DID NOT OCCUR BY READING
        THE CONTENTS OF THE NEW ROM PC(IT SHOULD INCREMENT BY ONE).

1898    ************************** TEST 6 ****************************
        CROM TEST OF JUMP(I) ALWAYS MICRO-PROCESSOR INSTRUCTION.
        PERFORM THE JUMP INSTRUCTION
        VERIFY THE JUMP BY READING THE CONTENTS OF THE NEW ROM PC

1952    ************************** TEST 7 ****************************
        CROM TEST OF JUMP(I) ON C BIT SET MICRO-PROCESSOR INSTRUCTION.
        SET THE C BIT, PERFORM THE JUMP INSTRUCTION,
        VERIFY THE JUMP BY READING THE CONTENTS OF THE NEW ROM PC

2009    ************************** TEST 10 ***************************
        CROM TEST OF JUMP(I) ON Z BIT SET MICRO-PROCESSOR INSTRUCTION.
        SET THE Z BIT, PERFORM THE JUMP INSTRUCTION,
        VERIFY THE JUMP BY READING THE CONTENTS OF THE NEW ROM PC

2066    ************************** TEST 11 ***************************
        CROM TEST OF JUMP(I) ON BRO SET MICRO-PROCESSOR INSTRUCTION.
        SET THE BRO BIT, PERFORM THE JUMP INSTRUCTION,
        VERIFY THE JUMP BY READING THE CONTENTS OF THE NEW ROM PC

2123    ************************** TEST 12 ***************************
        CROM TEST OF JUMP(I) ON BR1 SET MICRO-PROCESSOR INSTRUCTION.
        SET THE BR1 BIT, PERFORM THE JUMP INSTRUCTION,
        VERIFY THE JUMP BY READING THE CONTENTS OF THE NEW ROM PC

2180    ************************** TEST 13 ***************************
        CROM TEST OF JUMP(I) ON BR4 SET MICRO-PROCESSOR INSTRUCTION.
        SET THE BR4 BIT, PERFORM THE JUMP INSTRUCTION,
        VERIFY THE JUMP BY READING THE CONTENTS OF THE NEW ROM PC

2237    ************************** TEST 14 ***************************
        CROM TEST OF JUMP(I) ON BR7 SET MICRO-PROCESSOR INSTRUCTION.
        SET THE BR7 BIT, PERFORM THE JUMP INSTRUCTION,
        VERIFY THE JUMP BY READING THE CONTENTS OF THE NEW ROM PC

2294    ************************** TEST 15 ***************************
        CROM TEST OF JUMP(I) ON C BIT SET MICRO-PROCESSOR INSTRUCTION.
        CLEAR THE C BIT, PERFORM THE JUMP INSTRUCTION,
        VERIFY THAT THE JUMP DID NOT OCCUR BY READING
        THE CONTENTS OF THE NEW ROM PC(IT SHOULD INCREMENT BY ONE).

2352    ************************** TEST 16 ***************************
        CROM TEST OF JUMP(I) ON Z BIT SET MICRO-PROCESSOR INSTRUCTION.
        CLEAR THE Z BIT, PERFORM THE JUMP INSTRUCTION,
        VERIFY THAT THE JUMP DID NOT OCCUR BY READING
        THE CONTENTS OF THE NEW ROM PC(IT SHOULD INCREMENT BY ONE).
```

```
2410    ************************** TEST 17 ****************************
        CROM TEST OF JUMP(I) ON BR0 SET MICRO-PROCESSOR INSTRUCTION.
        CLEAR THE BR0 BIT, PERFORM THE JUMP INSTRUCTION,
        VERIFY THAT THE JUMP DID NOT OCCUR BY READING
        THE CONTENTS OF THE NEW ROM PC(IT SHOULD INCREMENT BY ONE).

2468    ************************** TEST 20 ****************************
        CROM TEST OF JUMP(I) ON BR1 SET MICRO-PROCESSOR INSTRUCTION.
        CLEAR THE BR1 BIT, PERFORM THE JUMP INSTRUCTION,
        VERIFY THAT THE JUMP DID NOT OCCUR BY READING
        THE CONTENTS OF THE NEW ROM PC(IT SHOULD INCREMENT BY ONE).

2526    ************************** TEST 21 ****************************
        CROM TEST OF JUMP(I) ON BR4 SET MICRO-PROCESSOR INSTRUCTION.
        CLEAR THE BR4 BIT, PERFORM THE JUMP INSTRUCTION,
        VERIFY THAT THE JUMP DID NOT OCCUR BY READING
        THE CONTENTS OF THE NEW ROM PC(IT SHOULD INCREMENT BY ONE).

2584    ************************** TEST 22 ****************************
        CROM TEST OF JUMP(I) ON BR7 SET MICRO-PROCESSOR INSTRUCTION.
        CLEAR THE BR7 BIT, PERFORM THE JUMP INSTRUCTION,
        VERIFY THAT THE JUMP DID NOT OCCUR BY READING
        THE CONTENTS OF THE NEW ROM PC(IT SHOULD INCREMENT BY ONE).
```

```
                                ;*MAINDEC-11-CZDMG-D   DMC11 CROM AND JUMP TESTS
                                ;*COPYRIGHT 1976, DIGITAL EQUIPMENT CORP., MAYNARD, MASS. 01754
                                ;*-------------------------------------------------------------------

                                ;STARTING PROCEDURE
                                ;LOAD PROGRAM
                                ;LOAD ADDRESS 000200
                                ;SWR=0  AUTOSIZE DMC11
                                ;SW07=1  USE CURRENT DMC11 PARAMETERS
                                ;SW00=1  INPUT NEW DMC11 PARAMETERS
                                ;PRESS START
                                ;PROGRAM WILL TYPE "MAINDEC-11-CZDMG-D   DMC11 CROM AND JUMP TESTS"
                                ;PROGRAM WILL TYPE STATUS MAP
                                ;PROGRAM WILL TYPE "R" TO INDICATE THAT TESTING HAS STARTED
                                ;AT THE END OF A PASS, PROGRAM WILL TYPE PASS COMPLETE MESSAGE
                                ;AND THEN RESUME TESTING
                                ;SUBSEQUENT RESTARTS WILL NOT TYPE PROGRAM TITLE



                                ;SWITCH REGISTER OPTIONS
                                ;----------------------
100000                          SW15=100000             ;=1,HALT ON ERROR
040000                          SW14=40000              ;=1,LOOP ON CURRENT TEST
020000                          SW13=20000              ;=1,INHIBIT ERROR TYPEOUT
010000                          SW12=10000              ;=1,DELETE TYPEOUT/BELL ON ERROR.
004000                          SW11=4000               ;=1,INHIBIT ITERATIONS
002000                          SW10=2000               ;=1,ESCAPE TO NEXT TEST ON ERROR
001000                          SW09=1000               ;=1,LOOP WITH CURRENT DATA
000400                          SW08=400                ;=1,LOOP ON ERROR
000200                          SW07=200                ;=1,USE CURRENT DMC11 PARAMETERS, =0,AUTOSIZE DMC11
000100                          SW06=100                ;=1, HALT BEFORE CLOCKING MICRO-PROCESSOR INSTRUCTION
000040                          SW05=40
000020                          SW04=20
000010                          SW03=10                 ;RESELECT DMC11'S TO BE TESTED (ACTIVE)
000004                          SW02=4                  ;LOCK ON TEST SELECT
000002                          SW01=2                  ;RESTART PROGRAM AT SELECTED TEST
000001                          SW00=1                  ;INPUT DMC11 PARAMETERS
```

```
                        ;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                  SP=%6           ;PROCESSOR STACK POINTER
000007                  PC=%7           ;PROGRAM COUNTER
                        
                        ;LOCATION EQUIVALENCIES
                        ;----------------------
                        
177776                  PS=177776       ;PROCESSOR STATUS WORD
001200                  STACK=1200      ;START OF PROCESSOR STACK
                        
                        ;INSTRUCTION DEFINITIONS
                        ;-----------------------
                        
005746                  PUSH1SP=5746    ;DECREMENT PROCESSOR STACK 1 WORD
005726                  POP1SP=5726     ;INCREMENT PROCESSOR STACK 1 WORD
010046                  PUSHR0=10046    ;SAVE R0 ON STACK
012600                  POPR0=12600     ;RESTORE R0 FROM STACK
024646                  PUSH2SP=24646   ;DECREMENT STACK TWICE
022626                  POP2SP=22626    ;INCREMENT STACK TWICE
                        .EQUIV  EMT,HLT ;BASIC DEFINITION OF ERROR CALL
                        
                        ;BIT DEFINITIONS
                        ;---------------
                        
100000                  BIT15=100000
040000                  BIT14=40000
020000                  BIT13=20000
010000                  BIT12=10000
004000                  BIT11=4000
002000                  BIT10=2000
001000                  BIT9=1000
000400                  BIT8=400
000200                  BIT7=200
000100                  BIT6=100
000040                  BIT5=40
000020                  BIT4=20
000010                  BIT3=10
000004                  BIT2=4
000002                  BIT1=2
000001                  BIT0=1
```

```
                                ;:**************************************************************************
                                ;--------------------------------------------------------------------------
                                        ;TRAPCATCAER FOR ILLEGAL INTERRUPTS
                                        ;THE STANDARD "TRAP CATCHER" IS PLACED
                                        ;BETWEEN ADDRESS 0 TO ADDRESS 776.
                                        ;IT LOOKS LIKE "PC+2 HALT".
                                ;--------------------------------------------------------------------------
                                ;:**************************************************************************

                    000000      .=0
                                ;STANDARD INTERRUPT VECTORS
                                ;-------------------------

                    000024      .=24
000024  005336                          .PFAIL                  ;POWER FAIL HANDLER
000026  000340                          340                     ;SERVICE AT LEVEL 7
000030  004750                          .HLT                    ;ERROR HANDLER
000032  000340                          340                     ;SERVICE AT LEVEL 7
000034  004716                          .TRPSRV                 ;GENERAL HANDLER DISPATCH SERVICE
000036  000340                          340                     ;SERVICE AT LEVEL 7
                    000040      .=40
000040  000000                          0                       ;SAVE FOR ACT-11 OR XXDP
000042  000000                          0                       ;RETURN ADDRESS IF UNDER ACT-11 OR XXDP
000044  000000                          0                       ;SAVE FOR ACT-11 OR XXDP
000046  003522                          $ENDAD                  ;FOR USE WITH ACT-11 OR XXDP
                    000052      .=52
000052  000000                          0                       ;ACT-11 PROGRAM CHARACTERISTICS

                    000174      .=174
000174  000000                 DISPREG:0                        ;SOFTWARE DISPLAY REGISTER
000176  000000                 SWREG:  0                        ;SOFTWARE SWITCH REGISTER

                    000200      .=200
000200  000137  002002                  JMP     .START          ;GO TO START OF PROGRAM

                    001000      .=1000
001000  005377  040515  047111 MTITLE: .ASCII  <377><12>/MAINDEC-11-CZDMG-D/<377>
(2)     001025      104  041515  030461          .ASCIZ  /DMC11 CROM AND JUMP TESTS/<377>
(2)
                    001200      .=1200

                                ;INDIRECT POINTERS TO SWITCH REGISTER AND LIGHT DISPLAY
                                ;-----------------------------------------------------

001200  177570                 DISPLAY:177570
001202  177570                 SWR:    177570
```

```
                                ;INDIRECT POINTERS TO TELETYPE VECTORS AND REGISTERS
                                ;--------------------------------------------------
001204  177560  TKCSR:  177560          ;TELETYPE KEYBOARD CONTROL REGISTER
001206  177562  TKDBR:  177562          ;TELETYPE KEYBOARD DATA BUFFER
001210  177564  TPCSR:  177564          ;TELEPRINTER CONTROL REGISTER
001212  177566  TPDBR:  177566          ;TELEPRINTER DATA BUFFER

                                ;PROGRAM CONTROL PARAMETERS
                                ;--------------------------
001214  000000  RETURN: 0              ;SCOPE ADDRESS FOR LOOP ON TEST
001216  000000  NEXT:   0              ;ADDRESS OF NEXT TEST TO BE EXECUTED
001220  000000  LOCK:   0              ;ADDRESS FOR LOCK ON CURRENT DATA
001222  000003  ICOUNT: 3              ;NUMBER OF ITERATIONS THAT CURRENT TEST WILL BE EXECUTED
001224  000000  LPCNT:  0              ;NUMBER OF ITEREATIONS COMPLETED
001226  000000  TSTNO:  0              ;NUMBER OF TEST IN PROGRESS
001230  000000  PASCNT: 0              ;NUMBER OF PASSES COMPLETED
001232  000000  ERRCNT: 0              ;TOTAL NUMBER OF ERRORS
001234  000000  LSTERR: 0              ;PC OF LAST ERROR CALL

                                ;PROGRAM VARIABLES
                                ;-----------------
001236  000000  STRTSW: 0              ;SWITCHES AT START OF PROGRAM
001240  000000  STAT:   0              ;DM STATUS WORD STORAGE
001242  000000  CLKX:   0
001244  000000  MASKX:  0
001246  000000  TEMP1:  0              ;TEMPORARY STORAGE
001250  000000  TEMP2:  0              ;TEMPORARY STORAGE
001252  000000  TEMP3:  0              ;TEMPORARY STORAGE
001254  000000  TEMP4:  0              ;TEMPORARY STORAGE
001256  000000  TEMP5:  0              ;TEMPORARY STORAGE
001260  000000  SAVR0:  0              ;R0 STORAGE
001262  000000  SAVR1:  0              ;R1 STORAGE
001264  000000  SAVR2:  0              ;R2 STORAGE
001266  000000  SAVR3:  0              ;R3 STORAGE
001270  000000  SAVR4:  0              ;R4 STORAGE
001272  000000  SAVR5:  0              ;R5 STORAGE
001274  000000  SAVSP:  0              ;STACK POINTER STORAGE
001276  000000  SAVPC:  0              ;PROGRAM COUNTER STORAGE
001300  000000  ZERO:   0
001302  000001  ONE:    1
001304  000000  MEMLIM: 0              ;HIGHEST LOCATION FOR NPR'S
001306  000001  DMACTV: .BLKW 1        ;DMC11'S SELECTED ACTIVE.
001310  000001  DMNUM:  .BLKW 1        ;OCTAL NUMBER OF DMC11'S.
001312  000001  SAVACT: .BLKW 1        ;ORIGINAL ACTV  DEVICES
001314  000001  SAVNUM: .BLKW 1        ;WORKABLE NUMBER
001316  000000  RUN:    0              ;POINTER TO RUNNING DEVICE.

                .EVEN
001320  001472  CREAM:  DM.MAP-6       ;TABLE POINTER.
001322  001676  MILK:   CNT.MAP-4      ;TABLE POINTER
```

```
                                ;PROGRAM CONTROL FLAGS
                                ;---------------------

001324  000             INIFLG: .BYTE   0               ;PROGRAM INITIALIZATION FLAG
001325  000             ERRFLG: .BYTE   0               ;ERROR OCCURED FLAG
001326  000             LOKFLG: .BYTE   0               ;LOCK ON CURRENT TEST FLAG
001327  000             QV.FLG: .BYTE   0               ;QUICK VERIFY FLAG.
                                                        ;ON FIRST PASS OF EACH DMC11 ITERATIONS WILL BE SUPPRESS

                                .EVEN

                                        ;DEFINITIONS FOR TRAP SUBROUTINE CALLS
                                        ;POINTERS TO SUBROUTINES CAN BE FOUND
                                        ;IN THE TABLE IMMEDIATLY FOLLOWING THE DEFINITIONS

                                ;:*****************************************************************
                                ;-----------------------------------------------------------------
001330                          .TRPTAB:
        104400  SCOPE=TRAP+0            ;CALL TO SCOPE LOOP AND ITERATION HANDLER
001330  003576          .SCOPE
        104401  SCOP1=TRAP+1            ;CALL TO LOOP ON CURRENT DATA HANDLER
001332  003736          .SCOP1
        104402  TYPE=TRAP+2             ;CALL TO TELETYPE OUTPUT ROUTINE
001334  003766          .TYPE
        104403  INSTR=TRAP+3            ;CALL TO ASCII STRING INPUT ROUTINE
001336  004050          .INSTR
        104404  INSTER=TRAP+4           ;CALL TO INPUT ERROR HANDLER
001340  004154          .INSTER
        104405  PARAM=TRAP+5            ;CALL TO NUMERICAL DATA INPUT ROUTINE
001342  004174          .PARAM
        104406  SAV05=TRAP+6            ;CALL TO REGISTER SAVE ROUTINE
001344  004374          .SAV05
        104407  RES05=TRAP+7            ;CALL TO REGISTER RESTORE ROUTINE
001346  004434          .RES05
        104410  CONVRT=TRAP+10          ;CALL TO DATA OUTPUT ROUTINE
001350  004466          .CONVRT
        104411  CNVRT=TRAP+11           ;CALL TO DATA OUTPUT ROUNTINE WITHOUT CR/LF.
001352  004472          .CNVRT
        104412  MSTCLR=TRAP+12          ;CALL TO ISUE A MASTER CLEAR
001354  005466          .MSTCLR
        104413  DELAY=TRAP+13           ;CALL TO DELAY
001356  005436          .DELAY
        104414  ROMCLK=TRAP+14          ;CALL TO CLOCK ROM ONCE
001360  005504          .ROMCLK
        104415  DATACLK=TRAP+15         ;CALL TO CLK DATA
001362  005552          .DATACLK
        104416  TIMER=TRAP+16           ;CALL TO DELAY A CLOCK TICK
001364  005616          .TIMER

                                ;-----------------------------------------------------------------
                                ;:*****************************************************************
```

```
                                ;DMC11 CONTROL INDICATORS FOR CURRENT DMC11 UNDER TEST
                                ;----------------------------------------------------

001366  000000                  STAT1:  0
001370  000000                  STAT2:  0
001372  000000                  STAT3:  0

                                ;DMC11 VECTOR AND REGISTER INDIRECT POINTERS
                                ;-------------------------------------------

001374  000000                  DMRVEC: 0               ;POINTER TO DMC11 RECEIVER INTERRUPT VECTOR
001376  000000                  DMRLVL: 0               ;POINTER TO DMC11 RECEIVER INTERRUPT SERVICE PS
001400  000000                  DMTVEC: 0               ;POINTER TO DMC11 TRANSMITTER INTERRUPT VECTOR
001402  000000                  DMTLVL: 0               ;POINTER TO DMC11 TRANSMITTER INTERRUPT SERVICE PS
001404  000000                  DMCSR:  0               ;POINTER TO DMC11 CONTROL STATUS REGISTER
001406  000000                  DMCSRH: 0               ;POINTER TO DMC11 CONTROL STATUS REGISTER HIGH BYTE.
001410  000000                  DMCTL:  0               ;POINTER TO DMC11 CONTOL OUT REGISTER
001412  000000                  DMP04:  0               ;POINTER TO DMC11 PORT REGISTER(SEL 4)
001414  000000                  DMP06:  0               ;POINTER TO DMC11 PORT REGISTER(SEL 6)

                                ;TEMP STORAGE
                                ;------------

001416  000000                  TEMP:   0
        001460                  .=.+40

                                ;DMC11 STATUS TABLE AND ADDRESS ASSIGNMENTS
                                ;------------------------------------------

        001500                  .=1500
001500                          DM.MAP:
001500  000001                  DMCR00: .BLKW   1       ;CONTROL STATUS REGISTER FOR DMC11 NUMBER 00
001502  000001                  DMS100: .BLKW   1       ;VECTOR FOR DMC11 NUMBER 00
001504  000001                  DMS200: .BLKW   1       ;DDCMP LINE# FOR DMC11 NUMBER 00
001506  000001                  DMS300: .BLKW   1       ;3RD STATUS WORD

001510  000001                  DMCR01: .BLKW   1       ;CONTROL STATUS REGISTER FOR DMC11 NUMBER 01
001512  000001                  DMS101: .BLKW   1       ;VECTOR FOR DMC11 NUMBER 01
001514  000001                  DMS201: .BLKW   1       ;DDCMP LINE# FOR DMC11 NUMBER 01
001516  000001                  DMS301: .BLKW   1       ;3RD STATUS WORD

001520  000001                  DMCR02: .BLKW   1       ;CONTROL STATUS REGISTER FOR DMC11 NUMBER 02
001522  000001                  DMS102: .BLKW   1       ;VECTOR FOR DMC11 NUMBER 02
001524  000001                  DMS202: .BLKW   1       ;DDCMP LINE# FOR DMC11 NUMBER 02
001526  000001                  DMS302: .BLKW   1       ;3RD STATUS WORD

001530  000001                  DMCR03: .BLKW   1       ;CONTROL STATUS REGISTER FOR DMC11 NUMBER 03
001532  000001                  DMS103: .BLKW   1       ;VECTOR FOR DMC11 NUMBER 03
001534  000001                  DMS203: .BLKW   1       ;DDCMP LINE# FOR DMC11 NUMBER 03
001536  000001                  DMS303: .BLKW   1       ;3RD STATUS WORD

001540  000001                  DMCR04: .BLKW   1       ;CONTROL STATUS REGISTER FOR DMC11 NUMBER 04
001542  000001                  DMS104: .BLKW   1       ;VECTOR FOR DMC11 NUMBER 04
001544  000001                  DMS204: .BLKW   1       ;DDCMP LINE# FOR DMC11 NUMBER 04
001546  000001                  DMS304: .BLKW   1       ;3RD STATUS WORD
```

```
 304  001550  000001  DMCR05: .BLKW   1          ;CONTROL STATUS REGISTER FOR DMC11 NUMBER 05
 305  001552  000001  DMS105: .BLKW   1          ;VECTOR FOR DMC11 NUMBER 05
 306  001554  000001  DMS205: .BLKW   1          ;DDCMP LINE# FOR DMC11 NUMBER 05
 307  001556  000001  DMS305: .BLKW   1          ;3RD STATUS WORD
 308
 309  001560  000001  DMCR06: .BLKW   1          ;CONTROL STATUS REGISTER FOR DMC11 NUMBER 06
 310  001562  000001  DMS106: .BLKW   1          ;VECTOR FOR DMC11 NUMBER 06
 311  001564  000001  DMS206: .BLKW   1          ;DDCMP LINE# FOR DMC11 NUMBER 06
 312  001566  000001  DMS306: .BLKW   1          ;3RD STATUS WORD
 313
 314  001570  000001  DMCR07: .BLKW   1          ;CONTROL STATUS REGISTER FOR DMC11 NUMBER 07
 315  001572  000001  DMS107: .BLKW   1          ;VECTOR FOR DMC11 NUMBER 07
 316  001574  000001  DMS207: .BLKW   1          ;DDCMP LINE# FOR DMC11 NUMBER 07
 317  001576  000001  DMS307: .BLKW   1          ;3RD STATUS WORD
 318
 319  001600  000001  DMCR10: .BLKW   1          ;CONTROL STATUS REGISTER FOR DMC11 NUMBER 10
 320  001602  000001  DMS110: .BLKW   1          ;VECTOR FOR DMC11 NUMBER 10
 321  001604  000001  DMS210: .BLKW   1          ;DDCMP LINE# FOR DMC11 NUMBER 10
 322  001606  000001  DMS310: .BLKW   1          ;3RD STATUS WORD
 323
 324  001610  000001  DMCR11: .BLKW   1          ;CONTROL STATUS REGISTER FOR DMC11 NUMBER 11
 325  001612  000001  DMS111: .BLKW   1          ;VECTOR FOR DMC11 NUMBER 11
 326  001614  000001  DMS211: .BLKW   1          ;DDCMP LINE# FOR DMC11 NUMBER 11
 327  001616  000001  DMS311: .BLKW   1          ;3RD STATUS WORD
 328
 329  001620  000001  DMCR12: .BLKW   1          ;CONTROL STATUS REGISTER FOR DMC11 NUMBER 12
 330  001622  000001  DMS112: .BLKW   1          ;VECTOR FOR DMC11 NUMBER 12
 331  001624  000001  DMS212: .BLKW   1          ;DDCMP LINE# FOR DMC11 NUMBER 12
 332  001626  000001  DMS312: .BLKW   1          ;3RD STATUS WORD
 333
 334  001630  000001  DMCR13: .BLKW   1          ;CONTROL STATUS REGISTER FOR DMC11 NUMBER 13
 335  001632  000001  DMS113: .BLKW   1          ;VECTOR FOR DMC11 NUMBER 13
 336  001634  000001  DMS213: .BLKW   1          ;DDCMP LINE# FOR DMC11 NUMBER 13
 337  001636  000001  DMS313: .BLKW   1          ;3RD STATUS WORD
 338
 339  001640  000001  DMCR14: .BLKW   1          ;CONTROL STATUS REGISTER FOR DMC11 NUMBER 14
 340  001642  000001  DMS114: .BLKW   1          ;VECTOR FOR DMC11 NUMBER 14
 341  001644  000001  DMS214: .BLKW   1          ;DDCMP LINE# FOR DMC11 NUMBER 14
 342  001646  000001  DMS314: .BLKW   1          ;3RD STATUS WORD
 343
 344  001650  000001  DMCR15: .BLKW   1          ;CONTROL STATUS REGISTER FOR DMC11 NUMBER 15
 345  001652  000001  DMS115: .BLKW   1          ;VECTOR FOR DMC11 NUMBER 15
 346  001654  000001  DMS215: .BLKW   1          ;DDCMP LINE# FOR DMC11 NUMBER 15
 347  001656  000001  DMS315: .BLKW   1          ;3RD STATUS WORD
 348
 349  001660  000001  DMCR16: .BLKW   1          ;CONTROL STATUS REGISTER FOR DMC11 NUMBER 16
 350  001662  000001  DMS116: .BLKW   1          ;VECTOR FOR DMC11 NUMBER 16
 351  001664  000001  DMS216: .BLKW   1          ;DDCMP LINE# FOR DMC11 NUMBER 16
 352  001666  000001  DMS316: .BLKW   1          ;3RD STATUS WORD
 353
 354  001670  000001  DMCR17: .BLKW   1          ;CONTROL STATUS REGISTER FOR DMC11 NUMBER 17
 355  001672  000001  DMS117: .BLKW   1          ;VECTOR FOR DMC11 NUMBER 17
 356  001674  000001  DMS217: .BLKW   1          ;DDCMP LINE# FOR DMC11 NUMBER 17
 357  001676  000001  DMS317: .BLKW   1          ;3RD STATUS WORD
 358
 359  001700  000000  DM.END: 000000
```

```
                        ;DMC11 PASS COUNT AND ERROR COUNT TABLE
                        ;-------------------------------------

001702                  CNT.MAP:
001702  000000          PACT00: 0               ;PASS COUNT FOR DMC11 NUMBER 00
001704  000000          ERCT00: 0               ;ERROR COUNT FOR DMC11 NUMBER 00

001706  000000          PACT01: 0               ;PASS COUNT FOR DMC11 NUMBER 01
001710  000000          ERCT01: 0               ;ERROR COUNT FOR DMC11 NUMBER 01

001712  000000          PACT02: 0               ;PASS COUNT FOR DMC11 NUMBER 02
001714  000000          ERCT02: 0               ;ERROR COUNT FOR DMC11 NUMBER 02

001716  000000          PACT03: 0               ;PASS COUNT FOR DMC11 NUMBER 03
001720  000000          ERCT03: 0               ;ERROR COUNT FOR DMC11 NUMBER 03

001722  000000          PACT04: 0               ;PASS COUNT FOR DMC11 NUMBER 04
001724  000000          ERCT04: 0               ;ERROR COUNT FOR DMC11 NUMBER 04

001726  000000          PACT05: 0               ;PASS COUNT FOR DMC11 NUMBER 05
001730  000000          ERCT05: 0               ;ERROR COUNT FOR DMC11 NUMBER 05

001732  000000          PACT06: 0               ;PASS COUNT FOR DMC11 NUMBER 06
001734  000000          ERCT06: 0               ;ERROR COUNT FOR DMC11 NUMBER 06

001736  000000          PACT07: 0               ;PASS COUNT FOR DMC11 NUMBER 07
001740  000000          ERCT07: 0               ;ERROR COUNT FOR DMC11 NUMBER 07

001742  000000          PACT10: 0               ;PASS COUNT FOR DMC11 NUMBER 10
001744  000000          ERCT10: 0               ;ERROR COUNT FOR DMC11 NUMBER 10

001746  000000          PACT11: 0               ;PASS COUNT FOR DMC11 NUMBER 11
001750  000000          ERCT11: 0               ;ERROR COUNT FOR DMC11 NUMBER 11

001752  000000          PACT12: 0               ;PASS COUNT FOR DMC11 NUMBER 12
001754  000000          ERCT12: 0               ;ERROR COUNT FOR DMC11 NUMBER 12

001756  000000          PACT13: 0               ;PASS COUNT FOR DMC11 NUMBER 13
001760  000000          ERCT13: 0               ;ERROR COUNT FOR DMC11 NUMBER 13

001762  000000          PACT14: 0               ;PASS COUNT FOR DMC11 NUMBER 14
001764  000000          ERCT14: 0               ;ERROR COUNT FOR DMC11 NUMBER 14

001766  000000          PACT15: 0               ;PASS COUNT FOR DMC11 NUMBER 15
001770  000000          ERCT15: 0               ;ERROR COUNT FOR DMC11 NUMBER 15

001772  000000          PACT16: 0               ;PASS COUNT FOR DMC11 NUMBER 16
001774  000000          ERCT16: 0               ;ERROR COUNT FOR DMC11 NUMBER 16

001776  000000          PACT17: 0               ;PASS COUNT FOR DMC11 NUMBER 17
002000  000000          ERCT17: 0               ;ERROR COUNT FOR DMC11 NUMBER 17
```

FORMAT OF STATUS TABLE

15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00	
C	O	N	T	R	O	L		R	E	G	I	S	T	E	R	CSR
*	*	*	*	*	*	*	*	*	V	E	C	T	O	R	*	STAT1
*	B	M		A	D	D	*	*	L	I	N	E		#	*	STAT2
														*	*	STAT3

DEFINITION OF FORMAT

CSR: CONTAINS DMC11 CSR ADDRESS

STAT1: BITS 00-08 IS DMC11 VECTOR ADDRESS
BIT15=1 MICRO-PROCESSOR HAS CRAM
BIT15=0 MICRO-PROCESSOR HAS CROM
BIT14=1 ???? TURNAROUND CONNECTOR IS ON
BIT14=0 NO TURNAROUND CONNECTOR
BIT13=0 LINE UNIT IS AN M8201
BIT13=1 LINE UNIT IS AN M8202
BIT12=1 NO LINE UNIT
BITS 09-11 IS DMC11 BR PRIORITY LEVEL

STAT2: LOW BYTE IS SWITCH PAC#1 (DDCMP LINE NUMBER)
HIGH BYTE IS SWITCH PAC#2 (BM873 BOOT ADD)

STAT3: BIT0=1 DO FREE RUNNING TESTS ON KMC
(MUST BE SET TO A ONE MANUALLY [PROGRAM DZDMI ONLY])
KMC MUST HAVE MICRO-CODE WRITTEN FROM RUNNING
DZDMG TEST 2 FIRST
BIT1=1 DMC11-AL LOCAL HIGH SPEED MICRO-CODE
BIT1=0 DMC11-AR REMOTE LOW SPEED MICRO-CODE

```
                                                ;PROGRAM INITIALIZATION
                                                ;LOCK OUT INTERRUPTS
                                                ;SET UP PROCESSOR STACK
                                                ;SET UP POWER FAIL VECTOR
                                                ;CLEAR PROGRAM CONTROL FLAGS AND COUNTS
                                                ;TYPE TITLE MESSAGE
002002  012737  000340  177776  .START: MOV     #340,PS         ;LOCK OUT INTERRUPTS
002010  012706  001200                  MOV     #STACK,SP       ;SET UP STACK
002014  012737  005336  000024          MOV     #.PFAIL,@#24    ;SET UP POWER FAIL VECTOR
002022  013737  001310  001314          MOV     DMNUM,SAVNUM    ;SAVE NUMBER OF DEVICES IN SYSTEM.
002030  005037  010020                  CLR     SWFLG           ;CLEAR SOFT TYPEOUT FLAG
002034  105037  001325                  CLRB    ERRFLG          ;CLEAR ERROR FLAG
002040  105037  001327                  CLRB    QV.FLG          ;ZERO QUICK VERIFY FLAG
002044  012737  001470  001320          MOV     #DM.MAP-10,CREAM;GET MAP POINTER.
002052  012737  001676  001322          MOV     #CNT.MAP-4,MILK ;GET PASS COUNT MAP POINTER
002060  012737  100000  001316          MOV     #BIT15,RUN      ;POINT POINTER TO FIRST DEVICE.
002066  012700  001702                  MOV     #CNT.MAP,R0     ;PASS COUNT POINTER TO R0
002072  005020                  23$:     CLR     (R0)+           ;CLEAR TABLE
002074  022700  002002                  CMP     #CNT.MAP+100,R0 ;DONE YET?
002100  001374                          BNE     23$             ;KEEP GOING
002102  005037  001234                  CLR     LSTERR          ;CLEAR LAST ERROR POINTER
002106  012737  000001  001226          MOV     #1,TSTNO        ;SET UP FOR TEST 1
002114  012737  002002  001214          MOV     #.START,RETURN  ;SET UP FOR POWER FAIL BEFORE
                                                                ;TESTING STARTS
002122  013746  000006                  MOV     @#6,-(SP)       ;SAVE CURRENT VECTORS
002126  013746  000004                  MOV     @#4,-(SP)       ;
002132  012737  002166  000004          MOV     #6$,@#4         ;SET UP FOR TIMEOUT
002140  012737  177570  001202          MOV     #177570,SWR     ;SET SWR TO HARD SWR ADDRESS
002146  012737  177570  001200          MOV     #177570,DISPLAY ;SET DISPLAY TO HARD SWR ADDRESS
002154  022777  177777  177020          CMP     #-1,@SWR        ;REFERENCE HARDWARE SWITCH REGISTER
002162  001402                          BEQ     6$+2            ;IF = -1 USE SOFT SWR ANYWAY
002164  000407                          BR      7$              ;IF IT EXISTS AND NOT = -1 USE HARD SWR
002166  022626                  6$:     CMP     (SP)+,(SP)+     ;ADJUST STACK
002170  012737  000176  001202          MOV     #SWREG,SWR      ;POINTER TO SOFT SWR
002176  012737  000174  001200          MOV     #DISPREG,DISPLAY;POINTER TO SOFT DISPLAY REG
002204  012637  000004          7$:     MOV     (SP)+,@#4       ;RESTORE VECTORS
002210  012637  000006                  MOV     (SP)+,@#6       ;
002214  105737  001324                  TSTB    INIFLG          ;HAS INITIALIZATION BEEN PERFORMED
002220  001006                          BNE     20$             ;BR IF YES
002222  022737  003522  000042          CMP     #$ENDAD,@#42    ;IF ACT-11 AUTOMATIC MODE, DON'T TYPE ID
002230  001402                          BEQ     20$
002232  104402  001000                  TYPE    ,MTITLE         ;TYPE TITLE MESSAGE
002236  004737  007610          20$:    JSR     PC,CKSWR        ;CHECK FOR SOFT SWR
002242  017737  176734  001236          MOV     @SWR,STRTSW     ;STORE STARTING SWITCHES
002250  005737  000042                  TST     @#42            ;IS IT RUNNING IN AUTO MODE?
002254  001402                          BEQ     .+6             ;BR IF NO
002256  005037  001236                  CLR     STRTSW          ;IF YES, CLEAR SWITCHES
002262  032737  000001  001236          BIT     #SW00,STRTSW    ;IF SW00=1, QUESTIONS ARE ASKED.
002270  001012                          BNE     17$             ;BR IF SW00=1
002272  105737  001236                  TSTB    STRTSW          ;BIT7=1??
002276  100007                          BPL     17$             ;BR IF SW07=0
002300  005737  001306                  TST     DMACTV          ;ARE ANY DEVICES SELECTED?
002304  001006                          BNE     16$             ;BR IF YES
002306  104402  007155                  TYPE,   NOACT           ;NO DEVICES SELECTED.
```

```
                        HALT                    ;STOP THE SHOW
                        BR      .-2             ;DISQUALIFY CONTINUE SWITCH
17$:                    JSR     PC,AUTO.SIZE    ;GO DO THE AUTO SIZE
16$:                    TSTB    INIFLG          ;FIRST TIME?
                        BEQ     21$             ;BR IF YES
                        TSTB    STRTSW          ;IF USING SAME PARAMETERS DONT TYPE MAP
                        BMI     1$
                        BIT     #BIT1!BIT2,STRTSW;IS TEST NO. OR LOCK SELECTED
                        BEQ     24$             ;IF NO THEN TYPE STATUS
                        BR      1$              ;IF YES DO NOT TYPE STATUS
21$:                    COM     INIFLG          ;SET FLAG
24$:                    TYPE    ,XHEAD          ;TYPE HEADER
                        MOV     #DM.MAP,R4      ;SET POINTER
5$:                     MOV     R4,TEMP1        ;SET ADDRESS
                        MOV     (R4)+,TEMP2     ;SET CSR
                        BEQ     1$              ;ALL DONE IF ZERO
                        MOV     (R4)+,TEMP3     ;SET STAT1
                        MOV     (R4)+,TEMP4     ;SET STAT2
                        MOV     (R4)+,TEMP5     ;SET STAT3
                        CONVRT                  ;TYPE OUT STATUS MAP
                        XSTATQ                  ;
                        BR      5$
1$:                     MOV     #DM.MAP,R0      ;R0 POINTS TO STATUS TABLE

;;*************************************************************************
;;*AUTO SIZE TEST
;;*THIS TEST VERIFYS THAT THE DMC11S AND/OR KMC11S ARE AT THE CORRECT FLOATING
;;*ADDRESSES FOR YOUR SYSTEM. IF THIS TEST FAILS, IT IS NOT A HARDWARE ERROR.
;;*CHECK THE ADDRESSES OF ALL FLOATING DEVICES (DJ,DH,DQ,DU,DUP,LK,DMC,DZ,KMC).
;;*IF THERE ARE NO OTHER FLOATING DEVICES BEFORE THE DMC11, THE FIRST
;;*DMC11 ADDRESS IS 760070, KMC11 IS 760110. NO DEVICE SHOULD EVER BE AT
;;*ADDRESS 760000. THIS TEST MAY REQUIRE 2 OR MORE ATTEMPTS TO GET THE
;;*RIGHT ADDRESSES. AFTER YOU HAVE CHANGED THE ADDRESS TO WHAT IT TOLD
;;*YOU THE FIRST TIME, IT MAY COME BACK AND TELL YOU A DIFFERENT ADDRESS
;;*THE NEXT TIME YOU RUN IT. PLEASE HAVE PATIENCE, THE FINAL ADDRESS
;;*WILL BE CORRECT (AS LONG AS ALL DEVICES IN FRONT OF THE DMC'S ARE
;;*CORRECT).
;;*************************************************************************

                        MOV     @#4,-(SP)       ;SAVE LOC 4
                        MOV     @#6,-(SP)       ;SAVE LOC 6
                        CLR     @#6             ;CLEAR VEC+2
                        CLR     TEMP3           ;CLEAR FLAG
                        CLR     R5              ;R5=0=DMC, R5=-1=KMC
AUSTRT:                 MOV     (R0),DMCSR      ;GET NEXT DMC CSR
                        BEQ     AUDONE          ;BR IF DONE
                        TST     R5              ;DMC OR KMC?
                        BNE     1$              ;BR IF KMC
                        BIT     #BIT15,2(R0)    ;CHECK FOR DMC CSR
                        BNE     SKIP            ;SKIP IF NOT DMC
                        BR      2$              ;ITS A DMC SO CONTINUE
1$:                     BIT     #BIT15,2(R0)    ;CHECK FOR KMC CSR
                        BEQ     SKIP            ;SKIP IF NOT KMC
2$:                     MOV     #NODEV,@#4      ;SET UP FOR TIMEOUT
                        TST     R5              ;DMC OR KMC?
                        BNE     3$              ;BR IF KMC
```

```
 580  002514  012703  000006                MOV     #6,R3           ;R3 IS COUNT OF DEVICES BEFORE DMC
 581  002520  000402                        BR      4$              ;GO ON
 582  002522  012703  000010          3$:   MOV     #10,R3          ;R3 IS COUNT OF DEVICES BEFORE KMC
 583  002526  012702  003010          4$:   MOV     #DEVTAB,R2      ;R2 IS DEVICE TABLE PONTER
 584  002532  012701  160010                MOV     #160010,R1      ;START WITH ADDRESS 160010
 585  002536  005711                FLOAT:  TST     (R1)            ;CHECK ADDRESS IN R1
 586  002540  111204                        MOVB    (R2),R4         ;IF NO TIMEOUT, GET NEXT ADDRESS
 587  002542  060401                        ADD     R4,R1           ;IN R1
 588  002544  005201                        INC     R1              ;
 589  002546  040401                        BIC     R4,R1           ;
 590  002550  005703                        TST     R3              ;ANY MORE DEVICES TO CHECK FOR?
 591  002552  001371                        BNE     FLOAT           ;BR IF YES
 592  002554  012737  002700  000004        MOV     #ERR,@#4        ;OK ONLY DMC'S ARE LEFT, SET UP FOR TIMEOUT
 593  002562  010137  003022                MOV     R1,XLOC         ;SAVE FIRST DMC/KMC ADDRESS
 594  002566  005705                FY:     TST     R5              ;DMC OR KMC?
 595  002570  001005                        BNE     1$              ;BR IF KMC
 596  002572  032760  100000  000002        BIT     #BIT15,2(R0)    ;CHECK FOR DMC CSR
 597  002600  001014                        BNE     SKIP            ;SKIP IF NOT DMC
 598  002602  000404                        BR      2$              ;ITS A DMC SO CONTINUE
 599  002604  032760  100000  000002  1$:   BIT     #BIT15,2(R0)    ;CHECK FOR KMC CSR
 600  002612  001407                        BEQ     SKIP            ;SKIP IF NOT KMC
 601  002614  005711                2$:     TST     (R1)            ;CHECK DMC ADDRESS
 602  002616  020137  001404                CMP     R1,DMCSR        ;DOES IT MATCH
 603  002622  001411                        BEQ     OK              ;BR IF YES
 604  002624  062701  000010                ADD     #10,R1          ;GET NEXT DMC ADDRESS
 605  002630  000756                        BR      FY              ;DO IT AGAIN
 606  002632  062700  000010          SKIP: ADD     #10,R0          ;SKIP TO NEXT CSR IN TABLE
 607  002636  011037  001404                MOV     (R0),DMCSR      ;GET NEXT CSR
 608  002642  001470                        BEQ     AUDONE          ;BR IF DONE
 609  002644  000750                        BR      FY              ;ELSE CONTINUE
 610  002646  062700  000010          OK:   ADD     #10,R0          ;SKIP TO NEXT DMC CSR
 611  002652  062737  000010  003022        ADD     #10,XLOC        ;UPDATE EXPECTED DMC/KMC ADDRESS
 612  002660  011037  001404                MOV     (R0),DMCSR      ;GET NEXT DMC/KMC CSR
 613  002664  001457                        BEQ     AUDONE          ;BR IF DONE
 614  002666  013701  003022                MOV     XLOC,R1         ;GET EXPECTED DMC/KMC ADDRESS
 615  002672  000735                        BR      FY              ;CONTINUE
 616  002674  122243                NODEV:  CMPB    (R2)+,-(R3)     ;ON TIMEOUT, INC R2, DEC R3
 617  002676  000002                        RTI                     ;RETURN
 618  002700  005737  001252          ERR:  TST     TEMP3           ;CHECK FLAG IF = 0 TYPE HEADER
 619  002704  001014                        BNE     1$              ;SKIP HEADER
 620  002706  104402                        TYPE                    ;TYPEOUT HEADER MESSAGE
 621  002710  007224                        CONERR                  ;CONFIGURATION ERROR!!!!
 622  002712  012737  002700  001276        MOV     #ERR,SAVPC      ;SAVE PC FOR TYPEOUT
 623  002720  104411                        CNVRT                   ;TYPE OUT ERROR PC
 624  002722  002770                        ERRPC                   ;
 625  002724  104402                        TYPE                    ;TYPE REST OF HEADER
 626  002726  007300                        CNERR                   ;
 627  002730  012737  177777  001252        MOV     #-1,TEMP3       ;SET FLAG SO IT ONLY GETS TYPED ONCE
 628  002736  010137  001262          1$:   MOV     R1,SAVR1        ;SAVE R1 FOR TYPEOUT
 629  002742  104410                        CONVRT                  ;
 630  002744  002776                        CONTAB                  ;TYPE CSR VALUES
 631  002746  005705                        TST     R5              ;DMC OR KMC ?
 632  002750  001003                        BNE     3$              ;BR IF KMC
 633  002752  104402                        TYPE                    ;
 634  002754  007321                        DMCM                    ;
 635  002756  000402                        BR      4$              ;CONTINUE
```

```
636  002760  104402                 3$:     TYPE
637  002762  007331                         KMCM
638  002764  022626                 4$:     CMP     (SP)+,(SP)+     ;ADJUST STACK
639  002766  000727                         BR      OK              ;BR TO GET OUT
640  002770  000001                 ERRPC:  1
641  002772     006      002                .BYTE   6,2
642  002774  001276                         SAVPC
643  002776  000002                 CONTAB: 2
644  003000     006      004                .BYTE   6,4
645  003002  003022                         XLOC
646  003004     006      002                .BYTE   6,2
647  003006  001404                         DMCSR
648  003010     007                 DEVTAB: .BYTE   7               ;DJ
649  003011     017                         .BYTE   17              ;DH
650  003012     007                         .BYTE   7               ;DQ
651  003013     007                         .BYTE   7               ;DU
652  003014     007                         .BYTE   7               ;DUP
653  003015     007                         .BYTE   7               ;LK
654  003016     007                         .BYTE   7               ;DMC
655  003017     007                         .BYTE   7               ;DZ
656  003020     007                         .BYTE   7               ;KMC
657       003022                 .EVEN
658  003022  000000                 XLOC:   0
659  003024  005705                 AUDONE: TST     R5              ;DMC?
660  003026  001005                         BNE     1$              ;BR IF KMC AND ALL DONE
661  003030  012705  177777                 MOV     #-1,R5          ;SET R5 TO -1 (KMC)
662  003034  012700  001500                 MOV     #DM.MAP,R0      ;RESET R0 TO START OF TABLE
663  003040  000602                         BR      AUSTRT          ;GO DO KMC'S
664  003042  012637  000006         1$:     MOV     (SP)+,@#6       ;RESTORE LOC 6
665  003046  012637  000004                 MOV     (SP)+,@#4       ;RESTORE LOC 4
666  003052  032737  000010  001236         BIT     #SW03,STRTSW    ;SELECT SPECIFIC DEVICES??
667  003060  001422                         BEQ     3$              ;BR IF NO.
668  003062  104402  006145                 TYPE    ,MNEW           ;TYPE THE MESSAGE.
669  003066  005000                         CLR     R0              ;ZERO DATA LIGHTS
670  003070  000000                         HALT                    ;WAIT FOR USER TO TELL WHAT DEVICES TO RUN
671  003072  027737  176104  001312         CMP     @SWR,SAVACT     ;IS THE NUMBER VALID?
672  003100  101404                         BLOS    2$              ;BR IF NUMBER IS OK.
673  003102  104402  006006                 TYPE    ,MERR3          ;TELL USER OF INVALID NUMBER.
674  003106  000000                         HALT                    ;STOP EVERY THING.
675  003110  000776                         BR      .-2             ;RESTART THE PROGRAM AGAIN.
676  003112  017737  176064  001306 2$:     MOV     @SWR,DMACTV     ;GET NEW DEVICE PATTERN
677  003120  013700  001306                 MOV     DMACTV,R0       ;SHOW THE USER WHAT HE SELECTED.
678  003124  000000                         HALT                    ;CONTINUE DYNAMIC SWITCHES.
679  003126  012700  000300         3$:     MOV     #300,R0         ;PREPARE TO CLEAR THE FLOATING
680  003132  012701  000302                 MOV     #302,R1         ;VECTOR AREA. 300-776
681  003136  010120                 4$:     MOV     R1,(R0)+        ;START PUTTING "PC+2 - HALT"
682  003140  005021                         CLR     (R1)+           ;IN VECTOR AREA.
683  003142  022021                         CMP     (R0)+,(R1)+     ;POP POINTERS
684  003144  022700  001000                 CMP     #1000,R0        ;ALL DONE??
685  003150  001372                         BNE     4$              ;BR IF NO.
686
687                                 ;TEST START AND RESTART
688                                 ;----------------------
689
690  003152  012706  001200         .BEGIN: MOV     #STACK,SP       ;SET UP STACK
691  003156  013746  000006                 MOV     @#6,-(SP)       ;SAVE LOC 6
```

F03


```
 692  003162  013746  000004                 MOV     @#4,-(SP)       ;SAVE LOC 4
 693  003166  005000                         CLR     R0              ;START AT 0
 694  003170  012737  003234  000004         MOV     #2$,@#4         ;SET UP FOR TIME OUT
 695  003176  005037  000006                 CLR     @#6             ;TO AUTOSIZE MEMORY
 696  003202  005720                 6$:     TST     (R0)+           ;CHECK ADDRESS IN R0
 697  003204  022700  157776                 CMP     #157776,R0      ;IS IT AT LEAST 28K
 698  003210  001374                         BNE     6$              ;BR IF NO
 699  003212  162700  007776                 SUB     #7776,R0        ;SAVE 2K FOR MONITORS
 700  003216  010037  001304         7$:     MOV     R0,MEMLIM       ;STORE MEMORY LIMIT
 701  003222  012637  000004                 MOV     (SP)+,@#4       ;RESTORE LOC 4
 702  003226  012637  000006                 MOV     (SP)+,@#6       ;RESTORE LOC 6
 703  003232  000413                         BR      10$             ;CONTINUE
 704  003234  022626                 2$:     CMP     (SP)+,(SP)+     ;ADJUST STACK
 705  003236  162700  000004                 SUB     #4,R0           ;GET LAST GOOD ADDRESS
 706  003242  162700  007776                 SUB     #7776,R0        ;SAVE 2K FOR MONITORS
 707  003246  022700  030000                 CMP     #30000,R0       ;IS IT 8K?
 708  003252  001361                         BNE     7$              ;BR IF NO
 709  003254  012700  037400                 MOV     #37400,R0       ;IF 8K DON'T SAVE 2K
 710  003260  000756                         BR      7$              ;
 711  003262  012737  000340  177776 10$:    MOV     #340,PS         ;LOCK OUT INTERRUPTS
 712  003270  032737  000004  001236         BIT     #BIT2,STRTSW    ;CHECK FOR LOCK ON TEST
 713  003276  001411                         BEQ     1$              ;BR IF NO LOCK DESIRED.
 714  003300  104402  006044                 TYPE    ,MLOCK          ;TYPE LOCK SELECTED.
 715  003304  012737  000240  003612         MOV     #NOP,TTST       ;ADJUST SCOPE ROUTINE.
 716  003312  012737  000240  003614         MOV     #NOP,TTST+2     ;SET UP TO LOCK
 717  003320  000406                         BR      3$              ;CONTINUE ALONG.
 718  003322  013737  003730  003612 1$:     MOV     BRW,TTST        ;PREPARE NORMAL SCOPE ROUTINE
 719  003330  013737  003732  003614         MOV     BRX,TTST+2      ;LOCK NOT SELECTED, SET UP FOR NORMAL SCOPE LOOP
 720  003336  012737  010062  001214 3$:     MOV     #CYCLE,RETURN   ;START AT "CYCLE" FIND WHICH DEVICE TO TEST
 721  003344  032737  000002  001236 4$:     BIT     #SW01,STRTSW    ;IS TEST NO. SELECTED?
 722  003352  001002                         BNE     5$              ;BR IF YES
 723  003354  104402  005756                 TYPE    ,MR             ;TYPE R
 724  003360  000177  175630         5$:     JMP     @RETURN         ;START TESTING
```

```
                                                ;END OF PASS
                                                ;TYPE NAME OF TEST
                                                ;UPDATE PASS COUNT
                                                ;CHECK FOR EXIT TO ACT-11
                                                ;RESTART TEST
                                                
003364  000005                          .EOP:   RESET                   ;MAKE THE  WORLD CLEAN AGAIN.
003366  005037  001234                          CLR     LSTERR          ;CLEAR LAST ERROR PC
003372  105037  001325                          CLRB    ERRFLG          ;CLEAR ERROR FLAG
003376  005237  001230                          INC     PASCNT          ;UPDATE PASS COUNT
003402  013777  001230  175570                  MOV     PASCNT,@DISPLAY ;DISPLAY PASS COUNT
003410  104402  005733                          TYPE    ,MEPASS         ;TYPE END PASS
003414  104402  006073                          TYPE    ,MCSRX          ;TYPE CSR
003420  104411  003546                          CNVRT   ,XCSR           ;SHOW IT
003424  104402  006101                          TYPE    ,MVECX          ;TYPE VECTOR
003430  104411  003554                          CNVRT   ,XVEC           ;SHOW IT
003434  104402  006107                          TYPE    ,MPASSX         ;TYPE PASSES
003440  104411  003562                          CNVRT   ,XPASS          ;SHOW IT
003444  104402  006120                          TYPE    ,MERRX          ;TYPE ERRORS
003450  104411  003570                          CNVRT   ,XERR           ;SHOW IT
003454  013700  001322                          MOV     MILK,R0         ;GET POINTER TO PASS COUNT
003460  013720  001230                          MOV     PASCNT,(R0)+    ;STORE PASS COUNT FOR THIS DMC11
003464  013720  001232                          MOV     ERRCNT,(R0)+    ;STORE ERROR COUNT FOR THIS DMC11
003470  005337  001314                          DEC     SAVNUM          ;ARE ALL DEVICES TESTED?
003474  001017                                  BNE     RESTRT          ;BR IF NO.
003476  112737  000377  001327                  MOVB    #377,QV.FLG     ;SET THE QUICK VERIFY FLAG.
003504  013737  001310  001314                  MOV     DMNUM,SAVNUM    ;RESTORE THE COUNT
003512  013701  000042                          MOV     @#42,R1         ;CHECK FOR ACT-11 OR DDP
003516  001406                                  BEQ     RESTRT          ;IF NOT, CONTINUE TESTING
003520  000005                                  RESET                   ;STOP THE SHOW--CLEAR THE WORLD
003522                                  $ENDAD:
003522  004711                                  JSR     PC,(R1)
003524  000240                                  NOP
003526  000240                                  NOP
003530  000240                                  NOP
003532  000240                                  NOP
003534  012737  010062  001214          RESTRT: MOV     #CYCLE,RETURN
003542  000137  010062                          JMP     CYCLE
003546  000001                          XCSR:   1
003550     006     002                          .BYTE   6,2
003552  001404                                  DMCSR
003554  000001                          XVEC:   1
003556     004     002                          .BYTE   4,2
003560  001374                                  DMRVEC
003562  000001                          XPASS:  1
003564     006     002                          .BYTE   6,2
003566  001230                                  PASCNT
003570  000001                          XERR:   1
003572     006     002                          .BYTE   6,2
003574  001232                                  ERRCNT
                                                
                                                ;SCOPE LOOP AND INTERATION HANDLER
                                                ;---------------------------------
                                                
003576  004737  007610                  .SCOPE: JSR     PC,CKSWR        ;CKECK FOR SOFT SWR
003602  010016                                  MOV     R0,(SP)         ;SAVE R0 ON THE STACK
```

```
                                        BIT     #BIT14,@SWR     ;"LOOP ON THIS TEST"?
                                TTST:   BEQ     1$              ;BR IF NO.  (IF LOCK SW01=1; THIS LOC =240)
                                        BR      3$              ;GOTO 3$    (IF LOCK SW01=1; THIS LOC =240)
                                        TST     DONE            ;WAS TKCSR DONE SET?
                                        BEQ     3$              ;BR IF NO (LOCKED ON TEST)
                                        CLR     DONE            ;YES, CLEAR FLAG
                                        BR      2$              ;GO TO NEXT TEST
                                1$:     BIT     #SW11,@SWR      ;DELETE ITERATION?  (QUICK PASS)
                                        BNE     2$              ;BR IF YES
                                        TSTB    QV.FLG          ;HAVE PASSES BEECOMPLETED?
                                        BEQ     2$              ;BR IF QUICK PASS.
                                        INC     LPCNT           ;UPDATE ITERATION COUNTER
                                        CMP     LPCNT,ICOUNT    ;ARE ALL ITERATIONS DONE??
                                        BLOS    3$              ;BR IF NOT YET
                                2$:     CLRB    ERRFLG          ;PREPARE FOR NEW TEST
                                        CLR     LPCNT           ;START ICOUNTER AT 0
                                        CLR     LOCK
                                        MOV     #20,ICOUNT      ;RESET ITERATIONS
                                        MOV     NEXT,RETURN     ;GET NEXT TEST
                                3$:     MOV     (SP),R0         ;POP R0 OFF OF THE STACK
                                        POP2SP                  ;FAKE AN "RTI"
                                        MOV     DMCSR,R1        ;R1 CONTAINS BASE DMC ADDRESS
                                        JMP     @RETURN         ;GO DO THE TEST
                                BRW:    1407
                                BRX:    437
                                DONE:   0

                                        ;CHECK FOR FREEZE ON CURRENT DATA
                                        ;--------------------------------

                                .SCOP1: JSR     PC,CKSWR        ;CHECK FOR SOFT SWR
                                        BIT     #SW09,@SWR      ;IS SW09=1(SET)?
                                        BEQ     1$              ;BR IF NOT SET.
                                        TST     LOCK
                                        BEQ     1$
                                        MOV     LOCK,(SP)       ;GOTO THE ADDRESS IN LOCK.
                                1$:     RTI                     ;GO BACK.

                                        ;TELETYPE OUTPUT ROUTINE
                                        ;-----------------------

                                .TYPE:  MOV     R5,-(SP)        ;SAVE R5 ON THE STACK.
                                        MOV     @2(SP),R5       ;GET ADDRESS OF MESSAGE.
                                        ADD     #2,2(SP)        ;POP OVER ADDRESS.
                                4$:     TST     SWFLG           ;SOFT SWR MESSAGE?
                                        BNE     1$              ;IF YES TYPE IT OUT REGARDLESS OF SW12
                                        BIT     #SW12,@SWR      ;INHIBIT ALL PRINT OUT??
                                        BNE     3$              ;BR IF NO PRINT OUT WANTED (SW12=1)
                                1$:     TSTB    (R5)            ;IS NUMBER MINUS? (MSB=1(BIT7))
                                        BPL     2$              ;BR IF NUMBER IS PLUS
                                        TYPE    ,MCRLF          ;TYPE A CR/LF!
                                2$:     TSTB    @TPCSR          ;TTY READY?
                                        BPL     2$              ;BR IF NO.
                                        MOVB    (R5)+,@TPDBR    ;PRINT CURRENT CHAR.
                                        BNE     4$              ;IF NOT ZERO KEEP PRINTING!
                                3$:     MOV     (SP)+,R5        ;END OF OUTPUT. RESTORE R5
```

```
                                        RTI                         ;GO HOME
                                        ;---------------------------
                                .INSTR: MOV     R3,-(SP)            ;SAVE R3 ON STACK
                                        MOV     R4,-(SP)            ;SAVE R4 ON STACK
004046  000002                          MOV     @4(SP),.MSG
004050  010346                          ADD     #2,4(SP)
004052  010446                  .INST1: TYPE
004054  017637  000004  004072  .MSG:   0
004062  062766  000002  000004          MOV     #INBUF,R4
004070  104402                          MOV     #7,R3
004072  000000                  1$:     TSTB    @TKCSR
004074  012704  007504                  BPL     1$
004100  012703  000007                  MOVB    @TKDBR,(R4)
004104  105777  175074                  BICB    #200,(R4)
004110  100375                          CMPB    (R4)+,#15
004112  117714  175070                  BEQ     INSTR2
004116  142714  000200          2$:     TSTB    @TPCSR
004122  122427  000015                  BPL     2$
004126  001417                          MOV     @TKDBR,@TPDBR
004130  105777  175054                  DEC     R3
004134  100375                          BNE     1$
004136  017777  175044  175046          MOV     (SP)+,R4
004144  005303                          MOV     (SP)+,R3
004146  001356                  .INSTE: TYPE    ,MQM
004150  012604                          MOV     R3,-(SP)
004152  012603                          MOV     R4,-(SP)
004154  104402  005666                  BR      .INST1
004160  010346                  INSTR2: MOV     (SP)+,R4            ;RESTORE R4
004162  010446                          MOV     (SP)+,R3            ;RESTORE R3
004164  000741                          RTI
004166  012604
004170  012603
004172  000002

                                        ;CONVERT ASCII STRING TO OCTAL
                                        ;-----------------------------

004174  010546                  .PARAM: MOV     R5,-(SP)
004176  010446                          MOV     R4,-(SP)
004200  016605  000004                  MOV     4(SP),R5
004204  012537  004364                  MOV     (R5)+,LOLIM
004210  012537  004366                  MOV     (R5)+,HILIM
004214  012537  004370                  MOV     (R5)+,DEVADR
004220  112537  004372                  MOVB    (R5)+,LOBITS
004224  112537  004373                  MOVB    (R5)+,ADRCNT
004230  010566  000004                  MOV     R5,4(SP)
004234  005005                  PARAM1: CLR     R5
004236  012704  007504                  MOV     #INBUF,R4
004242  122714  000015                  CMPB    #15,(R4)
004246  001420                          BEQ     PARERR
004250  121427  000060          1$:     CMPB    (R4),#60
004254  002415                          BLT     PARERR
004256  121427  000067                  CMPB    (R4),#67
004262  003012                          BGT     PARERR
004264  142714  000060                  BICB    #60,(R4)
004270  152405                          BISB    (R4)+,R5
004272  122714  000015                  CMPB    #15,(R4)
004276  001406                          BEQ     LIMITS
```

```
                        ASL     R5
                        ASL     R5
                        ASL     R5
                        BR      1$
PARERR:                 INSTER
                        BR      PARAM1

                        ;TEST TO SEE IF NUMBER IS WITHIN LIMITS
                        ;--------------------------------------

LIMITS:                 CMP     R5,HILIM
                        BHI     PARERR
                        CMP     R5,LOLIM
                        BLO     PARERR
                        BITB    LOBITS,R5
                        BNE     PARERR

                        ;STORE NUMBER AT SPECIFIED ADDRESS

                        MOV     DEVADR,R4
1$:                     MOV     R5,(R4)+
                        ADD     #2,R5
                        DECB    ADRCNT
                        BNE     1$
                        MOV     (SP)+,R4
                        MOV     (SP)+,R5
                        RTI
LOLIM:                  0
HILIM:                  0
DEVADR:                 0
LOBITS:                 0
ADRCNT=LOBITS+1

                        ;SAVE PC OF TEST THAT FAILED AND R0-R5
                        ;-------------------------------------

.SAV05:                 MOV     4(SP),SAVPC     ;SAVE R7 (PC)

                        ;SAVE R0-R5

SV05:                   MOV     R5,SAVR5        ;SAVE R5
                        MOV     R4,SAVR4        ;SAVE R4
                        MOV     R3,SAVR3        ;SAVE R3
                        MOV     R2,SAVR2        ;SAVE R2
                        MOV     R1,SAVR1        ;SAVE R1
                        MOV     R0,SAVR0        ;SAVE R0
                        RTI                     ;LEAVE.

                        ;RESTORE R0-R5

.RES05:                 MOV     SAVR0,R0        ;RESTORE R0
                        MOV     SAVR1,R1        ;RESTORE R1
                        MOV     SAVR2,R2        ;RESTORE R2
                        MOV     SAVR3,R3        ;RESTORE R3
                        MOV     SAVR4,R4        ;RESTORE R4
                        MOV     SAVR5,R5        ;RESTORE R5
```

```
                                        RTI                     ;LEAVE
                                        ;CONVERT OCTAL NUMBER TO ASCII AND OUTPUT TO TELEPRINTER
                                        ;-----------------------------------------------------
004466  104402  005672        .CONVR: TYPE    ,MCRLF
004472  010046                .CNVRT: MOV     R0,-(SP)
004474  010146                        MOV     R1,-(SP)
004476  010346                        MOV     R3,-(SP)
004500  010446                        MOV     R4,-(SP)
004502  010546                        MOV     R5,-(SP)
004504  017601  000012                MOV     @12(SP),R1
004510  062766  000002  000012        ADD     #2,12(SP)
004516  012137  004710                MOV     (R1)+,WRDCNT
004522  112137  004712          1$:   MOVB    (R1)+,CHRCNT
004526  112137  004713                MOVB    (R1)+,SPACNT
004532  013137  004714                MOV     @(R1)+,BINWRD
004536  122737  000003  004712        CMPB    #3,CHRCNT
004544  001003                        BNE     2$
004546  042737  177400  004714        BIC     #177400,BINWRD
004554  013704  004714          2$:   MOV     BINWRD,R4
004560  113705  004712                MOVB    CHRCNT,R5
004564  012700  001416                MOV     #TEMP,R0
004570  010403                  3$:   MOV     R4,R3
004572  042703  177770                BIC     #177770,R3
004576  062703  000060                ADD     #060,R3
004602  110320                        MOVB    R3,(R0)+
004604  000241                        CLC
004606  006004                        ROR     R4
004610  000241                        CLC
004612  006004                        ROR     R4
004614  000241                        CLC
004616  006004                        ROR     R4
004620  005305                        DEC     R5
004622  001362                        BNE     3$
004624  012703  007546                MOV     #MDATA,R3
004630  114023                  4$:   MOVB    -(R0),(R3)+
004632  105337  004712                DECB    CHRCNT
004636  001374                        BNE     4$
004640  105737  004713                TSTB    SPACNT
004644  001405                        BEQ     6$
004646  112723  000040          5$:   MOVB    #040,(R3)+
004652  105337  004713                DECB    SPACNT
004656  001373                        BNE     5$
004660  105013                  6$:   CLRB    (R3)
004662  104402  007546                TYPE    ,MDATA
004666  005337  004710                DEC     WRDCNT
004672  001313                        BNE     1$
004674  012605                        MOV     (SP)+,R5
004676  012604                        MOV     (SP)+,R4
004700  012603                        MOV     (SP)+,R3
004702  012601                        MOV     (SP)+,R1
004704  012600                        MOV     (SP)+,R0
004706  000002                        RTI
004710  000000          WRDCNT: 0
004712  000000          CHRCNT: 0
```

```
1005                004713                  SPACNT=CHRCNT+1
1006   004714       000000                  BINWRD: 0
1007
1008
1009                                                ;TRAP DISPATCH SERVICE
1010                                                ;ARGUMENT OF TRAP IS EXTRACTED
1011                                                ;AND USED AS OFFSET TO OBTAIN POINTER
1012                                                ;TO SELECTED SUBROUTINE
1013
1014   004716       011646                  .TRPSR: MOV     (SP),-(SP)      ;GET PC OF RETURN
1015   004720       162716       000002             SUB     #2,(SP)         ;=PC OF TRAP
1016   004724       017616       000000             MOV     @(SP),(SP)      ;GET TRP
1017   004730       006316                  TRPOK:  ASL     (SP)            ;MULTIPLY TRAP ARG BY 2
1018   004732       042716       177001             BIC     #177001,(SP)    ;CLEAR UNWANTED BITS
1019   004736       062716       001330             ADD     #.TRPTAB,(SP)   ;POINTER TO SUBROUTINE ADDRESS
1020   004742       017616       000000             MOV     @(SP),(SP)      ;SUBROUTINE ADDRESS
1021   004746       000136                          JMP     @(SP)+          ;GO TO SUBROUTINE
1022
1023                                                ;ERROR HANDLER
1024                                                ;-------------
1025
1026   004750       004737       007610  .HLT:   JSR     PC,CKSWR        ;CHECK FOR SOFT SWR
1027   004754       032777       010000  174220     BIT     #SW12,@SWR      ;BELL ON ERROR?
1028   004762       001406                          BEQ     XBX             ;BR IF NO BELL
1029   004764       105777       174220             TSTB    @TPCSR          ;TTY READY.
1030   004770       100003                          BPL     XBX             ;DON'T WAIT IF TTY NOT READY.
1031   004772       112777       000207  174212     MOVB    #207,@TPDBR     ;PUSH A BELL AT THE TTY.
1032   005000       032777       020000  174174  XBX:    BIT     #SW13,@SWR      ;DELETE ERROR PRINT OUT?
1033   005006       001105                          BNE     HALTS           ;BR IF NO PRINT OUT WANTED.
1034   005010       021637       001234             CMP     (SP),LSTERR     ;WAS THIS ERROR FOUND LAST TIME?
1035   005014       001404                          BEQ     1$              ;BR IF YES
1036   005016       011637       001234             MOV     (SP),LSTERR     ;RECORD BEING HERE
1037   005022       105037       001325             CLRB    ERRFLG          ;PREPARE HEADER
1038   005026       104406                  1$:     SAVOS                   ;SAVE ALL PROC REGISTERS
1039   005030       011605                          MOV     (SP),R5         ;GET THE PC OF ERROR
1040   005032       162705       000002             SUB     #2,R5           ;GET ADDRESS OF TRAP CALL
1041   005036       011504                          MOV     (R5),R4         ;GET HLT INSTRUCTION
1042   005040       006304                          ASL     R4              ;MULT BY TWO
1043   005042       061504                          ADD     (R5),R4         ;DOUBLE IT
1044   005044       006304                          ASL     R4              ;MULT AGAIN
1045   005046       042704       177001             BIC     #177001,R4      ;CLEAR JUNK
1046   005052       062704       027604             ADD     #.ERRTAB,R4     ;GET POINTER
1047   005056       012437       005172             MOV     (R4)+,ERRMSG    ;GET ERROR MESSAGE
1048   005062       012437       005204             MOV     (R4)+,DATAHD    ;GET DATA HEADRER
1049   005066       011437       005216             MOV     (R4),DATABP     ;GET DATA TABLE
1050   005072       105737       001325             TSTB    ERRFLG          ;TYPE HEADREER
1051   005076       001403                          BEQ     TYPMSG          ;BR IF YES
1052   005100       005737       005216             TST     DATABP          ;DOES DATA TABLE EXIST?
1053   005104       001040                          BNE     TYPDAT          ;BR IF YES.
1054   005106       104402       005672  TYPMSG: TYPE    ,MCRLF
1055   005112       104402       005672             TYPE    ,MCRLF
1056   005116       005737       001220             TST     LOCK
1057   005122       001402                          BEQ     1$
1058   005124       104402       006143             TYPE    ,MASTEK
1059   005130       104402       006131  1$:     TYPE    ,MTSTN
1060   005134       104411       005330             CNVRT   ,XTSTN          ;SHOW IT
```

```
                                        TYPE    ,MERRPC         ;TYPE PC.
                                        CNVRT   ,ERTABO         ;SHOW IT
                                        TYPE    ,MCRLF          ;GIVE A CR/LF
1064  005154  112737  177777  001325   MOVB    #-1,ERRFLG      ;NO MORE HEADER UNLESS NO DATA TABLE.
1065  005162  005737  005172           TST     ERRMSG          ;IS THERE AN ERROR MESSAGE?
1066  005166  001402                   BEQ     WRKO.FM         ;BR IF NO.
1067  005170  104402                   TYPE                    ;TYPE
1068  005172  000000           ERRMSG: 0                       ;    ERROR MESSAGE
1069  005174                   WRKO.FM:                        ;
1070  005174  005737  005204           TST     DATAHD          ;DATA HEADER?
1071  005200  001402                   BEQ     TYPDAT          ;BR IF NO
1072  005202  104402                   TYPE                    ;TYPE
1073  005204  000000           DATAHD: 0                       ;    DATA HEADER
1074  005206  005737  005216   TYPDAT: TST     DATABP          ;DATA TABLE?
1075  005212  001402                   BEQ     RESREG          ;BR IF NO.
1076  005214  104410                   CONVRT                  ;SHOW
1077  005216  000000           DATABP: 0                       ;    DATA TABLE
1078  005220  104407           RESREG: RESO5                   ;RESTORE PROC REGISTERS
1079  005222  022737  003522  000042 HALTS:  CMP     #$ENDAD,@#42    ;IF ACT-11 AUTOMATIC MODE, HALT!!
1080  005230  001403                   BEQ     1$
1081  005232  005777  173744           TST     @SWR            ;HALT ON ERROR?
1082  005236  100005                   BPL     EXITER          ;BR IF NO HALT ON ERROR
1083  005240  010046           1$:     PUSHRO                  ;SAVE RO
1084  005242  016600  000002           MOV     2(SP),RO        ;SHOW ERROR PC IN DATA LIGHTS
1085  005246  000000                   HALT                    ;HALT
1086  005250  012600                   POPRO                   ;GET RO
1087  005252  005237  001232   EXITER: INC     ERRCNT          ;UPDATE ERROR COUNT
1088  005256  032777  000400  173716   BIT     #SW08,@SWR      ;GOTO TOP OF TEST?
1089  005264  001007                   BNE     1$              ;BR IF YES
1090  005266  032777  002000  173706   BIT     #SW10,@SWR      ;GOTO NEXT TEST?
1091  005274  001411                   BEQ     2$              ;BR IF NO
1092  005276  013737  001216  001214   MOV     NEXT,RETURN     ;SET FOR NEXT TEST
1093  005304  012706  001200           MOV     #STACK,SP       ;RESET SP
1094  005310  013701  001404           MOV     DMCSR,R1        ;SET UP R1
1095  005314  000177  173674           JMP     @RETURN         ;GOTO SPECIFIED TEST
1096  005320  000002           2$:     RTI                     ;RETURN
1097  005322  000001           ERTABO: 1
1098  005324     006     002           .BYTE   6,2
1099  005326  001276                   SAVPC
1100  005330  000001           XTSTN:  1
1101  005332     003     002           .BYTE   3,2
1102  005334  001226                   TSTNO
1103                                   ;ENTER HERE ON POWER FAILURE
1104                                   ;--------------------------
1105
1106
1107  005336                   .PFAIL:
1108  005336  012737  005350  000024           MOV     #RESTART,24             ;SET UP FOR POWER UP TRAP
1109  005344  000000                   HALT                                    ;HALT ON POWER DOWN NORMAL
1110  005346  000777                   BR      .
1111
1112                                   ;PROCESSOR WILL TRAP HERE WHEN POWER IS RESTORED
1113
1114  005350                   RESTAR:
1115  005350  012737  005336  000024           MOV     #.PFAIL,24      ;SET UP FOR POWER FAILURE
1116  005356  012706  001200           MOV     #STACK,SP       ;RESET THE STACK POINTER
```

```
1117  005362  013701  001404                  MOV     DMCSR,R1        ;RESTORE R1
1118  005366  005037  001416                  CLR     TEMP            ;READY FOR TIMMER
1119  005372  005237  001416                  INC     TEMP            ;PLUS ONE TO THE TIMER!
1120  005376  001375                          BNE     .-4             ;BR IF MORE TO GO
1121  005400  104402  005675                  TYPE    ,MPFAIL         ;TYPE THE MESSAGE
1122  005404  104411  005430                  CNVRT   ,PFTAB          ;TELL WHAT TEST TO RETURN TO.
1123  005410  105037  001325                  CLRB    ERRFLG          ;START CLEAN
1124  005414  005037  001234                  CLR     LSTERR          ;
1125  005420  005011                          CLR     (R1)            ;CLEAR MAINT BITS
1126  005422  104412                          MSTCLR                  ;START CLEAN UP OF DEVICE
1127  005424  000177  173564                  JMP     @RETURN         ;START DOING THAT TEST AGAIN.
1128  005430  000001                  PFTAB:  1
1129  005432     003     002           .BYTE   3,2
1130  005434  001226                          TSTNO
1131
1132  005436                          .DELAY:
1133  005436  012777  000020  173746          MOV     #20,@DMP04
1134  005444  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1135  005446  121111                          121111                  ;POKE CLOCK DELAY BIT
1136  005450                          1$:
1137  005450  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1138  005452  121224                          121224                  ;PORT4←IBUS#11
1139  005454  032777  000020  173730          BIT     #BIT4,@DMP04    ;IS CLOCK BIT SET?
1140  005462  001772                          BEQ     1$              ;BR IF NO
1141  005464  000002                          RTI
1142
1143  005466                          .MSTCLR:
1144  005466  152777  000100  173712          BISB    #BIT6,@DMCSRH   ;SET MASTER CLEAR
1145  005474  142777  000300  173704          BICB    #BIT6!BIT7,@DMCSRH  ;CLEAR MASTER CLEAR AND RUN
1146  005502  000002                          RTI                     ;RETURN
1147
1148  005504                          .ROMCLK:
1149  005504  152777  000002  173674          BISB    #BIT1,@DMCSRH   ;SET ROMI
1150  005512  013677  173676                  MOV     @(SP)+,@DMP06   ;LOAD INSTRUCTION IN SEL6
1151  005516  062746  000002                  ADD     #2,-(SP)        ;ADJUST STACK
1152  005522  032777  000100  173452          BIT     #SW06,@SWR      ;HALT IF SW06 =1
1153  005530  001401                          BEQ     1$              ;BR IF SW06 =0
1154  005532  000000                          HALT                    ;HALT BEFORE CLOCKING INSTRUCTION
1155  005534  152777  000003  173644  1$:     BISB    #BIT1!BIT0,@DMCSRH ;CLOCK INSTRUCTION
1156  005542  142777  000007  173636          BICB    #BIT2!BIT1!BIT0,@DMCSRH  ;CLEAR ROMO, ROMI, STEP
1157  005550  000002                          RTI
1158
1159  005552                          .DATACLK:
1160  005552  013637  001416                  MOV     @(SP)+,TEMP     ;PUT TICK COUNT IN TEMP
1161  005556  062746  000002                  ADD     #2,-(SP)        ;ADJUST STACK
1162  005562  152777  000020  173616  1$:     BISB    #BIT4,@DMCSRH   ;SET STEP LU
1163  005570  027777  173610  173606          CMP     @DMCSR,@DMCSR   ;WASTE TIME
1164  005576  142777  000020  173602          BICB    #BIT4,@DMCSRH   ;CLEAR STEP LU
1165  005604  005337  001416                  DEC     TEMP            ;DEC TICK COUNT
1166  005610  001364                          BNE     1$              ;BR IF NOT DONE
1167  005612  000002                          RTI                     ;RETURN
1168  005614  000001                  3$:     .BLKW 1
1169
1170  005616                          .TIMER:
1171  005616  013637  001416                  MOV     @(SP)+,TEMP     ;MOVE COUNT TO TEMP
1172  005622  062746  000002                  ADD     #2,-(SP)        ;ADJUST STACK
```

```
 1173  005626                          1$:
 1174  005626  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 1175  005630  021364                          021364                  ;PORT4←IBUS* REG11
 1176  005632  032777  000002  173552          BIT     #2,@DMP04       ;IS PGM CLOCK BIT CLEAR?
 1177  005640  001772                          BEQ     1$              ;BR IF YES
 1178  005642                          2$:
 1179  005642  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 1180  005644  021364                          021364                  ;PORT4←IBUS* REG11
 1181  005646  032777  000002  173536          BIT     #2,@DMP04       ;IS PGM CLOCK BIT SET?
 1182  005654  001372                          BNE     2$              ;BR IF YES
 1183  005656  005337  001416                  DEC     TEMP            ;DEC COUNT
 1184  005662  001361                          BNE     1$              ;BR IF NOT DONE
 1185  005664  000002                          RTI                     ;RETURN
 1186
 1187  005666  020040  000077          MQM:    .ASCIZ  /  ?/
   (2) 005672  005015     000          MCRLF:  .ASCIZ   <15><12>
   (2) 005675     377  053520  020122  MPFAIL: .ASCIZ  <377>/PWR FAILED, RESTART AT TEST /
   (2) 005733     377  047105  020104  MEPASS: .ASCIZ  <377>/END PASS CZDMGD  /
   (2) 005756  051377     000          MR:     .ASCIZ  <377>/R/
   (2) 005761     377  047516  042040  MERR2:  .ASCIZ  <377>/NO DEVICES PRESENT./
   (2) 006006  044777  051516  043125  MERR3:  .ASCIZ  <377>/INSUFFICIENT DATA!/
   (2) 006032  052377  051505  020124  MTSTPC: .ASCIZ  <377>/TEST PC-/
   (2) 006044  046377  041517  020113  MLOCK:  .ASCIZ  <377>/LOCK ON SELECTED TEST/
   (2) 006073     103  051123  020072  MCSRX:  .ASCIZ   /CSR: /
   (2) 006101     126  041505  020072  MVECX:  .ASCIZ  /VEC: /
   (2) 006107     120  051501  042523  MPASSX: .ASCIZ  /PASSES: /
   (2) 006120  051105  047522  051522  MERRX:  .ASCIZ  /ERRORS: /
   (2) 006131     124  051505  020124  MTSTN:  .ASCIZ  /TEST NO: /
   (2) 006143     052     000          MASTEK: .ASCIZ  /*/
   (2) 006145     377  042523  020124  MNEW:   .ASCIZ  <377>/SET SWITCH REG TO DMC11'S DESIRED ACTIVE./
   (2) 006220  041520  020072     000  MERRPC: .ASCIZ  /PC: /
   (2) 006225     212  020040  020040  XHEAD:  .ASCII  <212>/           MAP OF DMC11 STATUS/
   (2) 006264  020377  020040  020040          .ASCII  <377>/           --------------------/
   (2) 006323     212  020040  041520          .ASCII  <212>/  PC      CSR     STAT1    STAT2    STAT3/
   (2) 006375     377  026455  026455          .ASCIZ  <377>/------  ------  ------   ------   ------/
   (2) 006451     377  047510  020127  NUM:    .ASCIZ  <377>/HOW MANY DMC11'S TO BE TESTED?/
   (2) 006511     377  051503  020122  CSR:    .ASCIZ  <377>/CSR ADDRESS?/
   (2) 006527     377  042526  052103  VEC:    .ASCIZ  <377>/VECTOR ADDRESS?/
   (2) 006550  041377  020122  051120  PRIO:   .ASCIZ  <377>/BR PRIORITY LEVEL? (4,5,6,7)?/
   (2) 006607     377  043111  042040  CRAM:   .ASCIZ  <377>/IF DMC HAS CRAM (M8204) TYPE "Y", IF CROM (M8200) TYPE "N" ?/
   (2) 006705     377  044127  041511  MODU:   .ASCIZ  <377>/WHICH LINE UNIT? IF NONE TYPE "N", IF M8201 TYPE "1", IF M8202 TYP
   (2) 007017     377  053523  052111  LINE:   .ASCIZ  <377>/SWITCH PAC#1 (DDCMP LINE #)?/
   (2) 007055     377  053523  052111  BM:     .ASCIZ  <377>/SWITCH PAC#2 (BM873 BOOT ADD)?/
   (2) 007115     377  051511  052040  CONN:   .ASCIZ  <377>/IS THE LOOP BACK CONNECTOR ON?/
   (2) 007155     377  047516  042040  NOACT:  .ASCIZ  <377>/NO JEVICES ARE SELECTED/
   (2) 007206  005377  053523  036522  SWMES:  .ASCIZ  <377><12>/SWR= /
   (2) 007216  042516  037527  000040  SWMES1: .ASCIZ  /NEW? /
   (2) 007224  177777  046504  030503  CONERR: .ASCIZ  <377><377>/DMC11 FOUND AT NON-STANDARD ADDRESS  PC: /
   (2) 007300  042777  050130  041505  CNERR:  .ASCIZ  <377>/EXPECTED  FOUND/
   (2) 007321     040  042050  041515  DMCM:   .ASCIZ  / (DMC) /
   (2) 007331     040  045450  041515  KMCM:   .ASCIZ  / (KMC) /
   (2) 007341     377  046504  030503  SPEED:  .ASCIZ  <377>/DMC11-AR(REMOTE,LOW SPEED) OR DMC11-AL(LOCAL,HIGH SPEED) TYPE "R"
   (2)         007456                  .EVEN
   (2) 007456  000005                  XSTATQ: 5
 1188  007460     006      003                 .BYTE   6,3
 1189  007462  001246                          TEMP1
```

```
1190  007464  000006  000003            .BYTE   6,3
1191  007466  001250                    TEMP2
1192  007470  000006  000003            .BYTE   6,3
1193  007472  001252                    TEMP3
1194  007474  000006  000003            .BYTE   6,3
1195  007476  001254                    TEMP4
1196  007500  000006  000002            .BYTE   6,2
1197  007502  001256                    TEMP5
1198                            .EVEN
1199
1200                            ;BUFFERS FOR INPUT-OUTPUT
1201
1202  007504  000000            INBUF:  0
1203          007546            .=.+40
1204  007546  000000            MDATA:  0
1205          007610            .=.+40
1206
1207
1208                            ;ROUTINE USED TO CHANGE SOFTWARE SWITCH
1209                            ;REGISTER USING THE CONSOLE TERMINAL
1210                            ;-------------------------------------
1211
1212  007610  022737  000176  001202  CKSWR:  CMP     #SWREG,SWR      ;IS THE SOFT SWR BEING USED?
1213  007616  001077                    BNE     CKSWR5          ;BR IF NO
1214  007620  105777  171360            TSTB    @TKCSR          ;IS DONE SET?
1215  007624  100003                    BPL     2$              ;GO ON IF NOT SET
1216  007626  012737  177777  003734    MOV     #-1,DONE        ;IF DONE SET, SET FLAG
1217  007634  022777  000007  171344  2$:     CMP     #7,@TKDBR       ;WAS CTRL G TYPED? (7 BIT ASCII)
1218  007642  001404                    BEQ     1$              ;BR IF YES
1219  007644  022777  000207  171334    CMP     #207,@TKDBR     ;WAS CTRL G TYPED? (8 BIT ASCII)
1220  007652  001061                    BNE     CKSWR5          ;BR IF NO
1221  007654  010246            1$:     MOV     R2,-(SP)        ;STORE R2
1222  007656  010346                    MOV     R3,-(SP)        ;STORE R3
1223  007660  010446                    MOV     R4,-(SP)        ;STORE R4
1224  007662  012737  177777  010020    MOV     #-1,SWFLG       ;SET SOFT TYPE OUT FLAG
1225  007670  005002            CKSWR1: CLR     R2              ;CLEAR NEW SWR CONTENTS
1226  007672  012704  177777            MOV     #-1,R4          ;SET FLAG TO ALL ONES
1227  007676  104402  007206            TYPE    ,SWMES          ;TYPE "SWR= "
1228  007702  104411            CKSWR2: CNVRT                   ;TYPE OUT PRESENT CONTENTS
1229  007704  010054                    SOFTSW                  ;OF SOFT SWITCH REGISTER
1230  007706  104402  007216    CKSWR3: TYPE    ,SWMES1         ;TYPE "NEW? "
1231  007712  004737  010022    CKSWR4: JSR     PC,INCHAR       ;GET RESPONSE
1232  007716  022703  000015            CMP     #15,R3          ;WAS IT A CR?
1233  007722  001424                    BEQ     5$              ;BR IF YES
1234  007724  022703  000012            CMP     #12,R3          ;WAS IT A LF?
1235  007730  001416                    BEQ     4$              ;BR IF YES
1236  007732  022703  000025            CMP     #25,R3          ;WAS IT CTRL U?
1237  007736  001754                    BEQ     CKSWR1          ;BR IF YES(START OVER)
1238  007740  022703  000007            CMP     #7,R3           ;IF CNTL G GET NEXT CHAR
1239  007744  001762                    BEQ     CKSWR4
1240  007746  005004                    CLR     R4              ;IT MUST BE A DIGIT SO CLR FLAG
1241  007750  042703  177770            BIC     #177770,R3      ;ONLY 0-7 ARE LEGAL SO MASK OFF BITS
1242  007754  006302                    ASL     R2              ;SHIFT R2 3 TIMES
1243  007756  006302                    ASL     R2
1244  007760  006302                    ASL     R2
1245  007762  050302                    BIS     R3,R2           ;ADD LAST DIGIT
```

```
1246  007764  000752                        BR      CKSWR4          ;GET NEXT CHARACTER
1247  007766  012766  002002  000006  4$:   MOV     #.START,6(SP)   ;IF WAS TYPED SO GO TO START
1248  007774  005704                  5$:   TST     R4              ;IS FLAG CLEAR?
1249  007776  001002                        BNE     6$              ;IF NOT DON'T CHANGE SOFT SWR
1250  010000  010277  171176                MOV     R2,@SWR         ;IF YES THEN WRITE NEW CONTENTS TO SOFT SWR
1251  010004  005037  010020          6$:   CLR     SWFLG           ;CLEAR TYPEOUT FLAG
1252  010010  012604                        MOV     (SP)+,R4        ;RESTORE R4
1253  010012  012603                        MOV     (SP)+,R3        ;RESTORE R3
1254  010014  012602                        MOV     (SP)+,R2        ;RESTORE R2
1255  010016  000207                  CKSWR5: RTS     PC              ;RETURN
1256
1257  010020  000000                  SWFLG:  0
1258
1259  010022  105777  171156          INCHAR: TSTB    @TKCSR
1260  010026  100375                        BPL     .-4
1261  010030  017703  171152                MOV     @TKDBR,R3
1262  010034  105777  171150                TSTB    @TPCSR
1263  010040  100375                        BPL     .-4
1264  010042  010377  171144                MOV     R3,@TPDBR
1265  010046  042703  000200                BIC     #BIT7,R3
1266  010052  000207                        RTS     PC
1267
1268  010054  000001                  SOFTSW: 1
1269  010056     006      002                .BYTE   6,2
1270  010060  000176                        SWREG
```

```
                                                ;
                                                ;ROUTINE USED TO "CYCLE" THROUGH UP TO 16 DMC11'S
                                                ;THIS ROUTINE SETS UP THE CONTROL ADDRESS FOR THE DIAGNOSTIC
                                                ;AND RUNS THE SPECIFIED DMC11'S.   THIS ROUTINE *MUST*
                                                ;BE RUN FIRST BEFORE ENTERING THE DIAGNOSTIC FOR THE
                                                ;SETUP NECESSARY.
                                                ;
010062  005737  001306          CYCLE:  TST     DMACTV          ;ARE ANY DMC11'S TO BE TESTED?
010066  001004                          BNE     1$              ;BR IF OK.
010070  104402  007155                  TYPE    ,NOACT          ;NO DMC11'S SELECTED!!
010074  000000                          HALT                    ;STOP THE SHOW.
010076  000776                          BR      .-2             ;DISQUALIFY CONT. SW.
010100  000241                  1$:     CLC                     ;CLEAR PROC. CARRY BIT.
010102  006137  001316                  ROL     RUN             ;UPDATE POINTER
010106  005537  001316                  ADC     RUN             ;CATCH CARRY FROM RUN
010112  062737  000004  001322          ADD     #4,MILK         ;UPDATE POINTER
010120  062737  000010  001320          ADD     #10,CREAM       ;UPDATE ADDRESS POINTER.
010126  022737  001700  001320          CMP     #DM.MAP+200,CREAM
010134  001006                          BNE     2$              ;KEEP GOING; NOT ALL TESTED FOR.
010136  012737  001500  001320          MOV     #DM.MAP,CREAM   ;RESET ADDRESS POINTER.
010144  012737  001702  001322          MOV     #CNT.MAP,MILK   ;RESET PASS COUNT POINTER
010152  033737  001316  001306  2$:     BIT     RUN,DMACTV      ;IS THIS ONE ACTIVE?
010160  001747                          BEQ     1$              ;BR IF NO
010162  013700  001320                  MOV     CREAM,R0        ;GET ADDRESS POINTER
010166  013702  001322                  MOV     MILK,R2         ;GET PASS COUNT POINTER
010172  012037  001404                  MOV     (R0)+,DMCSR     ;LOAD SYSTEM CTRL. REG
010176  011037  001374                  MOV     (R0),DMRVEC     ;LOAD VECTOR
010202  042737  177000  001374          BIC     #177000,DMRVEC  ;CLEAR UNWANTED BITS
010210  012037  001366                  MOV     (R0)+,STAT1     ;LOAD STAT1
010214  012037  001370                  MOV     (R0)+,STAT2     ;LOAD STAT2
010220  012037  001372                  MOV     (R0)+,STAT3     ;LOAD STAT3
010224  012237  001230                  MOV     (R2)+,PASCNT    ;LOAD PASS COUNT
010230  012237  001232                  MOV     (R2)+,ERRCNT    ;LOAD ERROR COUNT
010234  012700  000002                  MOV     #2,R0           ;SAVE CORE THIS WAY!
010240  013737  001404  001406          MOV     DMCSR,DMCSRH
010246  005237  001406                  INC     DMCSRH
010252  013737  001406  001410          MOV     DMCSRH,DMCTL
010260  005237  001410                  INC     DMCTL
010264  013737  001410  001412          MOV     DMCTL,DMP04
010272  060037  001412                  ADD     R0,DMP04
010276  013737  001412  001414          MOV     DMP04,DMP06
010304  060037  001414                  ADD     R0,DMP06

010310  013737  001374  001376          MOV     DMRVEC,DMRLVL   ;PTY LVL
010316  060037  001376                  ADD     R0,DMRLVL       ;
010322  013737  001376  001400          MOV     DMRLVL,DMTVEC   ;TX VEC
010330  060037  001400                  ADD     R0,DMTVEC       ;
010334  013737  001400  001402          MOV     DMTVEC,DMTLVL   ;TX LVL
010342  060037  001402                  ADD     R0,DMTLVL

010346  032737  000002  001236          BIT     #SW01,STRTSW    ;IS TEST NO. SELECTED
010354  001450                          BEQ     7$              ;BR IF NO
010356                          4$:
010356  005737  000042                  TST     @#42            ;RUNNING IN AUTO MODE?
```

```
1327  010362  001045                    BNE     7$                  ;BR IF YES
1328  010364  104402  005672            TYPE    ,MCRLF
1329  010370  104403                    INSTR                       ;GET TEST NO.
1330  010372  006131                    MTSTN
1331  010374  104405                    PARAM
1332  010376  000001                    1
1333  010400  001000                    1000
1334  010402  001226                    TSTNO
1335  010404     000          .BYTE     0
1336  010405     001          .BYTE     1
1337  010406  012700  022324            MOV     #TST1,R0
1338  010412  022710           5$:      CMP     (PC)+,(R0)          ;CMP FIRST WORD TO 12737
1339  010414  012737                    MOV     (PC)+,@(PC)+
1340  010416  001020                    BNE     6$                  ;BR IF NOT SAME
1341  010420  023760  001226  000002    CMP     TSTNO,2(R0)         ;DOES TSTNO MATCH?
1342  010426  001014                    BNE     6$                  ;BR IF NO
1343  010430  022760  001226  000004    CMP     #TSTNO,4(R0)        ;IS LAST WORD OK?
1344  010436  001010                    BNE     6$                  ;BR IF NO
1345  010440  010037  001214            MOV     R0,RETURN           ;IT IS A LEGAL TEST SO DO IT
1346  010444  104402  005756            TYPE    ,MR
1347  010450  042737  000002  001236    BIC     #SW01,STRTSW
1348  010456  000412                    BR      8$
1349  010460  005720           6$:      TST     (R0)+               ;POP R0
1350  010462  020027  026474            CMP     R0,#TLAST+10        ;AT END YET?
1351  010466  001351                    BNE     5$                  ;BR IF NO
1352  010470  104402  005666            TYPE    ,MQM                ;YES ILLEGAL TEST NO.
1353  010474  000730                    BR      4$                  ;TRY AGAIN
1354
1355  010476  012737  022324  001214    7$:     MOV     #TST1,RETURN    ;PREPARE RETURN ADDRESS
1356  010504  013701  001404            8$:     MOV     DMCSR,R1        ;R1 = BASE DMC11 ADDRESS
1357  010510  000177  170500            JMP     @RETURN             ;GO START TESTING.
1358
1359
1360                                    ;ROUTINE USED TO "AUTO SIZE" THE DMC11
1361                                    ;CSR AND VECTOR.
1362                                    ;NOTE:   THE CSR MAY BE ANY WHERE IN THE FLOATING
1363                                    ;        ADDRESS RANGE (160000:164000)
1364                                    ;        AND THE VECTOR MAY BE ANY WHERE IN THE
1365                                    ;        FLOATING VECTOR RANGE (300:770)
1366                                    ;
1367
1368  010514                    AUTO.SIZE:
1369  010514  000005                    RESET                       ;INSURE A BUS INIT.
1370  010516  012702  001500    CSRMAP: MOV     #DM.MAP,R2          ;LOAD MAP POINTER.
1371  010522  005022            1$:     CLR     (R2)+               ;ZERO ENTIRE MAP
1372  010524  022702  001700            CMP     #DM.END,R2          ;ALL DONE?
1373  010530  001374                    BNE     1$                  ;BR IF NO
1374  010532  005037  001310            CLR     DMNUM               ;SET OCTAL NUMBER OF DMC11'S TO 0
1375  010536  012702  001500            MOV     #DM.MAP,R2          ;R2 POINTS TO DMC MAP
1376  010542  005037  001306            CLR     DMACTV              ;CLEAR ACTIVE
1377  010546  032737  000001  001236    BIT     #SW00,STRTSW        ;QUESTIONS?
1378  010554  001002                    BNE     .+6                 ;BR IF YES
1379  010556  000137  011254            JMP     7$                  ;IF NO SKIP QUESTIONS
1380  010562  012737  000001  001256    MOV     #1,TEMP5            ;START WITH 1
1381  010570  104403                    INSTR
1382  010572  006451                    NUM
```

```
1383  010574  104405                        PARAM
1384  010576  000001                        1
1385  010600  000020                        16.
1386  010602  001252                        TEMP3
1387  010604     000                        .BYTE   0
1388  010605     001                        .BYTE   1
1389  010606  013737  001252  001310        MOV     TEMP3,DMNUM     ;DMNUM = HOW MANY
1390  010614  104402  005672        12$:    TYPE    ,MCRLF
1391  010620  104410                        CONVRT                  ;TYPE WHICH DMC IS BEING DONE
1392  010622  012004                        WHICH                   ;TEMP5 IS WHICH DMC
1393  010624  005237  001256                INC     TEMP5
1394  010630  104403                        INSTR
1395  010632  006511                        CSR
1396  010634  104405                        PARAM
1397  010636  160000                        160000
1398  010640  164000                        164000
1399  010642  001254                        TEMP4
1400  010644     000                        .BYTE   0
1401  010645     001                        .BYTE   1
1402  010646  013722  001254                MOV     TEMP4,(R2)+     ;STORE CSR IN MAP
1403  010652  104403                        INSTR
1404  010654  006527                        VEC
1405  010656  104405                        PARAM
1406  010660  000000                        0
1407  010662  000776                        776
1408  010664  001254                        TEMP4
1409  010666     000                        .BYTE   0
1410  010667     001                        .BYTE   1
1411  010670  013712  001254                MOV     TEMP4,(R2)      ;STORE VECTOR IN MAP
1412  010674  104402        10$:    TYPE
1413  010676  006550                        PRIO                    ;ASK WHAT BR LEVEL
1414  010700  004737  012270                JSR     PC,INTTY        ;GET RESPONSE
1415  010704  022703  000024                CMP     #24,R3          ;
1416  010710  101014                        BHI     50$             ;BR IF LESS THAN 4
1417  010712  022703  000027                CMP     #27,R3          ;
1418  010716  103411                        BLO     50$             ;BR IF GREATER THAN 7
1419  010720  012704  000011                MOV     #11,R4          ;R4 = NUMBER OF SHIFTS
1420  010724  006303                        ASL     R3              ;SHIFT R3 LEFT
1421  010726  005304                        DEC     R4              ;DEC SHIFT COUNT
1422  010730  001375                        BNE     .-4             ;BR IF NOT DONE
1423  010732  042703  170777                BIC     #170777,R3      ;BIC UNWANTED BITS
1424  010736  050312                        BIS     R3,(R2)         ;PUT BR LEVEL IN STATUS MAP
1425  010740  000403                        BR      8$              ;CONTINUE
1426  010742  104402        50$:    TYPE
1427  010744  005666                        MQM                     ;RESPONSE IS OUT OF LIMITS
1428  010746  000752                        BR      10$             ;TRY AGAIN
1429  010750  104402        8$:     TYPE
1430  010752  006607                        CRAM                    ;DOES DMC HAVE CRAM?
1431  010754  004737  012270                JSR     PC,INTTY        ;GET REPLY
1432  010760  022703  000131                CMP     #131,R3
1433  010764  001427                        BEQ     9$              ;YES
1434  010766  022703  000116                CMP     #116,R3         ;NO
1435  010772  001403                        BEQ     40$             ;NOT A Y OR N
1436  010774  104402                        TYPE
1437  010776  005666                        MQM                     ;TYPE "?"
1438  011000  000763                        BR      8$              ;ASK AGAIN
```

```
1439  011002  104402                  40$:    TYPE
1440  011004  007341                          SPEED                           ;DMC11-AR OR DMC11-AL?
1441  011006  004737  012270                  JSR     PC,INTTY                ;GET RESPONSE
1442  011012  022703  000122                  CMP     #122,R3                 ;IS IT R
1443  011016  001414                          BEQ     16$                     ;BR IF REMOTE
1444  011020  022703  000114                  CMP     #114,R3                 ;IS IT L
1445  011024  001403                          BEQ     41$                     ;BR IF LOCAL
1446  011026  104402                          TYPE
1447  011030  005666                          MQM
1448  011032  000763                          BR      40$                     ;TRY AGAIN
1449  011034  052762  000002  000004  41$:    BIS     #BIT1,4(R2)             ;SET BIT1 IN STAT3
1450  011042  000402                          BR      16$                     ;CONTINUE
1451  011044  052712  100000          9$:     BIS     #BIT15,(R2)             ;SET BIT 15 IF CRAM
1452  011050  104402                  16$:    TYPE
1453  011052  006705                          MODU                            ;ASK WHICH LINE UNIT
1454  011054  004737  012270                  JSR     PC,INTTY                ;GET REPLY
1455  011060  022703  000021                  CMP     #21,R3                  ;"1"
1456  011064  001417                          BEQ     30$
1457  011066  022703  000022                  CMP     #22,R3                  ;"2"
1458  011072  001412                          BEQ     31$
1459  011074  022703  000116                  CMP     #116,R3                 ;"N"
1460  011100  001403                          BEQ     32$
1461  011102  104402                          TYPE
1462  011104  005666                          MQM                             ;IF NOT A 1,2 OR N TYPE "?"
1463  011106  000760                          BR      16$                     ;TRY AGIAN
1464  011110  052722  010000          32$:    BIS     #BIT12,(R2)+            ;SET BIT 12 IN STAT2 IF NO LU
1465  011114  022222                          CMP     (R2)+,(R2)+             ;POP OVER STAT2 AND STAT3
1466  011116  000447                          BR      33$
1467  011120  052712  020000          31$:    BIS     #BIT13,(R2)             ;SET BIT 13 IN STAT2 IF M8202
1468  011124  104402                  30$:    TYPE
1469  011126  007115                          CONN                            ;ASK IF LOOP-BACK IS ON
1470  011130  004737  012270                  JSR     PC,INTTY                ;GET REPLY
1471  011134  022703  000131                  CMP     #131,R3                 ;Y
1472  011140  001406                          BEQ     17$
1473  011142  022703  000116                  CMP     #116,R3                 ;N
1474  011146  001406                          BEQ     18$
1475  011150  104402                          TYPE
1476  011152  005666                          MQM                             ;IF NOT Y OR N TYPE "?"
1477  011154  000763                          BR      30$                     ;TRY AGAIN
1478  011156  052722  040000          17$:    BIS     #BIT14,(R2)+            ;TURNAROUND IS CONNECTED
1479  011162  000402                          BR      19$
1480  011164  042722  040000          18$:    BIC     #BIT14,(R2)+            ;NO TURNAROUND
1481  011170                          19$:
1482  011170  104403                          INSTR
1483  011172  007017                          LINE
1484  011174  104405                          PARAM
1485  011176  000000                          0
1486  011200  000377                          377
1487  011202  001254                          TEMP4
1488  011204     000                          .BYTE   0
1489  011205     001                          .BYTE   1
1490  011206  113722  001254                  MOVB    TEMP4,(R2)+             ;STORE SWITCH PAC IN MAP
1491  011212  104403                          INSTR
1492  011214  007055                          BM
1493  011216  104405                          PARAM
1494  011220  000000                          0
```

```
1495  011222  000377                        377
1496  011224  001254                        TEMP4
1497  011226     000                        .BYTE   0
1498  011227     001                        .BYTE   1
1499  011230  113722  001254                MOVB    TEMP4,(R2)+     ;STORE SWITCH PAC IN MAP
1500  011234  005722                        TST     (R2)+           ;POP OVER STAT3
1501  011236  005337  001252          33$:  DEC     TEMF3           ;DEC DMC COUNT
1502  011242  001402                        BEQ     34$             ;BR IF DONE
1503  011244  000137  010614                JMP     12$             ;JUMP IF NOT
1504  011250  000137  011704          34$:  JMP     13$             ;CONTINUE
1505  011254  012701  160000           7$:  MOV     #160000,R1      ;SET FOR FIRST ADDRESS TO BE TESTED
1506  011260  012737  011776  000004       MOV     #6$,@#4         ;SET FOR NON-EXISTANT DEVICE TIME OUT
1507  011266  005011                   2$:  CLR     (R1)            ;CLEAR SEL0
1508  011270  005711                        TST     (R1)            ;IF DMC11 DMCSR S/B 0
1509  011272  001172                        BNE     3$              ;IF NO DEV ; TRAP TO 4. IF NO BIT 8 THEN NO DMC11
1510  011274  005061  000006               CLR     6(R1)           ;CLEAR SEL6
1511  011300  005761  000006               TST     6(R1)           ;IF DMC11 THEN DMRIC S/B =0!
1512  011304  001165                        BNE     3$              ;BR IF NOT DMC11
1513  011306  012711  002000               MOV     #BIT10,(R1)     ;SET ROMO
1514  011312  005061  000004               CLR     4(R1)           ;CLEAR SEL4
1515  011316  012761  125252  000006       MOV     #125252,6(R1)   ;WRITE THIS TO SEL6
1516  011324  052711  020000               BIS     #BIT13,(R1)     ;WRITE IT!
1517  011330  022761  125252  000004       CMP     #125252,4(R1)   ;WAS IT WRITTEN?
1518  011336  001004                        BNE     21$             ;IF NO IT IS NOT CRAM
1519  011340  052762  100000  000002       BIS     #BIT15,2(R2)    ;SET BIT15 IF CRAM
1520  011346  000431                        BR      22$
1521  011350  012711  001000          21$:  MOV     #BIT9,(R1)      ;SET ROMI
1522  011354  012761  100430  000006       MOV     #100430,6(R1)   ;PUT INSTRUCTION IN SEL6
1523  011362  012711  001400               MOV     #BIT9!BIT8,(R1) ;CLOCK INSTRUCTION (MICRO PROC PC TO 0)
1524  011366  012711  002000               MOV     #BIT10,(R1)     ;SET ROMO
1525  011372  022761  016472  000006       CMP     #016472,6(R1)   ;IS IT LOCAL CROM?
1526  011400  001411                        BEQ     23$             ;BR IF YES
1527  011402  022761  016461  000006       CMP     #016461,6(R1)   ;IS IT REMOTE CROM?
1528  011410  001410                        BEQ     22$             ;BR IF YES
1529  011412  022761  177777  000006       CMP     #-1,6(R1)       ;NO CROM?
1530  011420  001404                        BEQ     22$             ;BR IF YES
1531  011422  000516                        BR      3$              ;NOT A DMC
1532  011424  052762  000002  000006  23$:  BIS     #BIT1,6(R2)     ;SET BIT 1 IN STAT3
1533                                   ;AT THIS POINT IT IS ASSUMED THAT R1 HOLDS A DMC11 CSR ADDRESS.
1534  011432  010122                   22$:  MOV     R1,(R2)+        ;STORE CSR IN CORE TABLE.
1535  011434  012711  001000           15$:  MOV     #BIT9,(R1)      ;CLEAR LINE UNIT LOOP
1536  011440  005061  000004               CLR     4(R1)           ;CLEAR PORT4
1537  011444  012761  122113  000006       MOV     #122113,6(R1)   ;LOAD INSTRUCTION (CLR DTR)
1538  011452  052711  000400               BIS     #BIT8,(R1)      ;CLOCK INSTRUCTION
1539  011456  012761  021264  000006       MOV     #021264,6(R1)   ;LOAD INSTRUCTION
1540  011464  052711  000400               BIS     #BIT8,(R1)      ;CLOCK INSTRUCTION
1541  011470  122761  000377  000004       CMPB    #377,4(R1)      ;IS IT ALL ONES?
1542  011476  001003                        BNE     .+10            ;BR IF NO
1543  011500  052712  010000               BIS     #BIT12,(R2)     ;IF YES, NO LINE UNIT, SET STATUS BIT
1544  011504  000436                        BR      20$
1545  011506  032761  000002  000004       BIT     #BIT1,4(R1)     ;IS SWITCH A ONE?
1546  011514  001403                        BEQ     .+10            ;BR IF M8201
1547  011516  052712  060000               BIS     #BIT13!BIT14,(R2) ;M8202 ASSUME CONNECTOR
1548  011522  000427                        BR      20$             ;CONNECTOR ON)
1549  011524  032761  000010  000004       BIT     #BIT3,4(R1)     ;IS MRDY SET
1550  011532  001023                        BNE     20$             ;BR IF M8201 NO CONNECTOR (ON LINE)
```

```
1551  011534  012761  000100  000004          MOV     #BIT6,4(R1)     ;LOAD PORT4
1552  011542  012761  122113  000006          MOV     #122113,6(R1)   ;LOAD INSTRUCTION
1553  011550  052711  000400                  BIS     #BIT8,(R1)      ;CLOCK INSTRUCTION(SET DTR)
1554  011554  012761  021264  000006          MOV     #021264,6(R1)   ;LOAD INSTRUCTION
1555  011562  052711  000400                  BIS     #BIT8,(R1)      ;CLOCK INSTRUCTION(READ MODEM REG)
1556  011566  032761  000010  000004          BIT     #BIT3,4(R1)     ;IS MRDY SET NOW?
1557  011574  001402                          BEQ     20$             ;BR IF NO CONNECTOR
1558  011576  052712  040000                  BIS     #BIT14,(R2)     ;SET STATUS BIT FOR CONNECTOR
1559  011602  005722                20$:      TST     (R2)+           ;POP POINTER
1560  011604  012761  021324  000006          MOV     #021324,6(R1)   ;PUT INSTRUCTION IN PORT6
1561  011612  012711  001400                  MOV     #BIT9!BIT8,(R1) ;PORT4←LU 15
1562  011616  156122  000004                  BISB    4(R1),(R2)+     ;STORE DDCMP LINE # IN TABLE
1563  011622  012761  021344  000006          MOV     #021344,6(R1)   ;PORT6←INSTRUCTION
1564  011630  012711  001400                  MOV     #BIT8!BIT9,(R1) ;CLOCK INSTR.
1565  011634  156122  000004                  BISB    4(R1),(R2)+     ;STORE BM873 ADD IN TABLE
1566  011640  005722                          TST     (R2)+           ;POP OVER STAT3
1567  011642  005011                          CLR     (R1)            ;CLEAR ROMI
1568  011644  005237  001310                  INC     DMNUM           ;UPDATE DEVICE COUNTER
1569  011650  022737  000020  001310          CMP     #20,DMNUM       ;ARE MAX. NO. OF DEV FOUND?
1570  011656  001412                          BEQ     13$             ;YES DON'T LOOK FOR ANY MORE.
1571  011660  005011                3$:       CLR     (R1)            ;CLEAR BIT 10
1572  011662  005061  000006                  CLR     6(R1)           ;CLEAR SEL 6
1573  011666  062701  000010        14$:      ADD     #10,R1          ;UPDATE CSR POINTER ADDRESS
1574  011672  022701  164000                  CMP     #164000,R1
1575  011676  001402                          BEQ     13$             ;BR IF DONE
1576  011700  000137  011266                  JMP     2$              ;JUMP IF NOT
1577  011704  005037  001306        13$:      CLR     DMACTV
1578  011710  005737  001310                  TST     DMNUM           ;WERE ANY DMC11'S FOUND AT ALL?
1579  011714  001423                          BEQ     5$              ;ERROR AUTO SIZER FOUND NO DMC11'S IN THIS SYS.
1580  011716  013701  001310                  MOV     DMNUM,R1
1581  011722  010137  001314                  MOV     R1,SAVNUM       ;SAVE NUMBER OF DEVICES
1582  011726  000241                4$:       CLC
1583  011730  006137  001306                  ROL     DMACTV          ;GENERATE ACTIVE REGISTER OF DEVICES.
1584  011734  005237  001306                  INC     DMACTV          ;SET THE BIT
1585  011740  005301                          DEC     R1
1586  011742  001371                          BNE     4$              ;BR IF MORE TO GENERATE
1587  011744  012737  000006  000004          MOV     #6,@#4          ;RESTORE TRAP VECTOR
1588  011752  013737  001306  001312          MOV     DMACTV,SAVACT   ;SAVE ACTIVE REGISTER
1589  011760  000137  012012                  JMP     VECMAP          ;GO FIND THE VECTOR NOW.
1590  011764  104402  005761        5$:       TYPE    ,MERR2          ;NOTIFY OPR THAT NO DMC11'S FOUND.
1591  011770  005000                          CLR     R0              ;MAKE DATA LIGHTS ZERO
1592  011772  000000                          HALT                    ;STOP THE SHOW
1593  011774  000776                          BR      .-2             ;DISABLE CONT. SW.
1594  011776  012716  011666        6$:       MOV     #14$,(SP)       ;ENTERED BY NON-EXISTANT TIME-OUT.
1595  012002  000002                          RTI                     ;RETURN TO MAINSTREAM
1596
1597  012004  000001                WHICH:    1
1598  012006      002       002               .BYTE   2,2
1599  012010  001256                          TEMP5
1600
1601  012012  032737  000001  001236  VECMAP: BIT     #SW00,STRTSW
1602  012020  001114                          BNE     5$
1603  012022  012737  000340  000022          MOV     #340,@#22       ;SET IOT TRAP PRIO TO 7
1604  012030  012737  012204  000020          MOV     #4$,@#20        ;SET IOT TRAP VECTOR
1605  012036  012702  001500                  MOV     #DM.MAP,R2      ;SET SOFTWARE POINTER
1606  012042  012700  000300                  MOV     #300,R0         ;FLOATING VECTORS START HERE.
```

```
1607  012046  012701  000302            MOV     #302,R1         ;PC OF IOT INSTR.
1608  012052  010120            1$:     MOV     R1,(R0)+        ;START FILLING VECTOR AREA
1609  012054  012721  000004            MOV     #4,(R1)+        ;WITH .+2; IOT
1610  012060  022021                    CMP     (R0)+,(R1)+     ;ADD 2 TO R0 +R1
1611  012062  020127  001000            CMP     R1,#1000
1612  012066  101771                    BLOS    1$              ;BR IF MORE TO FILL
1613  012070  013737  001306  001246    MOV     DMACTV,TEMP1    ;STORE TEMPORALLY
1614  012076  006037  001246    2$:     ROR     TEMP1           ;BRING OUT A BIT
1615  012102  103063                    BCC     5$              ;BR IF ALL DONE
1616  012104  012704  000012            MOV     #12,R4          ;R4 IS INDEX REGISTER
1617  012110  016437  012254  177776    MOV     BRLVL(R4),PS    ;SET PS TO 7
1618  012116  011201                    MOV     (R2),R1
1619  012120  012761  000200  000004    MOV     #200,4(R1)
1620  012126  012711  001000            MOV     #BIT9,(R1)      ;SET ROMI
1621  012132  012761  121111  000006    MOV     #121111,6(R1)   ;PUT INSTRUCTION IN PORT6
1622  012140  012711  001400            MOV     #BIT9!BIT8,(R1) ;FORCE AN INTERRUPT
1623  012144  105200            7$:     INCB    R0              ;STALL
1624  012146  001376                    BNE     .-2             ;FOR TIME TO INTERUPT
1625  012150  162704  000002            SUB     #2,R4           ;GET NEXT LOWEST PS LEVEL
1626  012154  001404                    BEQ     6$              ;BR IF R4 = 0
1627  012156  016437  012254  177776    MOV     BRLVL(R4),PS    ;MOVE NEXT LOWER LEVEL IN PS
1628  012164  000767                    BR      7$              ;BR TO DELAY
1629  012166  052762  005300  000002  6$:BIS     #5300,2(R2)     ;NO INTERUPT ASSUME 300 AT LEVEL 5 AND FIX DMC11 LATER
1630  012174  005011            3$:     CLR     (R1)            ;CLEAR ROMI
1631  012176  062702  000010            ADD     #10,R2          ;POP SOFTWARE POINTER
1632  012202  000735                    BR      2$              ;KEEP GOING
1633  012204  051662  000002    4$:     BIS     (SP),2(R2)      ;GET VECTOR ADDRESS
1634  012210  042762  000007  000002    BIC     #7,2(R2)        ;CLEAR JUNK
1635  012216  016405  012256            MOV     BRLVL+2(R4),R5  ;GET BR LEVEL OF DMC11
1636  012222  006305                    ASL     R5              ;SHIFT LEVEL 4 PLACES
1637  012224  006305                    ASL     R5              ;TO THE LEFT FOR THE
1638  012226  006305                    ASL     R5              ;STATUS TABLE
1639  012230  006305                    ASL     R5
1640  012232  042705  170777            BIC     #170777,R5      ;CLEAR UNWANTED BITS
1641  012236  050562  000002            BIS     R5,2(R2)        ;PUT BR LEVEL IN STATUS TABLE
1642  012242  022626                    CMP     (SP)+,(SP)+     ;POP IOT JUNK OFF STACK
1643  012244  012716  012174            MOV     #3$,(SP)        ;SET FOR RETURN
1644  012250  000002                    RTI
1645  012252  000207            5$:     RTS     PC              ;ALL DONE WITH "AUTO SIZING"
1646
1647  012254  000000            BRLVL:  0       ;LEVEL 0
1648  012256  000000                    0       ;LEVEL 0
1649  012260  000200                    200     ;LEVEL 4
1650  012262  000240                    240     ;LEVEL 5
1651  012264  000300                    300     ;LEVEL 6
1652  012266  000340                    340     ;LEVEL 7
1653
1654
1655  012270  105777  166710    INTTY:  TSTB    @TKCSR          ;WAIT FOR DONE
1656  012274  100375                    BPL     .-4
1657  012276  017703  166704            MOV     @TKDBR,R3       ;PUT CHAR IN R3
1658  012302  105777  166702            TSTB    @TPCSR          ;WAIT UNTIL PRINTER IS READY
1659  012306  100375                    BPL     .-4
1660  012310  010377  166676            MOV     R3,@TPDBR       ;ECHO CHAR
1661  012314  042703  000240            BIC     #BIT7!BIT5,R3   ;MASK OFF LOWER CASE
1662  012320  000207                    RTS     PC              ;RETURN
```

```
1663
1664
1665  012322  000000          ROMMAP: 0                    ;POINTER TO HI OR LO SPEED MICRO-CODE
1666
1667  012324                  LOMAP:                       ;LOW SPEED (REMOTE) MICRO-CODE
```

DMCNEW.MAC 26-OCT-77 13:33 TABLE OF CONTENTS

DMCGAL.MAC 18-JAN-78 16:05 TABLE OF CONTENTS

```
                        .TITLE  DMC-11 MICROPROCESSOR INSTRUCTIONS
                        .SBTTL  MACRO DEFINITIONS
                        ;
                        .SBTTL REVISION 00
                        .SBTTL FEBRUARY 25, 1975
                        .SBTTL
                        .SBTTL REVISION 01
                        .SBTTL MARCH 18,1975
                        .SBTTL NEW CSR BOARD CHANGES
                        .SBTTL
                        .SBTTL HARVEY M. SCHLESINGER
                        ;
                        .SBTTL COPYRIGHT 1975 DIGITAL EQUIPMENT CORPORATION
                        ;
```

```
                        NEW-0
                        ;MICROPROCESSOR INSTRUCTION WORD DEFINITIONS
000000                  MOVE=0                  ;OPCODE MOVE
100000                  JUMP=100000             ;OPCODE JUMP
020000                  IBUS=20000              ;SOURCE IBUS
000000                  IMM=0                   ;SOURCVE IMMEDIATE
040000                  MEMX=40000              ;SOURCE MEMORY
060000                  BRX=60000               ;SOURCE BR
060000                  BR=60000                ;SOURCE BR

060000                  DP=60000                ;SOURCE BR
010000                  LDMAR=10000             ;MA-LOAD MAR LO
014000                  INCMAR=14000            ;MA-INCREMENT MAR
000400                  WRTEBR=400              ;DEST-WRITE BR
001000                  WROUTX=1000             ;DEST-EXTENDED IBUS
001400                  SHFTBR=1400             ;DEST-SHIFT BR LEFT
002000                  WROUT=2000              ;DEST-WRITE OUTPUT
002400                  WRMEM=2400              ;DEST-WRITE MEMORY
003000                  SPX=3000                ;DEST-WRITE SP
003400                  SPBRX=3400              ;DEST-WRITE SP AND BR
                        ;FUNCTIONS
000200                  SELA=200                ;FUNCTION-SELECT A
000220                  SELB=220                ;FUNCTION-SELECT B
000240                  AORNB=240               ;FUNCTION-A OR NOT B
000260                  AANDB=260               ;FUNCTION A  AND B
000300                  AORB=300                ;FUNCTION-A OR B
000320                  AXORB=320               ;FUNCTION A XOR B
000340                  SUB=340                 ;SUBTRACT
                           SUBTC=360            ;FUNCTION- TWOS COMPLEMENT SUBTRACT
000000                  ADD=0                   ;ADD A+B
000020                  ADDC=20                 ;A+B+CARRY
000040                  SUBC=40                 ;A-B-C
000060                  INCA=60                 ;INCREMENT A
000100                  AC=100                  ;A PLUS CARRY
000120                  AA=120                  ;A PLUS A
000140                  AAC=140                 ;A PLUS A PLUS C
000160                  DECA=160                ;DECREMENT A
                        ;END FUNCTIONS
                        PAGE1=4000
                        PAGE2=10000
                        PAGE3=14000
001000                  CCOND=1000              ;CONDITION C
001400                  ZCOND=1400              ;CONDITION Z
000400                  ALCOND=400              ;ALWAYS
002000                  BR0CON=2000             ;CONDITION BR0
002400                  BR1CON=2400             ;CONDITION BR1
003000                  BR4CON=3000             ;CONDITION BR4
003400                  BR7CON=3400             ;CONDITION BR7
```

```
                                        .SBTTL  MICRO INSTRUCTION DEFINITIONS
                                        .SBTTL          BRANCH INSTRUCTIONS
                                ;
        100000                  JUMP=100000             ;JUMP OP CODE
                                ;
                                ;
                                ;
                                ;
                                ;
                                ;
                                ;
                                ;
                                        .SBTTL          INDEXED BRANCH INSTRUCTIONS
                                ;
                                ;
                                ;
                                ;
                                ;
                                ;
                                ;
                                        .SBTTL          MOVE INSTRUCTIONS
                                ;
        000000                  MOVE=0          ;MOVE OPCODE
                                ;
                                ;
                                ;
                                ;
                                ;
                                ;
                                ;
                                ;
                                ;
                                ;
                                ;
                                ;
                                ;
                                ;
                                ;
                                ;
                                ;
```

```
                        .SBTTL  INPUT/OUTPUT ASSIGNMENTS
                        ;IBUS   ASSIGNMENTS
100000                  INCON=0+100000          ;IN CONTROL CSR
100020                  MAIN=20+100000          ;MAINTAINENCE REGISTER
100040                  OCON=40+100000          ;OUT CONTROL CSR
100060                  UBADDR=60+100000        ;UNUSED
100100                  PORT1=100+100000        ;CSR4
100120                  PORT2=120+100000        ;CSR5
100140                  PORT3=140+100000        ;CSR6
100160                  PORT4=160+100000        ;CSR7
100200                  NPR=200+100000          ;NPR CONTROL
100220                  UBBR=220+100000         ;BR(INTERRUPT)CONTROL
000000                  INDAT1=0                ;INPUT DATA LOW BYTE
000020                  INDAT2=20               ;INPUT DATA HIGH BYTE
000140                  IOBA1=140               ;OUTPUT BA LOW BYTE
000160                  IOBA2=160               ;OUTPUT BA HIGH BYTE
000100                  IIBA1=100               ;INPUT BA LOW BYTE
000120                  IIBA2=120               ;INPUT BA HIGH BYTE
000200                  RCVDAT=200              ;RECEIVE DATA
000220                  TMTCON=220              ;TMTR CONTROL
000240                  RCVCON=240              ;RCVR CONTROL
000260                  MODEM=260               ;MODEM CONTROL
000300                  SYNREG=300              ;SYN REGISTER
000320                  LNOSW=320               ;LINE NUMBER SWITCH
000340                  BM873=340               ;BM873 ADDRESS
000360                  LUMAIN=360              ;LINE UNIT MAINTAINENCE
                        ;OBUS   ASSIGNMENTS
                        ;EXTENDED OBUS
000000                  OINCON=0                ;IN CONTROL CSR
000001                  OMAIN=1                 ;MAINT
000002                  OOCON=2                 ;OUT CONTROL CSR
000003                  OUBADD=3                ;UNUSED
000004                  OPORT1=4                ;CSR4
000005                  OPORT2=5                ;CSR5
000006                  OPORT3=6                ;CSR6
000007                  OPORT4=7                ;CSR7
000010                  ONPR=10                 ;NPR CONTROL
000011                  OBR=11                  ;BR CONTROL
                        ;UNEXTENDED OBUS
000000                  OINDAT1=0               ;OUTPUT OF.
000002                  OUTDA1=2                ;OUTPUT DATA LOW BYTE
000003                  OUTDA2=3                ;OUTPUT DATA HIGH BYTE
000006                  OBA1=6                  ;OUTPUT BA LOW BYTE
000007                  OBA2=7                  ;OUTPUT BA HIGH BYTE
000004                  IBA1=4                  ;INPUT BA LOW BYTE
000005                  IBA2=5                  ;INPUT BA HIGH BYTE
000010                  TMTDAT=10               ;TMTR DATA
000011                  OTMTCO=11               ;TMTR CONTROL
000012                  ORCVCO=12               ;RCVR CONTROL
000013                  OMODEM=13               ;MODEM CONTROL
000014                  SYNC=14                 ;SYN REGISTER
000017                  OLUMAN=17               ;LINE UNIT MAINT.
```

```
                        .SBTTL PROTOCOL DEPENDANT MACROS
                        ;
                        ;
                        ;
                        ;
                        ;
                        ;

177777                  MICPC=177777    ;INIT MICRO PC
```

```
                        .SBTTL  DMC11 DDCMP MICRO CODE ASSEMBLED FOR USE  WITH THE M8201 LINE UNIT
000000                  LOW=0
000000                  $LOW=0
```

```
                        .ABS
                        .LIST ME
                        ;
                                .TITLE DMC11 DDCMP PROTOCOL IMPLEMENTATION
                        ;
                        ; COPYRIGHT (C) 1977
                        ; DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.
                        ;
                        ; THIS SOFTWARE IS FURNISHED UNDER A LICENSE FOR USE ONLY ON A
                        ; SINGLE COMPUTER SYSTEM AND MAY  BE  COPIED   ONLY  WITH  THE
                        ; INCLUSION OF THE ABOVE COPYRIGHT NOTICE.  THIS SOFTWARE,  OR
                        ; ANY OTHER COPIES THEREOF, MAY NOT BE PROVIDED  OR  OTHERWISE
                        ; MADE AVAILABLE TO ANY OTHER PERSON  EXCEPT FOR  USE ON SUCH
                        ; SYSTEM AND TO ONE WHO AGREES TO THESE LICENSE  TERMS.  TITLE
                        ; TO AND OWNERSHIP OF THE SOFTWARE SHALL AT ALL  TIMES  REMAIN
                        ; IN DEC.
                        ;
                        ; THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT
                        ; NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL
                        ; EQUIPMENT CORPORATION.
                        ;
                        ; DEC ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF
                        ; ITS SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DEC.
                        ;
                                .SBTTL VERSION 00A FEBRUARY 26,1975
                                .SBTTL
                                .SBTTL HARVEY M. SCHLESINGER
                                .SBTTL
                                .SBTTL
                                .SBTTL  VERSION 00B     MARCH 17,1975
                                .SBTTL  CSR AND MICROPROCESSOR CHANGES
                                .SBTTL
                                .SBTTL  VERSION 00C     NOVEMBER 6, 1975
                                .SBTTL  RETRANSMISSION CHANGES
                                .SBTTL
                                .SBTTL  VERSION 00D     DECEMBER 3,1975
                                .SBTTL  TRANSMIT DONE CHANGES
                                .SBTTL
                                .SBTTL  VERSION 01A     6-MAY-77
                                .SBTTL  ACKNOWLEDGEMENT CHANGES - ACK A MESSAGE EVEN IF NO DATA TO RETURN
                                .SBTTL
                                .SBTTL  VERSION 01B     01-DEC-77
                                .SBTTL  MANY CHANGES
                                .SBTTL
                                .SBTTL  THE LATEST MODIFICATIONS WERE ADDED ON:
                                .SBTTL          01-DEC-77
```

```
                        .SBTTL  MICROPROCESSOR MAIN MEMORY ASSIGNMENTS
                        ;ALLOCATION OF MICROPROCESSOR MAIN MEMORY
000000                  NAKSR=0             ;NAKS RECD--DYNAMIC
000001                  NAKST=NAKSR+1       ;NAKS TMTED--DYNAMIC
000002                  REPSR=NAKST+1       ;REPS RECD--DYNAMIC
000003                  REPST=REPSR+1       ;REPS TMTED--DYNAMIC
000006                  NP=REPST+3          ;CONSTANT 0
000007                  NTLR=NP+1           ;NAKS-MSG NO BUFFERS CUMUL.
000010                  NHDR=NTLR+1         ;NAKS-MSG HEADER BAD
000011                  NDATR=NHDR+1        ;NAKS-DATA BAD
000012                  NTLS=NDATR+1        ;NAK SENT --NO BUFFERS
000013                  NHDS=NTLS+1         ;NAK SENT BAD HEADER
000014                  NDATS=NHDS+1        ;NAK SENT BAD DATA
000015                  REPCS=NDATS+1       ;REPS SENT CUMUL
000016                  REPCR=REPCS+1       ;REPS RECD CUMUL
000017                  BASE=REPCR+1                ;CORE TABLE BASE ADDRESS
000022                  SRC=BASE+3                  ;START OF INPUT CHAIN--NEXT RECV DONE
000023                  ERC=SRC+1                   ;END OF INPUT CHAIN
000024                  RCL1=ERC+1                  ;RECEIVE LINK #1
000031                  RCL2=RCL1+5                 ;   "       "   #2
000036                  RCL3=RCL2+5                 ;   "       "   #3
000043                  RCL4=RCL3+5
000050                  RCL5=RCL4+5
000055                  RCL6=RCL5+5
000062                  RCL7=RCL6+5
000067                  STC=RCL7+5                  ;START OF OUTPUT CHAIN---NEXT TMT DONE
000070                  ETC=STC+1                   ;END OF TRANSMIT CHAIN
000071                  TML1=ETC+1                  ;TRANSMIT LINK #1
000077                  TML2=TML1+6                 ;     "        "   #2
000105                  TML3=TML2+6                 ;     "        "   #3
000113                  TML4=TML3+6
000121                  TML5=TML4+6
000127                  TML6=TML5+6
000135                  TML7=TML6+6
000143                  TML8=TML7+6
000151                  T=TML8+6                            ;TYPE FIELD
000152                  ST=T+1                      ;SUBTYPE FIELD
000153                  ISP17=ST+1                  ;MSG ACKED IMAGE
000154                  IMG10=ISP17+1               ;IMAGE OF BIT 1 OF SP10
000155                  IMG11=IMG10+1               ;IMAGE OF SP11
000156                  IMG12=IMG11+1               ;IMAGE OF SP12
000157                  IMG14=IMG12+1               ;IMAGE OF SP14
000160                  IMG16=IMG14+1               ;IMAGE OF SP16
000161                  IMG17=IMG16+1               ;IMAGE OF SP17
000162                  TYPTAB=IMG17+1              ;TYPE TABLE---
                                                    ;72 TYPE TABLE REP
                                                    ;73  "      "   NAK
000164                  TYPSTT=TYPTAB+2             ;74  "      "   START
                                                    ;75  "      "   STACK
                                                    ;
                                                    ;
000167                  BC=TYPSTT+3                 ;RECEIVE BYTE COUNT
000171                  ISP11=BC+2                  ;SP11 IMAGE
000172                  ISP12=ISP11+1               ;SP12 IMAGE
000173                  INCONS=ISP12+1              ;IN CONTROL CSR IMAGE
000174                  RTHRS=INCONS+1              ;RECV THRESHOLD LINK
```

```
465
466
467          000210          TABST=210          ;TABLE UPDATE STATE
468          000211          PRTST=TABST+1      ;PORT STATE
469          000240          NXTINT=240         ;NEXT INTERRUPT POSITION
470          000241          NXTSP=NXTINT+1     ;END OF INTERUPT CHAIN
471          000242          INTSTK=NXTSF+1     ;STACK OF INTERUPTS
472          000400          MMEND=400          ;MAIN MEMORY END
```

;ALL LOCATIONS FROM 200 ON ARE NOT WRITTEN OUT DURING A TABLE UPDATE

```
                                .SBTTL  SCRATCH PAD ASSIGNMENTS
000000                          SP0=0   ;SP0---SCRATCH REGISTER
000001                          SP1=1   ;SP1---PORT STATUS WORD
                                        ;BIT ASSIGNMENTS
                                        ;BIT0--INIT MODE
                                        ;BIT1--NO BUFFER ASSIGNED IN BOOT MODE
                                        ;BIT2--UNUSED
                                        ;BIT3--DLE RECEIVED WHILE NOT IN MAINT MODE
                                        ;BIT4--INTERRUPT PENDING
                                        ;BIT6--DISCONNECT ERROR
                                        ;BIT7--BOOT MODE
000002                          SP2=2   ;SP2---TRANSMIT STATE POINTER
000003                          SP3=3   ;SP3---RECEIVE STATE POINTER
000004                          SP4=4   ;SP4---END RECV ADDRESS
000005                          SP5=5   ;SP5---END RECEIVE ADDRESS
000006                          SP6=6   ;SP6---END TRANSMIT ADDRESS
000007                          SP7=7   ;SP7---END TRANSMIT ADDRESS
000010                          SP10=10 ;SP10---LINE STATUS WORD
                                        ;BIT ASSIGNMENTS
                                        ;BIT0--UNNUMB PENDING
                                        ;BIT1--MESSAGE IN PROGRESS
                                        ;BIT2--LINE HAS GONE IDLE
                                        ;BIT3--START RECEIVED
                                        ;BIT4--CLEAR ACTIVE ON END
                                        ;BIT5--START MODE
                                        ;BIT6--HALF DUPLEX
                                        ;BIT7--OK TO SEND
000011                          SP11=11 ;SP11---R FIELD
000012                          SP12=12 ;SP12---N FIELD
000013                          SP13=13 ;SP13---TYPE
000014                          SP14=14 ;SP14---TIMER ENTRY---NUMBER OF ONE SECOND TICKS
000015                          SP15=15 ;SP15---RECEIVE LINK IMAGE
000016                          SP16=16 ;SP16---POINTER TO TMT LINK COPY IN MAIN MEM
000017                          SP17=17 ;SP17---LAST MESSAGE ACKNOWLEDGED
```

```
                                .SBTTL  INIT--INITIALIZATION ROUTINE
                                ;ZEROS MAIN MEMORY
                                ;LOOPS WAITING FOR RECEIVE DATA(BOOT?)
                                ;OR FOR RQI TO BE SET
                                ;WILL ACCEPT ONLY BASE FORMAT. ALL OTHERS WILL RETURN A PROCEDURE ERROR
                                ;
                                ;AT INITIALIZATION --- THE HARDWARE CLEARS THE BR AND MAR
                    027654  .=27654
027654              INIT:   SP      BR,SELB,SP0             ;CLEAR SP0
(1)         000000          MICPC=MICPC+1
(1)  027654 063220          <MOVE!SPX!BR!SELB!SP0>
027656              SP      BR,SELB,SP3             ;PAGE ONE TRANSFER ADDRESS
(1)         000001          MICPC=MICPC+1
(1)  027656 063223          <MOVE!SPX!BR!SELB!SP3>
027660              SP      BR,SELB,SP17            ;CLEAR SP17
(1)         000002          MICPC=MICPC+1
(1)  027660 063237          <MOVE!SPX!BR!SELB!SP17>
027662              SP      BR,SELB,SP12            ;CLEAR SP12
(1)         000003          MICPC=MICPC+1
(1)  027662 063232          <MOVE!SPX!BR!SELB!SP12>
027664              OUT     BR,<SELA!0INCON>        ;ZERO THE IN CONTROL CSR
(1)         000004          MICPC=MICPC+1
(1)  027664 061200          <MOVE!WROUTX!BR!<SELA!0INCON>>
027666              OUT     BR,<SELA!0OCON>         ;ZERO THE OUT CONTROL CSR
(1)         000005          MICPC=MICPC+1
(1)  027666 061202          <MOVE!WROUTX!BR!<SELA!0OCON>>
027670              SP      IMM,370,SP10            ;WRITE 5 ONE BITS TO THE HIGH ORDER
(1)         000006          MICPC=MICPC+1
(1)  027670 003370          <MOVE!SPX!IMM!370!SP10>
                                                    ;BITS OF SP10
027672              5$:     SP      BR,AA,SP10              ;SHIFT SP10 LEFT SETTING CARRY THE
(1)         000007          MICPC=MICPC+1
(1)  027672 063130          <MOVE!SPX!BR!AA!SP10>
                                                    ;FIRST 5 TIMES THRU THE LOOP
027674              MEMINC  BR,ADDC!SP3             ;WRITE A ONE TO THE FIRST 5 MEMORY
(1)         000010          MICPC=MICPC+1
(1)  027674 076423          <MOVE!WRMEM!INCMAR!BR!<ADDC!SP3>>
                                                    ;LOCATIONS AND ZERO THE REST
027676              SP      BR,INCA,SP0             ;INCREMENT COUNTER
(1)         000011          MICPC=MICPC+1
(1)  027676 063060          <MOVE!SPX!BR!INCA!SP0>
027700              Z       10$                     ;ALL DONE
(1)         000012          MICPC=MICPC+1
(1)  027700 101414          <JUMP!ZCOND!<10$-INIT&3000*4>!<10$-INIT&777/2>>
027702              ALWAYS  5$                      ;KEEP GOING
(1)         000013          MICPC=MICPC+1
(1)  027702 100407          <JUMP!ALCOND!<5$-INIT&3000*4>!<5$-INIT&777/2>>
027704              10$:    SPBR    IMM,1,SP1               ;WRITE A 1 TO THE BR AND SP1
(1)         000014          MICPC=MICPC+1
(1)  027704 003401          <MOVE!SPBRX!IMM!1!SP1>
027706              SP      BR,SELB,SP11            ;WRITE A 1 TO SP11
(1)         000015          MICPC=MICPC+1
(1)  027706 063231          <MOVE!SPX!BR!SELB!SP11>
027710              LDMA    IMM,TYPTAB              ;POINT MAR TO TYPE TABLE
(1)         000016          MICPC=MICPC+1
(1)  027710 010162          <MOVE!LDMAR!IMM!<TYPTAB&377>>
```

```
  535  027712
  (1)            000017
  (1)  027712    000626
  536  027714
  (1)            000020
  (1)  027714    062234
  537  027716
  (1)            000021
  (1)  027716    016403
  538  027720
  (1)            000022
  (1)  027720    016402
  539  027722
  (1)            000023
  (1)  027722    016406
  540  027724
  (1)            000024
  (1)  027724    016407
  541  027726
  (1)            000025
  (1)  027726    016401
  542  027730
  (1)            000026
  (1)  027730    010210
  543  027732
  (1)  027732
  (2)            000027
  (2)  027732    016455
  544  027734
  (1)  027734
  (2)            000030
  (2)  027734    016461
  545  027736
  (1)            000031
  (1)  027736    010067
  546  027740
  (1)            000032
  (1)  027740    016471
  547  027742
  (1)            000033
  (1)  027742    002471
  548  027744
  (1)            000034
  (1)  027744    043236
  549  027746
  (1)            000035
  (1)  027746    010022
  550  027750
  (1)            000036
  (1)  027750    016424
  551  027752
  (1)            000037
  (1)  027752    002424
  552  027754
  (1)            000040
  (1)  027754    043235
```

```
        BRWRTE  IMM,226                 ;WRITE SYNC TO MEMORY
        MICPC=MICPC+1
        <MOVE!WRITEBR!IMM!<226>>
        OUTPUT  BR,SELB!SYNC    ;LOAD THE SYNC REGISTER
        MICPC=MICPC+1
        <MOVE!WROUT!BR!<SELB!SYNC>>
        MEMINC  IMM,3                   ;REP
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<3>>
        MEMINC  IMM,2                   ;NAK
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<2>>
        MEMINC  IMM,6                   ;START
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<6>>
        MEMINC  IMM,7                   ;STACK
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<7>>
        MEMINC  IMM,1                   ;ACK
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<1>>
        LDMA    IMM,TABST               ;POINT TO TABLE UPDATE STATE
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<TABST&377>>
        PSTATI  I3                      ;INITIALIZE IT
        MEMINC  IMM,<<I3-INIT&777/2>>
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<<I3-INIT&777/2>>>
        PSTATI  NIDLE2                  ;INITIALIZE PORT STATUS
        MEMINC  IMM,<<NIDLE2-INIT&777/2>>
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<<NIDLE2-INIT&777/2>>>
        LDMA    IMM,STC                 ;LOAD ADDRESS OF LAST TMT CHAIN
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<STC&377>>
        MEMINC  IMM,TML1                ;STORE ADDRESS OF FIRST TMT LINK
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<TML1>>
        MEM     IMM,TML1
        MICPC=MICPC+1
        <MOVE!WRMEM!IMM!<TML1>>
        SP      MEMX,SELB,SP16          ;INITIALIZE LAST XMIT POINTER
        MICPC=MICPC+1
        <MOVE!SPX!MEMX!SELB!SP16>
        LDMA    IMM,SRC                 ;LOAD ADDRESS OF LAST RECV CHAIN
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<SRC&377>>
        MEMINC  IMM,RCL1                ;SET UP ADDRESS OF FIRST RECV LINK
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<RCL1>>
        MEM     IMM,RCL1
        MICPC=MICPC+1
        <MOVE!WRMEM!IMM!<RCL1>>
        SP      MEMX,SELB,SP15
        MICPC=MICPC+1
        <MOVE!SPX!MEMX!SELB!SP15>
```

```
   553  027756
   (1)          000041
   (1)  027756  010240
   554  027760
   (1)          000042
   (1)  027760  016642
   555  027762
   (1)          000043
   (1)  027762  002642
   556  027764
   (1)          000044
   (1)  027764  000600
   557  027766
   (1)          000045
   (1)  027766  061221
   558
   559  027770
   (1)          000046
   (1)  027770  110642
```

```
        LDMA    IMM,NXTINT              ;ADDRESS OF NEXT INTERRUPT POINTER TO MAR
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<NXTINT&377>>
        MEMINC  IMM,INTSTK                      ;INITIALIZE NEXT INTERRUPT POINTER
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<INTSTK>>
        MEM     IMM,INTSTK                      ;INITIALIZE INSERTION POINTER
        MICPC=MICPC+1
        <MOVE!WRMEM!IMM!<INTSTK>>
        BRWRTE  IMM,200                 ;MASK TO SET RUN BIT
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<200>>
        OUT     BR,<SELB!OMAIN>         ;WRITE  THE RUN BIT TO MAINT CSR
        MICPC=MICPC+1
        <MOVE!WROUTX!BR!<SELB!OMAIN>>
                                        ;FALL INTO IDLE LOOP
        ALWAYS  TEOM2
        MICPC=MICPC+1
        <JUMP!ALCOND!<TEOM2-INIT&3000*4>!<TEOM2-INIT&777/2>>
```

```
                                .SBTTL  IDLE--PROGRAM IDLE LOOP
                                ;PROGRAM IDLE LOOP
                                ;DISPATCHES TO APPROPRIATE SERVICE ROUTINES
                                ;USES STATE POINTERS FOR TMT,RCV,CSR ACTIVITY
                                ;
 561                    IDLE:
 562                            BRWRTE  IBUS,TMTCON
 563                            MICPC=MICPC+1
 564                            <MOVE!WRTEBR!IBUS!<TMTCON>>
 565                            .BR4    DP,SELA,<2!PAGE2>
 566 027772                    MICPC=MICPC+1
 567 027772                    <JUMP! BR4CON!DP!SELA!2!PAGE2>
 (1)         000047     I1:    BRWRTE  IBUS,RCVCON             ;READ LINE UNIT RECEIVE CONTROL WORD
 (1)  027772 020620             MICPC=MICPC+1
 568 027774                    <MOVE!WRTEBR!IBUS!<RCVCON>>
 (1)         000050            .BR4    BR,SELA,SP3!PAGE1       ;BRANCH BASED UPON RECV STATE
 (1)  027774 173202             MICPC=MICPC+1
 569 027776                    <JUMP! BR4CON!BR!SELA!SP3!PAGE1>
 (1)         000051     I2:    LDMA    IMM,TABST               ;POINT TO TABLE UPDATE STATE
 (1)  027776 020640             MICPC=MICPC+1
 570 030000                    <MOVE!LDMAR!IMM!<TABST&377>>
 (1)         000052            .ALWAY  MEMX,SELB,0
 (1)  030000 167203             MICPC=MICPC+1
 571 030002                    <JUMP!ALCOND!MEMX!SELB!0>
 (1)         000053     I3:
 (1)  030002 010210     IDLE0: SPBR    IBUS,UBBR,SPO           ;TIMER EXPIRES?
 572 030004                     MICPC=MICPC+1
 (1)         000054             <MOVE!SPBRX!IBUS!UBBR!SPO>
 (1)  030004 140620             BR4     TIMSRV
 573 030006                     MICPC=MICPC+1
 582 030006                     <JUMP!BR4CON!<TIMSRV-INIT&3000*4>!<TIMSRV-INIT&777/2>>
 (1)         000055     IDLE1:
 (1)  030006 123620             LDMA    IMM,PRTST               ;ADDRESS PORT STATE
 583 030010                     MICPC=MICPC+1
 (1)         000056             <MOVE!LDMAR!IMM!<PRTST&377>>
 (1)  030010 113246             .ALWAY  MEMX,SELB,0             ;INDEX
 584 030012                     MICPC=MICPC+1
 590 030012                     <JUMP!ALCOND!MEMX!SELB!0>
 (1)         000057
 (1)  030012 010211
 591 030014
 (1)         000060
 (1)  030014 140620
```

```
                        .SBTTL  NIDLE2---NO CSR ACTIVITY STATE
NIDLE2:  BRWRTE  BR,SELA!SP1            ;READ PORT STATUS WORD
         MICPC=MICPC+1
         <MOVE!WRTEBR!BR!<SELA!SP1>>
         BR4     OUTINT
         MICPC=MICPC+1
         <JUMP!BR4CON!<OUTINT-INIT&3000*4>!<OUTINT-INIT&777/2>>
         SPBR    IBUS,INCON,SPO         ;READ INPUT CONTROL CSR
         MICPC=MICPC+1
         <MOVE!SPBRX!IBUS!INCON!SPO>
         BRSHFT                         ;SHIFT IT RIGHT
         MICPC=MICPC+1
         <MOVE!SHFTBR!WRTEBR!SELB>
         BR4     INWAT1                 ;IF RQI SET -- BRANCH
         MICPC=MICPC+1
         <JUMP!BR4CON!<INWAT1-INIT&3000*4>!<INWAT1-INIT&777/2>>
                                        ;TO RE-READ THE IN CNTRL REGISTER TO AVOID
                                        ;A RACE IN MICRO-P READ/UNIBUS WRITE
NIDLE6:
10$:     SPBR    IBUS,MODEM,SPO         ;READ MODEM CONTROL CSR
         MICPC=MICPC+1
         <MOVE!SPBRX!IBUS!MODEM!SPO>
         BRWRTE  BR,AA!SPO              ;SHIFT IT LEFT
         MICPC=MICPC+1
         <MOVE!WRTEBR!BR!<AA!SPO>>
         BR4     SETDSR                 ;IF DSR SET, CLEAR FLAG
         MICPC=MICPC+1
         <JUMP!BR4CON!<SETDSR-INIT&3000*4>!<SETDSR-INIT&777/2>>
         BRWRTE  BR,SELA!SP10           ;READ LINE STATUS WORD
         MICPC=MICPC+1
         <MOVE!WRTEBR!BR!<SELA!SP10>>
         BRSHFT
         MICPC=MICPC+1
         <MOVE!SHFTBR!WRTEBR!SELB>
         BR4     IDLE                   ;START MODE
         MICPC=MICPC+1
         <JUMP!BR4CON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
         BRWRTE  BR,AA!SP1              ;READ PORT STATUS WORD
         MICPC=MICPC+1
         <MOVE!WRTEBR!BR!<AA!SP1>>
         BR1     IDLE                   ;INIT MODE
         MICPC=MICPC+1
         <JUMP!BR1CON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
         BR7     IDLE                   ;DISCONNECT ERROR ALREADY SENT
         MICPC=MICPC+1
         <JUMP!BR7CON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
         SPBR    IBUS,MAIN,SPO   ;READ THE MAINT REGISTER
         MICPC=MICPC+1
         <MOVE!SPBRX!IBUS!MAIN!SPO>
         BRWRTE  BR,ADD!SPO             ;SHIFT LEFT
         MICPC=MICPC+1
         <MOVE!WRTEBR!BR!<ADD!SPO>>
         BR4     IDLE                   ;LU LOOP -- EXIT
         MICPC=MICPC+1
         <JUMP!BR4CON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
         BRWRTE  IMM,100                ;WRITE DISCONNECT ERROR
```

DMCGAL.MAC 18-JAN-78 16:05

```
                                NIDLE2---NO CSR ACTIVITY STATE

(1)             000102                  MICPC=MICPC+1
(1)  030060     000500                  <MOVE!WRTEBR!IMM!<100>>
625  030062                             SP      BR,AORB,SP1             ;FLAG ERROR RECORDED
(1)             000103                  MICPC=MICPC+1
(1)  030062     063301                  <MOVE!SPX!BR!AORB!SP1>
626  030064                             ALWAYS  ERRXX                   ;MAKE A CONTROL OUT
(1)             000104                  MICPC=MICPC+1
(1)  030064     114657                  <JUMP!ALCOND!<ERRXX-INIT&3000*4>!<ERRXX-INIT&777/2>>
627  030066                     SETDSR:  BRWRTE  IMM,277                 ;CLEAR DISCONNECT ERROR FLAG
(1)             000105                  MICPC=MICPC+1
(1)  030066     000677                  <MOVE!WRTEBR!IMM!<277>>
628  030070                             ALWAYS  CLRIDL
(1)             000106                  MICPC=MICPC+1
(1)  030070     100646                  <JUMP!ALCOND!<CLRIDL-INIT&3000*4>!<CLRIDL-INIT&777/2>>
```

```
   635  030072
   (1)          000107
   (1)  030072  123400
   636  030074
   (1)          000110
   (1)  030074  103512
   637  030076
   (1)          000111
   (1)  030076  114444
   638  030100
   (1)          000112
   (1)  030100  060520
   639  030102
   (1)          000113
   (1)  030102  103606
   640  030104
   (1)          000114
   (1)  030104  120400
   641  030106
   (1)          000115
   (1)  030106  001620
   643  030110
   (1)          000116
   (1)  030110  103066
   650  030112
   (1)          000117
   (1)  030112  123400
   651  030114
   (1)          000120
   (1)  030114  102532
   652  030116
   (1)  030116
   (2)          000121
   (2)  030116  002655
   653  030120
   (1)          000122
   (1)  030120  102127
   654  030122
   (1)          000123
   (1)  030122  001620
   655  030124
   (1)  030124
   (2)          000124
   (2)  030124  002723
   656  030126
   (1)          000125
   (1)  030126  102527
   657  030130
   (1)  030130
   (2)          000126
   (2)  030130  002702
   658  030132
   (1)          000127
   (1)  030132  060601
   659  030134
   (1)          000130
```

```
IEIWAT: SPBR    IBUS,INCON,SPO          ;READ INPUT CONTROL CSR
        MICPC=MICPC+1
        <MOVE!SPBRX!IBUS!INCON!SPO>
        BR7     10$                     ;RDYI STILL SET
        MICPC=MICPC+1
        <JUMP!BR7CON!<10$-INIT&3000*4>!<10$-INIT&777/2>>
        ALWAYS  INWAT1                  ;GOT CLEARED BY 11 RE-SET IT
        MICPC=MICPC+1
        <JUMP!ALCOND!<INWAT1-INIT&3000*4>!<INWAT1-INIT&777/2>>
10$:    BRWRTE  BR,<AA!SPO>             ;CHECK IEI
        MICPC=MICPC+1
        <MOVE!WRTEBR!BR!<AA!SPO>>
        BR7     ININT                   ;IEI SET, GENERATE INTERRUPT
        MICPC=MICPC+1
        <JUMP!BR7CON!<ININT-INIT&3000*4>!<ININT-INIT&777/2>>
INWAT2: BRWRTE  IBUS,INCON              ;READ INPUT CONTROL CSR
        MICPC=MICPC+1
        <MOVE!WRTEBR!IBUS!<INCON>>
        BRSHFT
        MICPC=MICPC+1
        <MOVE!SHFTBR!WRTEBR!SELB>
        BR4     NIDLE6
        MICPC=MICPC+1
        <JUMP!BR4CON!<NIDLE6-INIT&3000*4>!<NIDLE6-INIT&777/2>>
INSRV:  SPBR    IBUS,INCON,SPO          ;READ THE INPUT CONTROL CSR
        MICPC=MICPC+1
        <MOVE!SPBRX!IBUS!INCON!SPO>
        BR1     30$                     ;--SENSE OR BASE
        MICPC=MICPC+1
        <JUMP!BR1CON!<30$-INIT&3000*4>!<30$-INIT&777/2>>
        PSTATE  CTLSRV                  ;ASSUME CNTL I
        MEM     IMM,<<CTLSRV-INIT&777/2>>
        MICPC=MICPC+1
        <MOVE!WRMEM!IMM!<<CTLSRV-INIT&777/2>>>
        BR0     20$                     ;CNTL I
        MICPC=MICPC+1
        <JUMP!BR0CON!<20$-INIT&3000*4>!<20$-INIT&777/2>>
        BRSHFT                          ;MUST BE BA/CC
        MICPC=MICPC+1
        <MOVE!SHFTBR!WRTEBR!SELB>
        PSTATE  RBASRV                  ;ASSUME IN
        MEM     IMM,<<RBASRV-INIT&777/2>>
        MICPC=MICPC+1
        <MOVE!WRMEM!IMM!<<RBASRV-INIT&777/2>>>
        BR1     20$                     ;REC BA/CC
        MICPC=MICPC+1
        <JUMP!BR1CON!<20$-INIT&3000*4>!<20$-INIT&777/2>>
        PSTATE  TBASRV                  ;XMIT BA/CC
        MEM     IMM,<<TBASRV-INIT&777/2>>
        MICPC=MICPC+1
        <MOVE!WRMEM!IMM!<<TBASRV-INIT&777/2>>>
20$:    BRWRTE  BR,SELA!SP1             ;INIT MODE
        MICPC=MICPC+1
        <MOVE!WRTEBR!BR!<SELA!SP1>>
        BR0     PROCER                  ;IF INIT MODE--ERROR
        MICPC=MICPC+1
```

```
(1)  030134  102133
660  030136
(1)          000131
(1)  030136  100447
661  030140
(1)          000132
(1)  030140  102143
662  030142
(1)  030142
(2)          000133
(2)  030142  002461
663  030144
(1)          000134
(1)  030144  000500
664  030146
(1)          000135
(1)  030146  061260
665  030150
(1)          000136
(1)  030150  010177
666  030152
(1)          000137
(1)  030152  016402
667  030154
(1)          000140
(1)  030154  002400
668  030156
(1)          000141
(1)  030156  042233
669  030160
(1)          000142
(1)  030160  114511
670  030162
(1)          000143
(1)  030162  060721
671  030164
(1)          000144
(1)  030164  102133
```

NIDLE2---NO CSR ACTIVITY STATE

```
         <JUMP!BROCON!<PROCER-INIT&3000*4>!<PROCER-INIT&777/2>>
         ALWAYS  IDLE
         MICPC=MICPC+1
         <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
30$:     BRO     INSRV1                          ;IF BASE---PROCESS
         MICPC=MICPC+1
         <JUMP!BROCON!<INSRV1-INIT&3000*4>!<INSRV1-INIT&777/2>>
PROCER:  PSTATE  NIDLE2                          ;RESET PORT STATUS
         MEM     IMM,<<NIDLE2-INIT&777/2>>
         MICPC=MICPC+1
         <MOVE!WRMEM!IMM!<<NIDLE2-INIT&777/2>>>
         BRWRTE  IMM,100                         ;CLEAR INPUT CONTROL CSR
         MICPC=MICPC+1
         <MOVE!WRTEBR!IMM!<100>>
         OUT     BR,AANDB!OINCON                 ;;
         MICPC=MICPC+1
         <MOVE!WROUTX!BR!<AANDB!OINCON>>
         LDMA    IMM,<<RTHRS+3>>                 ;ADDRESS ERROR LINK
         MICPC=MICPC+1
         <MOVE!LDMAR!IMM!<<RTHRS+3>&377>>
         MEMINC  IMM,2
         MICPC=MICPC+1
         <MOVE!WRMEM!INCMAR!IMM!<2>>
         MEM     IMM,0
         MICPC=MICPC+1
         <MOVE!WRMEM!IMM!<0>>
         OUTPUT  MEMX,SELB!OMODEM                ;CLEAR DATA TERMINAL READY
         MICPC=MICPC+1
         <MOVE!WROUT!MEMX!<SELB!OMODEM>>
         ALWAYS  RCEXX                           ;POST THE ERROR - FATAL
         MICPC=MICPC+1
         <JUMP!ALCOND!<RCEXX-INIT&3000*4>!<RCEXX-INIT&777/2>>
INSRV1:  BRWRTE  BR,AXORB!SP1                    ;INIT MODE?
         MICPC=MICPC+1
         <MOVE!WRTEBR!BR!<AXORB!SP1>>
         BRO     PROCER
         MICPC=MICPC+1
         <JUMP!BROCON!<PROCER-INIT&3000*4>!<PROCER-INIT&777/2>>
```

```
                        .SBTTL  BASSRV---- BASE SERVICE ROUTINE
673
674  030166             BASSRV: PSTATE  NIDLE2
(1)  030166                     MEM     IMM,<<NIDLE2-INIT&777/2>>
                                MICPC=MICPC+1
(2)             000145          <MOVE!WRMEM!IMM!<<NIDLE2-INIT&777/2>>>
(2)  030166     002461  
675  030170                     LDMA    IMM,BASE                        ;CLEAR TO MAR SO IT POINTS TO BASE POINT
(1)             000146          MICPC=MICPC+1
(1)  030170     010017          <MOVE!LDMAR!IMM!<BASE&377>>
676  030172                     MEMINC  IBUS,PORT1                      ;READ CSR4
(1)             000147          MICPC=MICPC+1
(1)  030172     136500          <MOVE!WRMEM!INCMAR!IBUS!<PORT1>>
677  030174                     MEMINC  IBUS,PORT2                      ;READ CSR5
(1)             000150          MICPC=MICPC+1
(1)  030174     136520          <MOVE!WRMEM!INCMAR!IBUS!<PORT2>>
678  030176                     MEM     IBUS,PORT4
(1)             000151          MICPC=MICPC+1
(1)  030176     122560          <MOVE!WRMEM!IBUS!<PORT4>>
679  030200                     SP      IBUS,INCON,SPO                  ;READ INPUT CONTROL CSR
(1)             000152          MICPC=MICPC+1
(1)  030200     123000          <MOVE!SPX!IBUS!INCON!SPO>
680  030202                     BRWRTE  IMM,100                         ;CLEAR THE BR
(1)             000153          MICPC=MICPC+1
(1)  030202     000500          <MOVE!WRTEBR!IMM!<100>>
681  030204                     OUT     BR,<AANDB!OINCON>               ;CLEAR THE INCONTROL CSR
(1)             000154          MICPC=MICPC+1
(1)  030204     061260          <MOVE!WROUTX!BR!<AANDB!OINCON>>
682  030206                     OUTPUT  IMM,<120!OMODEM>                ;MASK FOR HDX AND DTR
(1)             000155          MICPC=MICPC+1
(1)  030206     002133          <MOVE!WROUT!IMM!<120!OMODEM>>
683  030210                     BRWRTE  MEMX,SELB                       ;READ SEL6
(1)             000156          MICPC=MICPC+1
(1)  030210     040620          <MOVE!WRTEBR!MEMX!<SELB>>
684  030212                     BR4     RESUME                          ;IF SET RESUME
(1)             000157          MICPC=MICPC+1
(1)  030212     103167          <JUMP!BR4CON!<RESUME-INIT&3000*4>!<RESUME-INIT&777/2>>
685  030214                     LDMA    IMM,T                           ;LOAD ADDRESS OF TYPE FIELD
(1)             000160          MICPC=MICPC+1
(1)  030214     010151          <MOVE!LDMAR!IMM!<T&377>>
686  030216                     MEMINC  IMM,6                           ;WRITE START TYPE TO MEMORY
(1)             000161          MICPC=MICPC+1
(1)  030216     016406          <MOVE!WRMEM!INCMAR!IMM!<6>>
687  030220                     MEM     IMM,300                         ;WRITE SELECT AND FINAL TO MEMORY
(1)             000162          MICPC=MICPC+1
(1)  030220     002700          <MOVE!WRMEM!IMM!<300>>
688  030222                     SP      BR,DECA,SP1     ;TURN OFF INIT MODE
(1)             000163          MICPC=MICPC+1
(1)  030222     063161          <MOVE!SPX!BR!DECA!SP1>
689  030224             BS1:    BRWRTE  IMM,241                         ;SET OK TO SEND,STARTMODE AND UNNUM PENDING
(1)             000164          MICPC=MICPC+1
(1)  030224     000641          <MOVE!WRTEBR!IMM!<241>>
690  030226             BS0:    SP      IMM,374,SP14    ;RESET REP COUNTER
(1)             000165          MICPC=MICPC+1
(1)  030226     003374          <MOVE!SPX!IMM!374!SP14>
691  030230                     ALWAYS  SA3
(1)             000166          MICPC=MICPC+1
(1)  030230     110743          <JUMP!ALCOND!<SA3-INIT&3000*4>!<SA3-INIT&777/2>>
```

```
692  030232
(1)            000167
(1)  030232    003004
693  030234
(1)            000170
(1)  030234    063070
694
695  030236
(1)            000171
(1)  030236    010017
696  030240
(1)            000172
(1)  030240    000745
697  030242
(1)            000173
(1)  030242    110463
698  030244
(1)            000174
(1)  030244    010154
699  030246
(1)            000175
(1)  030246    057310
700  030250
(1)            000176
(1)  030250    057231
701  030252
(1)            000177
(1)  030252    057235
702  030254
(1)            000200
(1)  030254    043237
703  030256
(1)            000201
(1)  030256    043232
704  030260
(1)            000202
(1)  030260    063170
705
706  030262
(1)            000203
(1)  030262    063161
707  030264
(1)            000204
(1)  030264    000606
708  030266
(1)            000205
(1)  030266    114434
709  030270
(1)  030270
(2)            000206
(2)  030270    002514
710  030272
(1)            000207
(1)  030272    000415
711  030274
(1)            000210
```

```
RESUME: SP      IMM,SP4,4               ;SET UP SP4 FOR COUNTING NPRS
        MICPC=MICPC+1
        <MOVE!SPX!IMM!SP4!4>
        SP      BR,INCA,SP10            ;SET UNNUMB MESSAGE PENDING TO
        MICPC=MICPC+1
        <MOVE!SPX!BR!INCA!SP10>
                                        ;TRICK TRANSMITTER CODE
        LDMA    IMM,BASE                ;ADDRESS BASE TABLE ADDRESS
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<BASE&377>>
        STATE   FUDGE                   ;SET TMTR STATE TO ENTER TABLE UPDATE
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<FUDGE-INIT&777/2>>
        ALWAYS  TBO                     ;GO SET UP MXT BITS AND ADRESS OF BASE FOR NPRS
        MICPC=MICPC+1
        <JUMP!ALCOND!<TBO-INIT&3000*4>!<TBO-INIT&777/2>>
BS2:    LDMA    IMM,IMG10
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<IMG10&377>>
        SP      MEMX!INCMAR,AORB,SP10           ;RESTORE BIT 1 OF SP10
        MICPC=MICPC+1
        <MOVE!SPX!MEMX!INCMAR!AORB!SP10>
        SP      MEMX!INCMAR,SELB,SP11           ;RESTORE SP11
        MICPC=MICPC+1
        <MOVE!SPX!MEMX!INCMAR!SELB!SP11>
        SP      MEMX!INCMAR,SELB,SP15           ;RESTORE SP15
        MICPC=MICPC+1
        <MOVE!SPX!MEMX!INCMAR!SELB!SP15>
        SP      MEMX,SELB,SP17          ;RESTORE        SP17
        MICPC=MICPC+1
        <MOVE!SPX!MEMX!SELB!SP17>
        SP      MEMX,SELB,SP12          ;RESTORE        SP12
        MICPC=MICPC+1
        <MOVE!SPX!MEMX!SELB!SP12>
        SP      BR,DECA,SP10            ;TURN OFF UNNUM MESSAGE PENDING AND
        MICPC=MICPC+1
        <MOVE!SPX!BR!DECA!SP10>
                                        ;ZERO THE BRG
        SP      BR,DECA,SP1             ;CLEAR INIT MODE
        MICPC=MICPC+1
        <MOVE!SPX!BR!DECA!SP1>
        BRWRTE  IMM,206
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<206>>
        ALWAYS  NAK1
        MICPC=MICPC+1
        <JUMP!ALCOND!<NAK1-INIT&3000*4>!<NAK1-INIT&777/2>>
ININT:  PSTATE  INWAT2
        MEM     IMM,<<INWAT2-INIT&777/2>>
        MICPC=MICPC+1
        <MOVE!WRMEM!IMM!<<INWAT2-INIT&777/2>>>
        BRWRTE  IMM,15                  ;MASK FOR TURN OFF ALL BUT EXT MEM BITS + NXM
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<15>>
        SP      IBUS,UBBR,SP0           ;READ BR CONTROL REG
        MICPC=MICPC+1
```

```
(1)  030274  123220          <MOVE!SPX!IBUS!UBBR!SPO>
712  030276                  SP      BR,AANDB,SPO            ;MASK OFF VECTOR TO X04
(1)          000211          MICPC=MICPC+1
(1)  030276  063260          <MOVE!SPX!BR!AANDB!SPO>
713  030300                  BRWRTE  IMM,200                 ;MASK FOR INTERRUPT
(1)          000212          MICPC=MICPC+1
(1)  030300  000600          <MOVE!WRTEBR!IMM!<200>>
714  030302                  ALWAYS RB4
(1)          000213          MICPC=MICPC+1
(1)  030302  104457          <JUMP!ALCOND!<RB4-INIT&3000*4>!<RB4-INIT&777/2>>
```

```
                        OUTINT---SET UP OUTPUT INTERRUPT [RDYO]

                                .SBTTL  OUTINT---SET UP OUTPUT INTERRUPT [RDYO]
                        OUTINT:
                                PSTATE  OUTWAIT         ;PORT STATUS TO WAITING FOR OUT
                                MEM     IMM,<<OUTWAIT-INIT&777/2>>
                                MICPC=MICPC+1
                                <MOVE!WRMEM!IMM!<<OUTWAIT-INIT&777/2>>>
                                                        ;COMPLETION
                                LDMA    IMM,NXTINT              ;ADDRESS OF NEXT INTERRUPT POINTER
                                MICPC=MICPC+1
                                <MOVE!LDMAR!IMM!<NXTINT&377>>
                                LDMA    MEMX,SELB               ;NEXT INTERRUPT
                                MICPC=MICPC+1
                                <MOVE!LDMAR!MEMX!<SELB>>
                                SP      IBUS,OCON,SPO           ;READ THE OUTPUT CONTROL CSR
                                MICPC=MICPC+1
                                <MOVE!SPX!IBUS!OCON!SPO>
                                OUT     <MEMX!INCMAR>,<AORB!OOCON>      ;WRITE THE OUT CONTROL CSR
                                MICPC=MICPC+1
                                <MOVE!WROUTX!MEMX!INCMAR!<AORB!OOCON>>
                                LDMA    MEMX,SELB                       ;ADDRESS LINK
                                MICPC=MICPC+1
                                <MOVE!LDMAR!MEMX!<SELB>>
                                BRWRTE  <BR!INCMAR>,<AA!SPO>            ;KICK PAST LINK STATUS BYTE
                                MICPC=MICPC+1
                                <MOVE!WRTEBR!BR!INCMAR!<AA!SPO>>
                                                                        ;SHIFT CSRO IMAGE LEFT
                                                        ;***DO NOT CHANGE BR UNTIL BR7***
                                OUT     <MEMX!INCMAR>,<SELB!OPORT1>     ;WRITE LOW BYTE OF BA TO CSR
                                MICPC=MICPC+1
                                <MOVE!WROUTX!MEMX!INCMAR!<SELB!OPORT1>>
                                OUT     <MEMX!INCMAR>,<SELB!OPORT2>     ;WRITE HIGH BYTE OF BA TO CSR
                                MICPC=MICPC+1
                                <MOVE!WROUTX!MEMX!INCMAR!<SELB!OPORT2>>
                                OUT     <MEMX!INCMAR>,<SELB!OPORT4>     ;WRITE HIGH BYTE OF COUNT TO CSR
                                MICPC=MICPC+1
                                <MOVE!WROUTX!MEMX!INCMAR!<SELB!OPORT4>>
                                OUT     <MEMX!INCMAR>,<SELB!OPORT3>     ;WRITE THE LOW BYTE OF COUNT
                                MICPC=MICPC+1
                                <MOVE!WROUTX!MEMX!INCMAR!<SELB!OPORT3>>
                                                                ;***HERE IS BR7***
                                BR7     PE1                     ;INTERRUPT ENABLE IS SET
                                MICPC=MICPC+1
                                <JUMP!BR7CON!<PE1-INIT&3000*4>!<PE1-INIT&777/2>>
                                                                ;GENERATE AN INTERRUPT
                        PINT2:
                                LDMA    IMM,NXTINT              ;ADDRESS NEXT INTERRUPT QUEUE
                                MICPC=MICPC+1
                                <MOVE!LDMAR!IMM!<NXTINT&377>>
                                SP      MEMX,SELB,SPO           ;COPY ADDRESS FOR NEXT INT TO SPO
                                MICPC=MICPC+1
                                <MOVE!SPX!MEMX!SELB!SPO>
                                MEM     IMM,INTSTK              ;ASSUME WRAP AROUND CASE
                                MICPC=MICPC+1
                                <MOVE!WRMEM!IMM!<INTSTK>>
                                BRWRTE  IMM,<<MMEND-2>>         ;ADDRESS OF LAST INT IN STACK
                                MICPC=MICPC+1
```

```
    (1)  030342  000776                <MOVE!WRTEBR!IMM!<<MMEND-2>>>
751  030344                           CMP     BR,SPO                  ;SHOULD WE WRAP
    (1)          000234              MICPC=MICPC+1
    (1)  030344  060360              <SUBTC!BR!SPO>
752  030346                           Z       5$                      ;YES--BRANCH
    (1)          000235              MICPC=MICPC+1
    (1)  030346  101640              <JUMP!ZCOND!<5$-INIT&3000*4>!<5$-INIT&777/2>>
753  030350                           BRWRTE  IMM,2                   ;OFFSET FOR NEXT POINTER
    (1)          000236              MICPC=MICPC+1
    (1)  030350  000402              <MOVE!WRTEBR!IMM!<2>>
754  030352                           MEM     BR,ADD!SPO              ;UPDATE POINTER
    (1)          000237              MICPC=MICPC+1
    (1)  030352  062400              <MOVE!WRMEM!BR!<ADD!SPO>>
755  030354                  5$:      SP      MEMX,SELB,SPO           ;COPY POINTER TO SPO
    (1)          000240              MICPC=MICPC+1
    (1)  030354  043220              <MOVE!SPX!MEMX!SELB!SPO>
756  030356                           LDMA    IMM,NXTSP               ;PICK UP START OF IN QUEUE
    (1)          000241              MICPC=MICPC+1
    (1)  030356  010241              <MOVE!LDMAR!IMM!<NXTSP&377>>
757  030360                           CMP     MEMX,SPO                ;COMPARE TO END
    (1)          000242              MICPC=MICPC+1
    (1)  030360  040360              <SUBTC!MEMX!SPO>
758  030362                           Z       10$                     ;IF EQUAL--CLEAR INT PENDING
    (1)          000243              MICPC=MICPC+1
    (1)  030362  101645              <JUMP!ZCOND!<10$-INIT&3000*4>!<10$-INIT&777/2>>
759  030364                           ALWAYS  IDLE
    (1)          000244              MICPC=MICPC+1
    (1)  030364  100447              <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
760  030366                  10$:     BRWRTE  IMM,357                 ;MASK TO CLEAR INT PENDING
    (1)          000245              MICPC=MICPC+1
    (1)  030366  000757              <MOVE!WRTEBR!IMM!<357>>
761  030370                  CLRIDL:  SP      BR,AANDB,SP1
    (1)          000246              MICPC=MICPC+1
    (1)  030370  063261              <MOVE!SPX!BR!AANDB!SP1>
762  030372                           ALWAYS  IDLE
    (1)          000247              MICPC=MICPC+1
    (1)  030372  100447              <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
```

```
                                .SBTTL  OUTWAI--WAIT FOR RDYO TO GO AWAY
030374                  OUTWAI: SPBR    IBUS,OCON,SPO           ;READ OUTPUT CONTROL CSR
(1)     000250                  MICPC=MICPC+1
(1)     030374  123440          <MOVE!SPBRX!IBUS!OCON!SPO>
030376                          BR7     NIDLE6                  ;RDYO SET --GET OUT
(1)     000251                  MICPC=MICPC+1
(1)     030376  103466          <JUMP!BR7CON!<NIDLE6-INIT&3000*4>!<NIDLE6-INIT&777/2>>
030400                          BRWRTE  IMM,100         ;CLEAR CONTROL BITS
(1)     000252                  MICPC=MICPC+1
(1)     030400  000500          <MOVE!WRTEBR!IMM!<100>>
030402                          OUT     BR,OOCON!AANDB
(1)     000253                  MICPC=MICPC+1
(1)     030402  061262          <MOVE!WROUTX!BR!<OOCON!AANDB>>
030404                          ALWAYS  INS13
(1)     000254                  MICPC=MICPC+1
(1)     030404  100667          <JUMP!ALCOND!<INS13-INIT&3000*4>!<INS13-INIT&777/2>>
                                .SBTTL  CTLSRV--CNTL I SERVICE
030406                  CTLSRV: SPBR    IBUS,PORT4,SPO                  ;TO SPO
(1)     000255                  MICPC=MICPC+1
(1)     030406  123560          <MOVE!SPBRX!IBUS!PORT4!SPO>
030410                          BRSHFT
(1)     000256                  MICPC=MICPC+1
(1)     030410  001620          <MOVE!SHFTBR!WRTEBR!SELB>
030412                          BR1     HDSEL                   ;IF SET IS HALF DUPLEX
(1)     000257                  MICPC=MICPC+1
(1)     030412  102677          <JUMP!BR1CON!<HDSEL-INIT&3000*4>!<HDSEL-INIT&777/2>>
030414                          OUTPUT  IMM,<100!OMODEM>                ;MASK DTR, TURN OFF HDX
(1)     000260                  MICPC=MICPC+1
(1)     030414  002113          <MOVE!WROUT!IMM!<100!OMODEM>>
030416                  INS11:  BRWRTE  DP,<SELA!SPO>                   ;RESTORE THE CNTL WORD
(1)     000261                  MICPC=MICPC+1
(1)     030416  060600          <MOVE!WRTEBR!DP!<SELA!SPO>>
030420                          BR0     CBOOT                   ;IF SET IS BOOT
(1)     000262                  MICPC=MICPC+1
(1)     030420  116351          <JUMP!BR0CON!<CBOOT-INIT&3000*4>!<CBOOT-INIT&777/2>>
030422                  INS12:  SP      IBUS,INCON,SPO          ;READ THE INPUT CONTROL CSR
(1)     000263                  MICPC=MICPC+1
(1)     030422  123000          <MOVE!SPX!IBUS!INCON!SPO>
030424                          BRWRTE  IMM,100                 ;ZERO THE BR REGISTER EXCEPT INT ENABLE
(1)     000264                  MICPC=MICPC+1
(1)     030424  000500          <MOVE!WRTEBR!IMM!<100>>
030426                          OUT     BR,<AANDB!OINCON>               ;CLEAR IN CONTROL CSR
(1)     000265                  MICPC=MICPC+1
(1)     030426  061260          <MOVE!WROUTX!BR!<AANDB!OINCON>>
030430                          LDMA    IMM,PRTST                       ;ADDRESS PORT STATE
(1)     000266                  MICPC=MICPC+1
(1)     030430  010211          <MOVE!LDMAR!IMM!<PRTST&377>>
030432                  INS13:  PSTATE  NIDLE2
(1)     030432                  MEM     IMM,<<NIDLE2-INIT&777/2>>
(2)     000267                  MICPC=MICPC+1
(2)     030432  002461          <MOVE!WRMEM!IMM!<<NIDLE2-INIT&777/2>>>
030434                          ALWAYS  IDLE
(1)     000270                  MICPC=MICPC+1
(1)     030434  100447          <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
                                ;
030436                  TABMXT: SP      IBUS,UBBR,SPO           ;READ BR CONTROL
```

```
(1)             000271
(1)  030436     123220          MICPC=MICPC+1
                                <MOVE!SPX!IBUS!UBBR!SPO>
790  030440                     BRWRTE  IMM,115                 ;MASK TO CLEAR BR REQ
(1)             000272          MICPC=MICPC+1
(1)  030440     000515          <MOVE!WRTEBR!IMM!<115>>
791  030442                     SP      BR,AANDB,SPO
(1)             000273          MICPC=MICPC+1
(1)  030442     063260          <MOVE!SPX!BR!AANDB!SPO>
792  030444                     BRWRTE  IMM,4                   ;INCREMENT MXT
(1)             000274          MICPC=MICPC+1
(1)  030444     000404          <MOVE!WRTEBR!IMM!<4>>
793  030446                     OUT     BR,ADD!OBR
(1)             000275          MICPC=MICPC+1
(1)  030446     061011          <MOVE!WROUTX!BR!<ADD!OBR>>
794  030450                     ALWAYS  ECX
(1)             000276          MICPC=MICPC+1
(1)  030450     114734          <JUMP!ALCOND!<ECX-INIT&3000*4>!<ECX-INIT&777/2>>
795  030452     
(1)             000277  HDSEL:  BRWRTE  IMM,100                 ;HD MASK TO BR
(1)  030452     000500          MICPC=MICPC+1
796  030454                     <MOVE!WRTEBR!IMM!<100>>
(1)             000300          SP      BR,AORB,SP10            ;UPDATE PORT STATUS WORD
(1)  030454     063310          MICPC=MICPC+1
797  030456                     <MOVE!SPX!BR!AORB!SP10>
(1)             000301          ALWAYS  INS11
(1)  030456     100661          MICPC=MICPC+1
                                <JUMP!ALCOND!<INS11-INIT&3000*4>!<INS11-INIT&777/2>>
```

```
 799
 800  030460                 .SBTTL  TBASRV--TRANSMITTER BUFFER ADDRESS SERVICE
(1)           000302 TBASRV: LDMA    IMM,ETC                    ;GET POINTER TO END OF TMT CHAIN
(1)    030460 010070         MICPC=MICPC+1
                             <MOVE!LDMAR!IMM!<ETC&377>>
 801  030462                 LDMA    MEMX,<SELB!SPX!SPO>        ;FIND THE LINK
(1)           000303         MICPC=MICPC+1
(1)    030462 053220         <MOVE!LDMAR!MEMX!<SELB!SPX!SPO>>
 802  030464                 MEMINC  IMM,1                      ;BUFFER ASSIGNED IN IN LINK FLAGS
(1)           000304         MICPC=MICPC+1
(1)    030464 016401         <MOVE!WRMEM!INCMAR!IMM!<1>>
 803  030466                 BRWRTE  <IMM!INCMAR>,TML8                  ;POINT PAST NUMBER FIELD
(1)           000305         MICPC=MICPC+1
(1)    030466 014543         <MOVE!WRTEBR!IMM!INCMAR!<TML8>>
 804                                                            ;SET BR FOR ADDITION TO SPO
 805  030470                 MEMINC  IBUS,PORT1
(1)           000306         MICPC=MICPC+1
(1)    030470 136500         <MOVE!WRMEM!INCMAR!IBUS!<PORT1>>
 806  030472                 MEMINC  IBUS,PORT2
(1)           000307         MICPC=MICPC+1
(1)    030472 136520         <MOVE!WRMEM!INCMAR!IBUS!<PORT2>>
 807  030474                 MEMINC  IBUS,PORT4
(1)           000310         MICPC=MICPC+1
(1)    030474 136560         <MOVE!WRMEM!INCMAR!IBUS!<PORT4>>
 808  030476                 MEMINC  IBUS,PORT3
(1)           000311         MICPC=MICPC+1
(1)    030476 136540         <MOVE!WRMEM!INCMAR!IBUS!<PORT3>>
 809  030500                 LDMA    IMM,ETC
(1)           000312         MICPC=MICPC+1
(1)    030500 010070         <MOVE!LDMAR!IMM!<ETC&377>>
 810  030502                 MEM     IMM,TML1                   ;ASSUME QUEUE WRAP AROUND
(1)           000313         MICPC=MICPC+1
(1)    030502 002471         <MOVE!WRMEM!IMM!<TML1>>
 811  030504                 CMP     BR,SPO                     ;END OF CHAIN?
(1)           000314         MICPC=MICPC+1
(1)    030504 060360         <SUBTC!BR!SPO>
 812  030506                 Z       10$                        ;IF YES--BRANCH
(1)           000315         MICPC=MICPC+1
(1)    030506 101720         <JUMP!ZCOND!<10$-INIT&3000*4>!<10$-INIT&777/2>>
 813  030510                 BRWRTE  IMM,6                      ;QUEUE ENTRY LENGTH
(1)           000316         MICPC=MICPC+1
(1)    030510 000406         <MOVE!WRTEBR!IMM!<6>>
 814  030512                 MEM     BR,ADD!SPO                 ;UPDATE THE END POINTER IN MEMORY
(1)           000317         MICPC=MICPC+1
(1)    030512 062400         <MOVE!WRMEM!BR!<ADD!SPO>>
 815  030514          10$:   BRWRTE  IMM,2                      ;NUMBERED MSG PENDING MASK
(1)           000320         MICPC=MICPC+1
(1)    030514 000402         <MOVE!WRTEBR!IMM!<2>>
 816  030516                 SP      BR,AORB,SP10               ;UPDATE LINE STATUS
(1)           000321         MICPC=MICPC+1
(1)    030516 063310         <MOVE!SPX!BR!AORB!SP10>
 817  030520                 ALWAYS  INS12
(1)           000322         MICPC=MICPC+1
(1)    030520 100663         <JUMP!ALCOND!<INS12-INIT&3000*4>!<INS12-INIT&777/2>>
 818                          .SBTTL  RBASRV--RECEIVE BUFFER ADDRESS SERVICE
 819  030522          RBASRV: LDMA    IMM,ERC                    ;ADDRES END OF RECEIVE CHAIN
(1)           000323         MICPC=MICPC+1
```

```
(1)  030522  010023                    <MOVE!LDMAR!IMM!<ERC&377>>
820  030524                           LDMA    MEMX,<SELB!SPX!SPO>          ;GET THE POINTER TO LINK
(1)            000324                MICPC=MICPC+1
(1)  030524  053220                    <MOVE!LDMAR!MEMX!<SELB!SPX!SPO>>
821  030526                           MEMINC  IMM,1
(1)            000325                MICPC=MICPC+1
(1)  030526  016401                    <MOVE!WRMEM!INCMAR!IMM!<1>>
822  030530                           MEMINC  IBUS,PORT1
(1)            000326                MICPC=MICPC+1
(1)  030530  136500                    <MOVE!WRMEM!INCMAR!IBUS!<PORT1>>
823  030532                           MEMINC  IBUS,PORT2
(1)            000327                MICPC=MICPC+1
(1)  030532  136520                    <MOVE!WRMEM!INCMAR!IBUS!<PORT2>>
824  030534                           MEMINC  IBUS,PORT4
(1)            000330                MICPC=MICPC+1
(1)  030534  136560                    <MOVE!WRMEM!INCMAR!IBUS!<PORT4>>
825  030536                           MEMINC  IBUS,PORT3
(1)            000331                MICPC=MICPC+1
(1)  030536  136540                    <MOVE!WRMEM!INCMAR!IBUS!<PORT3>>
826                                   ;;;;NOTE INVERTED ORDER OF PORT 3 AND PORT4
827  030540                           LDMA    IMM,ERC
(1)            000332                MICPC=MICPC+1
(1)  030540  010023                    <MOVE!LDMAR!IMM!<ERC&377>>
828  030542                           MEM     IMM,RCL1                     ;ASSUME WRAP AROUND CASE
(1)            000333                MICPC=MICPC+1
(1)  030542  002424                    <MOVE!WRMEM!IMM!<RCL1>>
829  030544                           BRWRTE  IMM,RCL7                     ;GET ADDRESS OF END OF CAHIN AREA
(1)            000334                MICPC=MICPC+1
(1)  030544  000462                    <MOVE!WRTEBR!IMM!<RCL7>>
830  030546                           CMP     BR,SPO                               ;CHECK FOR END
(1)            000335                MICPC=MICPC+1
(1)  030546  060360                    <SUBTC!BR!SPO>
831  030550                           Z       INS12                                ;IF EQUAL BRANCH
(1)            000336                MICPC=MICPC+1
(1)  030550  101663                    <JUMP!ZCOND!<INS12-INIT&3000*4>!<INS12-INIT&777/2>>
832  030552                           BRWRTE  IMM,5                        ;CALCULATE ADDRESS OF NEXT LINK
(1)            000337                MICPC=MICPC+1
(1)  030552  000405                    <MOVE!WRTEBR!IMM!<5>>
833  030554                           MEM     BR,ADD!SPO                   ;..
(1)            000340                MICPC=MICPC+1
(1)  030554  062400                    <MOVE!WRMEM!BR!<ADD!SPO>>
834  030556                           ALWAYS  INS12                                ;EXIT
(1)            000341                MICPC=MICPC+1
(1)  030556  100663                    <JUMP!ALCOND!<INS12-INIT&3000*4>!<INS12-INIT&777/2>>
835  030560                  RA1:     BRWRTE  IMM,357                      ;MASK TO CLEAR CLR ACTIVE
(1)            000342                MICPC=MICPC+1
(1)  030560  000757                    <MOVE!WRTEBR!IMM!<357>>
836  030562                           SPBR    BR,AANDB,SP10                ;CLEAR BIT IN LINE STATUS WORD
(1)            000343                MICPC=MICPC+1
(1)  030562  063670                    <MOVE!SPBRX!BR!AANDB!SP10>
837  030564                           BRSHFT
(1)            000344                MICPC=MICPC+1
(1)  030564  001620                    <MOVE!SHFTBR!WRTEBR!SELB>
838  030566                           BR4     FLUSH
(1)            000345                MICPC=MICPC+1
(1)  030566  107015                    <JUMP!BR4CON!<FLUSH-INIT&3000*4>!<FLUSH-INIT&777/2>>
```

```
 839  030570
(1)           000346
(1)   030570  000400
 840  030572
(1)           000347
(1)   030572  063233
 841  030574
(1)           000350
(1)   030574  000427
 842  030576
(1)           000351
(1)   030576  104425
 843
```

```
RA3:    BRWRTE  IMM,0                          ;CLEAR BR
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<0>>
        SP      BR,SELB,SP13                   ;SET NUMB MESSAGE TYPE IN SP13
        MICPC=MICPC+1
        <MOVE!SPX!BR!SELB!SP13>
        STATE   RCVB                           ;CHANGE RECEIVE STATE POINTER TO STATE B
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<RCVB-INIT&777/2>>
        ALWAYS  REXIT
        MICPC=MICPC+1
        <JUMP!ALCOND!<REXIT-INIT&3000*4>!<REXIT-INIT&777/2>>
        ;
```

RBASRV--RECEIVE BUFFER ADDRESS SERVICE

```
 845  030600
(1)             000352
(1)   030600    000700
 846  030602
(1)             000353
(1)   030602    123220
 847  030604
(1)             000354
(1)   030604    061311
 852  030606
(1)             000355
(1)   030606    100630
 854
 855
 856  030610
(1)             000356
(1)   030610    120600
 857  030612
(1)             000357
(1)   030612    102047
 858  030614
(1)             000360
(1)   030614    114725
 859
 860
 861  030616
(1)             000361
(1)   030616    000404
 862  030620
(1)             000362
(1)   030620    114657
 863  030622
(1)             000363
(1)   030622    000720
 864  030624
(1)             000364
(1)   030624    110554
 865  030626
(1)             000365
(1)   030626    000727
 866  030630
(1)             000366
(1)   030630    063270
 867  030632
(1)             000367
(1)   030632    104504
 872
```

```
PE1:    BRWRTE  IMM,300                 ;MASK FOR INTERRUPT AND VECTOR
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<300>>
        SP      IBUS,UBBR,SPO           ;READ BR CONTROL REG
        MICPC=MICPC+1
        <MOVE!SPX!IBUS!UBBR!SPO>
        OUT     BR,<AORB!OBR>           ;INTERRUPT
        MICPC=MICPC+1
        <MOVE!WROUTX!BR!<AORB!OBR>>
        ALWAYS  PINT2
        MICPC=MICPC+1
        <JUMP!ALCOND!<PINT2-INIT&3000*4>!<PINT2-INIT&777/2>>
;
;
TBU1:   BRWRTE  IBUS,NPR
        MICPC=MICPC+1
        <MOVE!WRTEBR!IBUS!<NPR>>
        BRO     IDLE
        MICPC=MICPC+1
        <JUMP!BROCON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
        ALWAYS  EC2
        MICPC=MICPC+1
        <JUMP!ALCOND!<EC2-INIT&3000*4>!<EC2-INIT&777/2>>
;
OVRRUN: BRWRTE  IMM,4
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<4>>
        ALWAYS  NTRSO
        MICPC=MICPC+1
        <JUMP!ALCOND!<NTRSO-INIT&3000*4>!<NTRSO-INIT&777/2>>
HEH1:   STATE   TEOM
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<TEOM-INIT&777/2>>
        ALWAYS  XEXIT
        MICPC=MICPC+1
        <JUMP!ALCOND!<XEXIT-INIT&3000*4>!<XEXIT-INIT&777/2>>
ACK1:   BRWRTE  IMM,327
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<327>>
        SP      BR,AANDB,SP10
        MICPC=MICPC+1
        <MOVE!SPX!BR!AANDB!SP10>
        ALWAYS  RD5
        MICPC=MICPC+1
        <JUMP!ALCOND!<RD5-INIT&3000*4>!<RD5-INIT&777/2>>
```

```
                030654                  .=INIT+1000
                000377                  MICPC=377
                                        .SBTTL  RCVA--ROUTINE TO HANDLE FIRST DDCMP CHARACTER
                                        ;ENTERED FROM IDLE LOOP
                                        ;DETERMINES IF MESSAGE TYPE IS NUMBERED,UNNUMBERED OR BOOT
                                        ;SETS UP APROPRIATE STATES FOR REST OF MESSAGE.
030654                          RCVA:   SP      IBUS,RCVDAT,SPO         ;READ RECEIVE CHARACTER TO SPO
(1)             000400                  MICPC=MICPC+1
(1)     030654  023200                  <MOVE!SPX!IBUS!RCVDAT!SPO>
030656                                  BRWRTE  BR,SELA!SP1             ;READ PORT STATUS WORD
(1)             000401                  MICPC=MICPC+1
(1)     030656  060601                  <MOVE!WRTEBR!BR!<SELA!SP1>>
030660                                  BRO     5$                      ;IF INIT MODE---ONLY BOOT OK
(1)             000402                  MICPC=MICPC+1
(1)     030660  106012                  <JUMP!BROCON!<5$-INIT&3000*4>!<5$-INIT&777/2>>
030662                                  BR7     5$                      ;IF BOOT MODE---ONLY BOOT OK
(1)             000403                  MICPC=MICPC+1
(1)     030662  107412                  <JUMP!BR7CON!<5$-INIT&3000*4>!<5$-INIT&777/2>>
030664                                  BRWRTE  IMM,201                 ;SOH TO BR
(1)             000404                  MICPC=MICPC+1
(1)     030664  000601                  <MOVE!WRTEBR!IMM!<201>>
030666                                  CMP     BR,SPO                  ;COMPARE BR TO SPO
(1)             000405                  MICPC=MICPC+1
(1)     030666  060360                  <SUBTC!BR!SPO>
030670                                  Z       RA1                     ;IF EQUAL-IS NUMBERED MESSAGE
(1)             000406                  MICPC=MICPC+1
(1)     030670  101742                  <JUMP!ZCOND!<RA1-INIT&3000*4>!<RA1-INIT&777/2>>
030672                                  BRWRTE  IMM,5                   ;ENQ TO BR
(1)             000407                  MICPC=MICPC+1
(1)     030672  000405                  <MOVE!WRTEBR!IMM!<5>>
030674                                  CMP     BR,SPO                  ;COMPARE ENQ TO SPO
(1)             000410                  MICPC=MICPC+1
(1)     030674  060360                  <SUBTC!BR!SPO>
030676                                  Z       RA2                     ;IF EQUAL-IS UNNUMBERED MESSAGE
(1)             000411                  MICPC=MICPC+1
(1)     030676  105424                  <JUMP!ZCOND!<RA2-INIT&3000*4>!<RA2-INIT&777/2>>
030700                          5$:     BRWRTE  IMM,220                 ;DLE TO BR
(1)             000412                  MICPC=MICPC+1
(1)     030700  000620                  <MOVE!WRTEBR!IMM!<220>>
030702                                  CMP     BR,SPO                  ;COMPARE DLE TO SPO
(1)             000413                  MICPC=MICPC+1
(1)     030702  060360                  <SUBTC!BR!SPO>
030704                                  Z       BOOT                    ;IF EQUAL IS BOOT
(1)             000414                  MICPC=MICPC+1
(1)     030704  115760                  <JUMP!ZCOND!<BOOT-INIT&3000*4>!<BOOT-INIT&777/2>>
030706                          FLUSH:  OUTPUT  IMM,<200!ORCVCO>        ;FLUSH INPUT SILO
(1)             000415                  MICPC=MICPC+1
(1)     030706  002212                  <MOVE!WROUT!IMM!<200!ORCVCO>>
                                                                        ;(LOW ORDER BITS READ ONLY)
030710                                  RSTATE  RCVA                    ;SET STATE TO RCVA
(1)             000416                  MICPC=MICPC+1
(1)     030710  000400                  <MOVE!WRTEBR!IMM!<RCVA-INIT&777/2>>
(1)             000417                  MICPC=MICPC+1
(1)     030712  063223                  <MOVE!SPX!BR!SELB!SP3>
030714                                  BRWRTE  IMM,357                 ;MASK TO CLEAR--CLEAR ACTIVE
(1)             000420                  MICPC=MICPC+1
```

```
                                RCVA--ROUTINE TO HANDLE FIRST DDCMP CHARACTER

(1)  030714  000757                    <MOVE!WRTEBR!IMM!<357>>
897  030716                    SP      BR,AANDB,SP10           ;IN LINE STATUS WORD
(1)            000421                  MICPC=MICPC+1
(1)  030716  063270                    <MOVE!SPX!BR!AANDB!SP10>
898  030720                    FLUSHA: NOP     BR,INCA,SP10            ;CLEAR   "C" BIT FOR TIMEO
(1)            000422                  MICPC=MICPC+1
(1)  030720  060070                    <BR!INCA!SP10>
899  030722                            ALWAYS  TIMEO                   ;SEE IF WE CAN SEND AN ACK
(1)            000423                  MICPC=MICPC+1
(1)  030722  110665                    <JUMP!ALCOND!<TIMEO-INIT&3000*4>!<TIMEO-INIT&777/2>>
900  030724                    RA2:    STATE   RCVI                    ;CHANGE RECEIVE STATE TO I
(1)            000424                  MICPC=MICPC+1
(1)  030724  000703                    <MOVE!WRTEBR!IMM!<RCVI-INIT&777/2>>
901  030726                    REXIT:  SP      BR,SELB,SP3
(1)            000425                  MICPC=MICPC+1
(1)  030726  063223                    <MOVE!SPX!BR!SELB!SP3>
902  030730                            ALWAYS  IDLE
(1)            000426                  MICPC=MICPC+1
(1)  030730  100447                    <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
```

```
 904
 905
 906
 907
 908  030732
 909  030732                          .SBTTL  RCVB--ROUTINE TO HANDLE FIRST CHARACTER OF COUNT FIELD
                                      ;ENTERED FROM IDLE LOOP
                                      ;STORES COUNT FIELD AND SETS UP RCVC AS NEXT STATE
                                      ;
                      RCVB:
                                      SP      IBUS,RCVDAT,SP4         ;READ CHARACTER TO SP4
                                      MICPC=MICPC+1
                                      <MOVE!SPX!IBUS!RCVDAT!SP4>
                                      LDMA    BR,<SELA!SP15>                  ;LOAD MAR WITH ADDRESS OF CURRENT BA
                                      MICPC=MICPC+1
                                      <MOVE!LDMAR!BR!<SELA!SP15>>
                                      BRWRTE  MEMX!INCMAR,SELB        ;READ THE FLAGS BYTE
                                      MICPC=MICPC+1
                                      <MOVE!WRTEBR!MEMX!INCMAR!<SELB>>
                                      BR0     RB1                     ;BUFFER IS ASSIGNED
                                      MICPC=MICPC+1
                                      <JUMP!BR0CON!<RB1-INIT&3000*4>!<RB1-INIT&777/2>>
                                      BRWRTE  BR,SELA!SP1             ;NO BUFFER ASSIGNED - MAINT MODE?
                                      MICPC=MICPC+1
                                      <MOVE!WRTEBR!BR!<SELA!SP1>>
                                      BR7     RB3                     ;YES
                                      MICPC=MICPC+1
                                      <JUMP!BR7CON!<RB3-INIT&3000*4>!<RB3-INIT&777/2>>
                                      BRWRTE  IMM,310                 ;SET NAK REASON CODE
                                      MICPC=MICPC+1
                                      <MOVE!WRTEBR!IMM!<310>>
                                      LDMA    IMM,NTLS                ;ADDRESS CUMULATIVE COUNTER
                                      MICPC=MICPC+1
                                      <MOVE!LDMAR!IMM!<NTLS&377>>
                                      ALWAYS  RH5                     ;SEND A NAK
                                      MICPC=MICPC+1
                                      <JUMP!ALCOND!<RH5-INIT&3000*4>!<RH5-INIT&777/2>>
                      RB3:            BRWRTE  IMM,2                   ;FLAG FOR NO BUF ASSIGNED IN MAINT MODE
                                      MICPC=MICPC+1
                                      <MOVE!WRTEBR!IMM!<2>>
                                      SP      BR,AORB,SP1             ;SET THE FLAG
                                      MICPC=MICPC+1
                                      <MOVE!SPX!BR!AORB!SP1>
                      RB1:            STATE   RCVC
                                      MICPC=MICPC+1
                                      <MOVE!WRTEBR!IMM!<RCVC-INIT&777/2>>
                      RB0:            SP      BR,SELB,SP3
                                      MICPC=MICPC+1
                                      <MOVE!SPX!BR!SELB!SP3>
                                      OUTPUT  <MEMX!INCMAR>,<SELB!OBA1>       ;OUTPUT LOW ORDER BYTE OF ADDRESS
                                      MICPC=MICPC+1
                                      <MOVE!WROUT!MEMX!INCMAR!<SELB!OBA1>>
                                      OUTPUT  MEMX!INCMAR,<SELB!OBA2> ;OUTPUT HIGH BYTE OF ADDRESS
                                      MICPC=MICPC+1
                                      <MOVE!WROUT!MEMX!INCMAR!<SELB!OBA2>>
                                      SP      IBUS,UBBR,SP0           ;READ THE BUS REQ REGISTER
                                      MICPC=MICPC+1
                                      <MOVE!SPX!IBUS!UBBR!SP0>
                                      BRWRTE  IMM,101                 ;MASK OFF ALL BUT NXM AND VEC4 BITS
                                      MICPC=MICPC+1
                                      <MOVE!WRTEBR!IMM!<101>>
```

```
                        ;RCVB--ROUTINE TO HANDLE FIRST CHARACTER OF COUNT FIELD

926  030774
(1)            000450
(1)  030774    063260   SP      BR,AANDB,SP0            ;AND SAVE IN SP0
927  030776
(1)            000451   MICPC=MICPC+1
(1)  030776    003305   <MOVE!SPX!BR!AANDB!SP0>
928                     SP      IMM,300,SP5             ;MASK TO ISOLATE EX. MEM BITS
929                     MICPC=MICPC+1
930  031000             <MOVE!SPX!IMM!300!SP5>
(1)            000452                                  ;NOTE THIS REALLY WRITES A 305 BUT THE
(1)  031000    040665                                  ;5 GETS SHIFTED OUT
931  031002            BRWRTE  MEMX,AANDB!SP5          ;MASK ALL BUT EX. MEM BITS
(1)            000453   MICPC=MICPC+1
(1)  031002    001620   <MOVE!WRTEBR!MEMX!<AANDB!SP5>>
932  031004            BRSHFT                          ;SHIFT THEM INTO THE CORRECT POSITION
(1)            000454   MICPC=MICPC+1
(1)  031004    001620   <MOVE!SHFTBR!WRTEBR!SELB>
933  031006            BRSHFT
(1)            000455   MICPC=MICPC+1
(1)  031006    001620   <MOVE!SHFTBR!WRTEBR!SELB>
934  031010            BRSHFT
(1)            000456   MICPC=MICPC+1
(1)  031010    001620   <MOVE!SHFTBR!WRTEBR!SELB>
935  031012   RB4:     OUT     BR,AORB!OBR                    ;WRITE EX MEM BITS OUT
(1)            000457   MICPC=MICPC+1
(1)  031012    061311   <MOVE!WROUTX!BR!<AORB!OBR>>
936  031014            ALWAYS  IDLE
(1)            000460   MICPC=MICPC+1
(1)  031014    100447   <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
937  031016   RB2:     ALWAYS  I2
(1)            000461   MICPC=MICPC+1
(1)  031016    100453   <JUMP!ALCOND!<I2-INIT&3000*4>!<I2-INIT&777/2>>
938                     .SBTTL  RCVC--ROUTINE TO HANDLE SECOND CHARACTER OF COUNT FIELD, SELECT AND FINA
939                     ;ENTERED FROM IDLE LOOP
940                     ;INTERPRETS SELECT AND FINAL
941                     ;CHECKS FOR COUNT TOO LARGE
942                     ;
943  031020   RCVC:
945  031020            ALWAYS  SELQSY                  ;"CALL" SELECT/QSYNC SUBROUTINE
(1)            000462   MICPC=MICPC+1
(1)  031020    104756   <JUMP!ALCOND!<SELQSY-INIT&3000*4>!<SELQSY-INIT&777/2>>
950  031022            LDMA    IMM,BC                  ;LOAD MAR TO BYTE COUNT
(1)            000463   MICPC=MICPC+1
(1)  031022    010167   <MOVE!LDMAR!IMM!<BC&377>>
951  031024            MEMINC  BR,SELA!SP4             ;SAVE LOW BYTE
(1)            000464   MICPC=MICPC+1
(1)  031024    076604   <MOVE!WRMEM!INCMAR!BR!<SELA!SP4>>
952  031026            MEMINC  BR,SELA!SP5             ;AND NOW HIGH BYTE
(1)            000465   MICPC=MICPC+1
(1)  031026    076605   <MOVE!WRMEM!INCMAR!BR!<SELA!SP5>>
953  031030            STATE   RCVD                    ;SET NEXT STATE TO D
(1)            000466   MICPC=MICPC+1
(1)  031030    000470   <MOVE!WRTEBR!IMM!<RCVD-INIT&777/2>>
954  031032            ALWAYS  REXIT
(1)            000467   MICPC=MICPC+1
(1)  031032    104425   <JUMP!ALCOND!<REXIT-INIT&3000*4>!<REXIT-INIT&777/2>>
```

```
                        .SBTTL  RCVD--ROUTINE TO HANDLE RESPONSE FIELD FOR NUMBERED MESSAGES
                        ;
956
957
958  031034             RCVD:   STATE   RCVE
(1)                             MICPC=MICPC+1
(1)  031034  000470             <MOVE!WRTEBR!IMM!<RCVE-INIT&777/2>>
(1)  031034  000522     RD2:    SP      BR,SELB,SP3             ;SAVE THE STATE
959  031036                     MICPC=MICPC+1
(1)                             <MOVE!SPX!BR!SELB!SP3>
(1)  031036  000471             SPBR    IBUS,RCVDAT,SPO         ;INPUT THE CHARACTER
(1)  031036  063223             MICPC=MICPC+1
960  031040                     <MOVE!SPBRX!IBUS!RCVDAT!SPO>
(1)                             BRWRTE  BR,SUB!SP17             ;COMPARE NEW R TO LAST R
(1)  031040  000472             MICPC=MICPC+1
(1)  031040  023600             <MOVE!WRTEBR!BR!<SUB!SP17>>
961  031042                     BRWRTE  BR,AORNB!SP1            ;IF NEW IS THE SAME OR LESS OR
(1)                             MICPC=MICPC+1
(1)  031042  000473             <MOVE!WRTEBR!BR!<AORNB!SP1>>
(1)  031042  060757             BR7     RD7                     ;MAINT MODE - GET OUT
962  031044                     MICPC=MICPC+1
(1)                             <JUMP!BR7CON!<RD7-INIT&3000*4>!<RD7-INIT&777/2>>
(1)  031044  000474             BRWRTE  BR,SELA!SP10
(1)  031044  060641             MICPC=MICPC+1
963  031046                     <MOVE!WRTEBR!BR!<SELA!SP10>>
(1)                             BRSHFT
(1)  031046  000475             MICPC=MICPC+1
(1)  031046  107514             <MOVE!SHFTBR!WRTEBR!SELB>
964  031050                     BR4     IDLE
(1)                             MICPC=MICPC+1
(1)  031050  000476             <JUMP!BR4CON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
(1)  031050  060610             BRO     RD4                     ;MESSAGE IN PROGRESS
965  031052                     MICPC=MICPC+1
(1)                             <JUMP!BROCON!<RD4-INIT&3000*4>!<RD4-INIT&777/2>>
(1)  031052  000477     RD3:    LDMA    IMM,ISP17               ;ADDRESS LAST ACKED IMAGE
(1)  031052  001620             MICPC=MICPC+1
966  031054                     <MOVE!LDMAR!IMM!<ISP17&377>>
(1)                             MEM     BR,SELA!SPO             ;COPY THE CHAR
(1)  031054  000500             MICPC=MICPC+1
(1)  031054  103047             <MOVE!WRMEM!BR!<SELA!SPO>>
967  031056             RD5:    LDMA    IMM,REPST               ;POINT TO REP THRESHOLD COUNTER
(1)                             MICPC=MICPC+1
(1)  031056  000501             <MOVE!LDMAR!IMM!<REPST&377>>
(1)  031056  106110             MEM     IMM,1                   ;RESET REP THRESHOLD
968  031060                     MICPC=MICPC+1
(1)                             <MOVE!WRMEM!IMM!<1>>
(1)  031060  000502     RD6:    SP      IMM,374,SP14            ;RESET THE COUNT
(1)  031060  010153             MICPC=MICPC+1
969  031062                     <MOVE!SPX!IMM!374!SP14>
(1)                             ALWAYS  IDLE
(1)  031062  000503             MICPC=MICPC+1
(1)  031062  062600             <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
970  031064             RD4:    BRWRTE  BR,SELA!SPO
(1)                             MICPC=MICPC+1
(1)  031064  000504             <MOVE!WRTEBR!BR!<SELA!SPO>>
(1)  031064  010003             BRWRTE  BR,SUBTC!SP12
971  031066                     MICPC=MICPC+1
(1)                             <MOVE!WRTEBR!BR!<SUBTC!SP12>>
(1)  031066  000505
(1)  031066  002401
972  031070
(1)                 000506
(1)  031070  003374
973  031072
(1)                 000507
(1)  031072  100447
974  031074
(1)                 000510
(1)  031074  060600
975  031076
(1)                 000511
(1)  031076  060772
```

```
 976  031100
 (1)            000512
 (1)  031100    103447                BR7     IDLE
 977  031102                          MICPC=MICPC+1
 (1)            000513                <JUMP!BR7CON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
 (1)  031102    104502                ALWAYS  RD3
 978  031104                          MICPC=MICPC+1
 (1)            000514                <JUMP!ALCOND!<RD3-INIT&3000*4>!<RD3-INIT&777/2>>
 (1)  031104    060600  RD7:          BRWRTE  BR,SELA!SP0
 979  031106                          MICPC=MICPC+1
 (1)            000515                <MOVE!WRTEBR!BR!<SELA!SP0>>
 (1)  031106    060372                CMP     BR,SP12
 980  031110                          MICPC=MICPC+1
 (1)            000516                <SUBTC!BR!SP12>
 (1)  031110    105506                Z       RD6
 981  031112                          MICPC=MICPC+1
 (1)            000517                <JUMP!ZCOND!<RD6-INIT&3000*4>!<RD6-INIT&777/2>>
 (1)  031112    060530                BRWRTE  BR,AA!SP10                      ;IF HDX
 982  031114                          MICPC=MICPC+1
 (1)            000520                <MOVE!WRTEBR!BR!<AA!SP10>>
 (1)  031114    113705                BR7     TIME2                           ;QUEUE UP REP
 983  031116                          MICPC=MICPC+1
 (1)            000521                <JUMP!BR7CON!<TIME2-INIT&3000*4>!<TIME2-INIT&777/2>>
 (1)  031116    100447                ALWAYS  IDLE
 984                                  MICPC=MICPC+1
 985                                  <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
                                      .SBTTL  RCVE--ROUTINE TO HANDLE N FIELD OF NUMBERED MESSAGE
 986  031120                          ;
 (1)            000522  RCVE:   BRWRTE  BR,SELA!SP1             ;READ THE STATUS BYTE
 (1)  031120    060601                MICPC=MICPC+1
 987  031122                          <MOVE!WRTEBR!BR!<SELA!SP1>>
 (1)            000523                BR7     RCVQ
 (1)  031122    107716                MICPC=MICPC+1
 988  031124                          <JUMP!BR7CON!<RCVQ-INIT&3000*4>!<RCVQ-INIT&777/2>>
 (1)            000524                BRWRTE  IBUS,RCVDAT             ;INPUT THE CHARACTER
 (1)  031124    020600                MICPC=MICPC+1
 989  031126                          <MOVE!WRTEBR!IBUS!<RCVDAT>>
 (1)            000525                CMP     BR,SP11
 (1)  031126    060371                MICPC=MICPC+1
 990  031130                          <SUBTC!BR!SP11>
 (1)            000526                Z       5$
 (1)  031130    105531                MICPC=MICPC+1
 991  031132                          <JUMP!ZCOND!<5$-INIT&3000*4>!<5$-INIT&777/2>>
 (1)            000527                SP      BR,DECA,SP13            ;FORCE MSG TYPE TO -1
 (1)  031132    063173                MICPC=MICPC+1
 992  031134                          <MOVE!SPX!BR!DECA!SP13>
 (1)            000530                ALWAYS  RE2
 (1)  031134    104532                MICPC=MICPC+1
 993  031136          5$:   SP        BR,INCA,SP11            ;UPDATE R FIELD
 (1)            000531                MICPC=MICPC+1
 (1)  031136    063071                <MOVE!SPX!BR!INCA!SP11>
 994  031140          RE2:  STATE     RCVF                    ;NEXT RECEIVE STATE IS F
 (1)            000532                MICPC=MICPC+1
 (1)  031140    000534                <MOVE!WRTEBR!IMM!<RCVF-INIT&777/2>>
 995  031142                          ALWAYS  REXIT
 (1)            000533                MICPC=MICPC+1
 (1)  031142    104425                <JUMP!ALCOND!<REXIT-INIT&3000*4>!<REXIT-INIT&777/2>>
```

```
 996
 997  031144                        .SBTTL  RCVF--ROUTINE TO IGNORE ADDRESS
 (1)            000534
 (1)  031144    063164  RCVF:   SP      BR,DECA,SP4             ;DECREMENTLOW BYTE OF COUNT
                                MICPC=MICPC+1
                                <MOVE!SPX!BR!DECA!SP4>
 998  031146                            C       RCVFO           ;NO OVERFLOW
 (1)            000535          MICPC=MICPC+1
 (1)  031146    105137          <JUMP!CCOND!<RCVFO-INIT&3000*4>!<RCVFO-INIT&777/2>>
 999  031150                            SP      BR,DECA,SP5     ;OVERFLOW - DECREMENT HIGH BYTE
 (1)            000536          MICPC=MICPC+1
 (1)  031150    063165          <MOVE!SPX!BR!DECA!SP5>
1000  031152            RCVFO:  STATE   RCVG
 (1)            000537          MICPC=MICPC+1
 (1)  031152    000542          <MOVE!WRTEBR!IMM!<RCVG-INIT&777/2>>
1001  031154            RCVF1:  NOP     IBUS,RCVDAT,0           ;INPUT CHARACTER - AND DISCARD
 (1)            000540          MICPC=MICPC+1
 (1)  031154    020200          <IBUS!RCVDAT!0>
1002  031156                    ALWAYS  REXIT
 (1)            000541          MICPC=MICPC+1
 (1)  031156    104425          <JUMP!ALCOND!<REXIT-INIT&3000*4>!<REXIT-INIT&777/2>>
1003                            ;
1004                            .SBTTL  RCVG--ROUTINE TO IGNORE CRC1
1005                            ;
1006  031160            RCVG:   STATE   RCVH                    ;NEXT STATE IS RCVH
 (1)            000542          MICPC=MICPC+1
 (1)  031160    000544          <MOVE!WRTEBR!IMM!<RCVH-INIT&777/2>>
1007  031162                    ALWAYS  RCVF1
 (1)            000543          MICPC=MICPC+1
 (1)  031162    104540          <JUMP!ALCOND!<RCVF1-INIT&3000*4>!<RCVF1-INIT&777/2>>
```

```
1009
1010
1011  031164                          .SBTTL  RCVH--ROUTINE TO HANDLE CRC2 AND TO DISPATCH NUMBERED AND UNNUMBERED TYP
1012  031164                          ;
 (1)            000544
 (1)  031164    023200  RCVH:   SP      IBUS,RCVDAT,SPO                 ;GET CHAR IN SPO
                        MICPC=MICPC+1
                        <MOVE!SPX!IBUS!RCVDAT!SPO>
1013  031166           BRWRTE  IBUS,RCVCON                             ;READ RECVR CONTROL REGISTER
 (1)            000545 MICPC=MICPC+1
 (1)  031166    020640 <MOVE!WRTEBR!IBUS!<RCVCON>>
1014  031170           BRO     TDON1                                   ;IF BCC MATCH SET CRC IS GOOD
 (1)            000546 MICPC=MICPC+1
 (1)  031170    116157 <JUMP!BROCON!<TDON1-INIT&3000*4>!<TDON1-INIT&777/2>>
1015  031172           BRWRTE  BR,SELA!SP1                             ;READ STATUS BYTE
 (1)            000547 MICPC=MICPC+1
 (1)  031172    060601 <MOVE!WRTEBR!BR!<SELA!SP1>>
1016  031174           BR7     RHX                                     ;MAINT MODE
 (1)            000550 MICPC=MICPC+1
 (1)  031174    107746 <JUMP!BR7CON!<RHX-INIT&3000*4>!<RHX-INIT&777/2>>
1017  031176           BRWRTE  DP,<SELA!SP10>                          ;READ PORT STATUS WORD TO BR
 (1)            000551 MICPC=MICPC+1
 (1)  031176    060610 <MOVE!WRTEBR!DP!<SELA!SP10>>
1018  031200           BRSHFT
 (1)            000552 MICPC=MICPC+1
 (1)  031200    001620 <MOVE!SHFTBR!WRTEBR!SELB>
1019  031202           BR4     SNAK1                                   ;IF START MODE--PROCEED TO RESEND START
 (1)            000553 MICPC=MICPC+1
 (1)  031202    117303 <JUMP!BR4CON!<SNAK1-INIT&3000*4>!<SNAK1-INIT&777/2>>
1020  031204           LDMA    IMM,ISP17                               ;ELSE BCC ERROR--POINT TO IMAGE OF SP17 IN RAM
 (1)            000554 MICPC=MICPC+1
 (1)  031204    010153 <MOVE!LDMAR!IMM!<ISP17&377>>
1021  031206           MEM     BR,SELA!SP17                            ;RESTORE LAST ACKED IMAGE
 (1)            000555 MICPC=MICPC+1
 (1)  031206    062617 <MOVE!WRMEM!BR!<SELA!SP17>>
1022  031210           BRWRTE  IMM,301                                 ;WRITE HEADER BCC ERROR SUBTYPE
 (1)            000556 MICPC=MICPC+1
 (1)  031210    000701 <MOVE!WRTEBR!IMM!<301>>
1023  031212           LDMA    IMM,NHDS                                ;ADDRESS CUM ERROR COUNTER
 (1)            000557 MICPC=MICPC+1
 (1)  031212    010013 <MOVE!LDMAR!IMM!<NHDS&377>>
1024  031214   RH5:    SP      MEMX,SELB,SPO                           ;WRITE IT TO SPO
 (1)            000560 MICPC=MICPC+1
 (1)  031214    043220 <MOVE!SPX!MEMX!SELB!SPO>
1025  031216           MEM     BR,INCA!SPO                             ;INCREMENT IT
 (1)            000561 MICPC=MICPC+1
 (1)  031216    062460 <MOVE!WRMEM!BR!<INCA!SPO>>
1026  031220           LDMA    IMM,T                                           ;LOAD ADDRESS OF TYPE FIELD
 (1)            000562 MICPC=MICPC+1
 (1)  031220    010151 <MOVE!LDMAR!IMM!<T&377>>
1027  031222           MEMINC  IMM,2                                           ;WRITE NAK TYPE
 (1)            000563 MICPC=MICPC+1
 (1)  031222    016402 <MOVE!WRMEM!INCMAR!IMM!<2>>
1028  031224           MEM     BR,SELB                                         ;WRITE NAK SUBYTPE FIELD SUBTYPE
 (1)            000564 MICPC=MICPC+1
 (1)  031224    062620 <MOVE!WRMEM!BR!<SELB>>
1029  031226           LDMA    IMM,NAKST                               ;ADDRESS NAKS TMTED DYNAMIC
 (1)            000565 MICPC=MICPC+1
```

```
  (1)  031226  010001                <MOVE!LDMAR!IMM!<NAKST&377>>
1030   031230                        BRWRTE  MEMX,SELB                ;WRITE IT TO BR
  (1)           000566               MICPC=MICPC+1
  (1)  031230  040620                <MOVE!WRTEBR!MEMX!<SELB>>
1031   031232                        BSHFTB                           ;SHIFT IT RIGHT
  (1)           000567               MICPC=MICPC+1
  (1)  031232  061620                <MOVE!SHFTBR!SELB!BR>
1032   031234                        MEM     BR,SELB                  ;UPDATE IT
  (1)           000570               MICPC=MICPC+1
  (1)  031234  062620                <MOVE!WRMEM!BR!<SELB>>
1033   031236                        BRO     NTHRES                   ;BRANCH IF THRESHOLD EXCEEDED
  (1)           000571               MICPC=MICPC+1
  (1)  031236  116252                <JUMP!BROCON!<NTHRES-INIT&3000*4>!<NTHRES-INIT&777/2>>
1034   031240                        ALWAYS  SNAK
  (1)           000572               MICPC=MICPC+1
  (1)  031240  114700                <JUMP!ALCOND!<SNAK-INIT&3000*4>!<SNAK-INIT&777/2>>
1035   031242               RH3:     BRWRTE  DP,<DECA!SP13>           ;LOAD TYPE RECEIVED--DECREMENTING
  (1)           000573               MICPC=MICPC+1
  (1)  031242  060573                <MOVE!WRTEBR!DP!<DECA!SP13>>
1036   031244                        Z       RH1                      ;IF ALUOUT IS ALL ONES IS NUMBERED MSG
  (1)           000574               MICPC=MICPC+1
  (1)  031244  115456                <JUMP!ZCOND!<RH1-INIT&3000*4>!<RH1-INIT&777/2>>
1037   031246                        RSTATE  RCVA
  (1)           000575               MICPC=MICPC+1
  (1)  031246  000400                <MOVE!WRTEBR!IMM!<RCVA-INIT&777/2>>
  (1)           000576               MICPC=MICPC+1
  (1)  031250  063223                <MOVE!SPX!BR!SELB!SP3>
1038   031252                        BRWRTE  DP,<SELA!SP10>           ;LOAD LINE STATUS WORD IN BR
  (1)           000577               MICPC=MICPC+1
  (1)  031252  060610                <MOVE!WRTEBR!DP!<SELA!SP10>>
1040   031254                        BR4     FLUSH1
  (1)           000600               MICPC=MICPC+1
  (1)  031254  117024                <JUMP!BR4CON!<FLUSH1-INIT&3000*4>!<FLUSH1-INIT&777/2>>
1041   031256               CG1:
1046   031256                        BRSHFT                           ;SHIFT RIGHT
  (1)           000601               MICPC=MICPC+1
  (1)  031256  001620                <MOVE!SHFTBR!WRTEBR!SELB>
1047   031260                        BR4     10$
  (1)           000602               MICPC=MICPC+1
  (1)  031260  107210                <JUMP!BR4CON!<10$-INIT&3000*4>!<10$-INIT&777/2>>
1048   031262                        LDMA    IMM,TYPTAB               ;ADDRESS TYPE TABLE
  (1)           000603               MICPC=MICPC+1
  (1)  031262  010162                <MOVE!LDMAR!IMM!<TYPTAB&377>>
1049   031264                        CMP     <MEMX!INCMAR>,SP13
  (1)           000604               MICPC=MICPC+1
  (1)  031264  054373                <SUBTC!MEMX!INCMAR!SP13>
1050   031266                        Z       REP
  (1)           000605               MICPC=MICPC+1
  (1)  031266  115402                <JUMP!ZCOND!<REP-INIT&3000*4>!<REP-INIT&777/2>>
1051   031270                        CMP     <MEMX!INCMAR>,SP13
  (1)           000606               MICPC=MICPC+1
  (1)  031270  054373                <SUBTC!MEMX!INCMAR!SP13>
1052   031272                        Z       NAK
  (1)           000607               MICPC=MICPC+1
  (1)  031272  115426                <JUMP!ZCOND!<NAK-INIT&3000*4>!<NAK-INIT&777/2>>
1053   031274               10$:     LDMA    IMM,TYPSTT               ;SET POINTER TO START TYPE
```

```
 (1)            000610          MICPC=MICPC+1
 (1)  031274    010164          <MOVE!LDMAR!IMM!<TYPSTT&377>>
1054  031276                    CMP     <MEMX!INCMAR>,SP13
 (1)            000611          MICPC=MICPC+1
 (1)  031276    054373          <SUBTC!MEMX!INCMAR!SP13>
1055  031300                    Z       START
 (1)            000612          MICPC=MICPC+1
 (1)  031300    115516          <JUMP!ZCOND!<START-INIT&3000*4>!<START-INIT&777/2>>
1056                                                        ;STACK TYPE
1057  031302                    CMP     <MEMX!INCMAR>,SP13
 (1)            000613          MICPC=MICPC+1
 (1)  031302    054373          <SUBTC!MEMX!INCMAR!SP13>
1058  031304                    Z       STACK
 (1)            000614          MICPC=MICPC+1
 (1)  031304    115411          <JUMP!ZCOND!<STACK-INIT&3000*4>!<STACK-INIT&777/2>>
1059  031306                    CMP     <MEMX!INCMAR>,SP13      ;ACK TYPE
 (1)            000615          MICPC=MICPC+1
 (1)  031306    054373          <SUBTC!MEMX!INCMAR!SP13>
1060  031310                    Z       ACK
 (1)            000616          MICPC=MICPC+1
 (1)  031310    105773          <JUMP!ZCOND!<ACK-INIT&3000*4>!<ACK-INIT&777/2>>
1061  031312                    ALWAYS  IDLE                    ;OTHERWISE IGNORE--MUST BE OBS MSG
 (1)            000617          MICPC=MICPC+1
 (1)  031312    100447          <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
```

```
1064  031314
 (1)            000620
 (1)  031314    023640
1065  031316
 (1)            000621
 (1)  031316    060400
1066  031320
 (1)            000622
 (1)  031320    103451
1067  031322
 (1)            000623
 (1)  031322    110402
1069
1070
1071  031324
 (1)            000624
 (1)  031324    123600
1072  031326
 (1)            000625
 (1)  031326    102047
1073  031330
 (1)            000626
 (1)  031330    022203
1074  031332
 (1)            000627
 (1)  031332    000672
1075  031334
 (1)            000630
 (1)  031334    063223
1076  031336
 (1)            000631
 (1)  031336    000621
1077  031340
 (1)            000632
 (1)  031340    104641
1078
1079  031342
 (1)            000633
 (1)  031342    123600
1084  031344
 (1)            000634
 (1)  031344    102047
1086  031346
 (1)            000635
 (1)  031346    000646
1087  031350
 (1)            000636
 (1)  031350    063223
1088  031352
 (1)            000637
 (1)  031352    022203
1089  031354
 (1)            000640
 (1)  031354    000421
1090  031356
 (1)            000641
```

```
RCVH--ROUTINE TO HANDLE CRC2 AND TO DISPATCH NUMBERED AND UNNUMBERED TYPES

RCVCK:  SPBR    IBUS,RCVCON,SPO                 ;READ RCVR CONTROL CSR
        MICPC=MICPC+1
        <MOVE!SPBRX!IBUS!RCVCON!SPO>
        BRWRTE  BR,ADD!SPO                      ;SHIFT LEFT
        MICPC=MICPC+1
        <MOVE!WRTEBR!BR!<ADD!SPO>>
        BR7     I1
        MICPC=MICPC+1
        <JUMP!BR7CON!<I1-INIT&3000*4>!<I1-INIT&777/2>>
        ALWAYS  TA1
        MICPC=MICPC+1
        <JUMP!ALCOND!<TA1-INIT&3000*4>!<TA1-INIT&777/2>>
;*****************TIME CRITICAL CODE-- CHANGE WITH GREAT CARE*********************
        .SBTTL  RCVK01--ROUTINE TO HANDLE FIRST BYTE ODD RECEIVE
RCVK01: SPBR    IBUS,NPR,SPO                    ;READ NPR REGISTER
        MICPC=MICPC+1
        <MOVE!SPBRX!IBUS!NPR!SPO>
        BR0     IDLE
        MICPC=MICPC+1
        <JUMP!BR0CON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
        OUTPUT  IBUS,RCVDAT!OUTDA2              ;OUTPUT A CHAR
        MICPC=MICPC+1
        <MOVE!WROUT!IBUS!<RCVDAT!OUTDA2>>
        STATE   RKE1
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<RKE1-INIT&777/2>>
RCVK02: SP      BR,SELB,SP3                     ;SET STATE
        MICPC=MICPC+1
        <MOVE!SPX!BR!SELB!SP3>
        BRWRTE  IMM,221
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<221>>
        ALWAYS  RK7
        MICPC=MICPC+1
        <JUMP!ALCOND!<RK7-INIT&3000*4>!<RK7-INIT&777/2>>
        .SBTTL  RCVK0--PROCESS ODD CHARACTER
RCVK0:  SPBR    IBUS,NPR,SPO                    ;IS AN NPR GOING
        MICPC=MICPC+1
        <MOVE!SPBRX!IBUS!NPR!SPO>
        BR0     IDLE                            ;IF SO, GO BACK TO IDLE LOOP
        MICPC=MICPC+1
        <JUMP!BR0CON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
        STATE   RCVKE
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<RCVKE-INIT&777/2>>
        SP      BR,SELB,SP3                     ;SET STATE
        MICPC=MICPC+1
        <MOVE!SPX!BR!SELB!SP3>
        OUTPUT  IBUS,RCVDAT!OUTDA2              ;OUTPUT A CHAR
        MICPC=MICPC+1
        <MOVE!WROUT!IBUS!<RCVDAT!OUTDA2>>
RK8:    BRWRTE  IMM,21                          ;SET OUT NPR (C1) AND NPR REQ
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<21>>
RK7:    OUT     BR,<AORB!ONPR>                  ;WRITE NPR REGISTER
        MICPC=MICPC+1
```

```
  (1)  031356  061310
1091   031360
  (1)          000642
  (1)  031360  100447
1092   031362
1096   031362
  (1)          000643
  (1)  031362  123600
1100   031364
  (1)          000644
  (1)  031364  000720
1101   031366
  (1)          000645
  (1)  031366  104630
1102
1103   031370
  (1)          000646
  (1)  031370  120600
1108   031372
  (1)          000647
  (1)  031372  102047
1110   031374
  (1)          000650
  (1)  031374  023140
1111   031376
  (1)          000651
  (1)  031376  062066
1112   031400
  (1)          000652
  (1)  031400  063164
1113   031402
  (1)          000653
  (1)  031402  105256
1114   031404
  (1)          000654
  (1)  031404  063165
1115   031406
  (1)          000655
  (1)  031406  105724
1116   031410
  (1)          000656
  (1)  031410  022202
1117   031412
  (1)          000657
  (1)  031412  023140
1118   031414
  (1)          000660
  (1)  031414  062066
1120   031416
  (1)          000661
  (1)  031416  023160
1121   031420
  (1)          000662
  (1)  031420  062107
1123   031422
  (1)          000663
```

RCVKO--PROCESS ODD CHARACTER

```
        <MOVE!WROUTX!BR!<AORB!ONPR>>
        ALWAYS  IDLE
        MICPC=MICPC+1
        <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
RL4:
        SPBR    IBUS,NPR,SPO
        MICPC=MICPC+1
        <MOVE!SPBRX!IBUS!NPR!SPO>
        STATE   RCVL
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<RCVL-INIT&777/2>>
        ALWAYS  RCVKO2
        MICPC=MICPC+1
        <JUMP!ALCOND!<RCVKO2-INIT&3000*4>!<RCVKO2-INIT&777/2>>
        .SBTTL  RCVKE--HANDLE EVEN BYTES
RCVKE:  BRWRTE  IBUS,NPR                        ;READ NPR CONTROL REGISTER
        MICPC=MICPC+1
        <MOVE!WRTEBR!IBUS!<NPR>>
        BRO     IDLE
        MICPC=MICPC+1
        <JUMP!BROCON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
RK5:    SP      IBUS,IOBA1,SPO                  ;READ LOW BYTE OF BA TO SP
        MICPC=MICPC+1
        <MOVE!SPX!IBUS!IOBA1!SPO>
        OUTPUT  DP,<INCA!OBA1>                  ;WRITE INCREMENTED BA
        MICPC=MICPC+1
        <MOVE!WROUT!DP!<INCA!OBA1>>
RK50:   SP      BR,DECA,SP4                     ;DECREMENT CHARACTER COUNT
        MICPC=MICPC+1
        <MOVE!SPX!BR!DECA!SP4>
        C       10$                             ;NO OVERFLOW
        MICPC=MICPC+1
        <JUMP!CCOND!<10$-INIT&3000*4>!<10$-INIT&777/2>>
        SP      BR,DECA,SP5                     ;OVERFLOW - DECREMENT HIGH BYTE
        MICPC=MICPC+1
        <MOVE!SPX!BR!DECA!SP5>
        Z       RL3                             ;BYTE COUNT ZERO
        MICPC=MICPC+1
        <JUMP!ZCOND!<RL3-INIT&3000*4>!<RL3-INIT&777/2>>
10$:    OUTPUT  IBUS,<RCVDAT!OUTDA1>            ;READ CHARACTER AND WRITE IT
        MICPC=MICPC+1
        <MOVE!WROUT!IBUS!<RCVDAT!OUTDA1>>
        SP      IBUS,IOBA1,SPO                  ;READ INCREMENTED BA
        MICPC=MICPC+1
        <MOVE!SPX!IBUS!IOBA1!SPO>
        OUTPUT  DP,<INCA!OBA1>                  ;WRITE INCREMENTED BA
        MICPC=MICPC+1
        <MOVE!WROUT!DP!<INCA!OBA1>>
        SP      IBUS,IOBA2,SPO                  ;GET BA HIGH
        MICPC=MICPC+1
        <MOVE!SPX!IBUS!IOBA2!SPO>
        OUTPUT  DP,<AC!OBA2>                    ;INCREMENT IF CARRY SET
        MICPC=MICPC+1
        <MOVE!WROUT!DP!<AC!OBA2>>
        C       ICBA23                          ;OVER FLOW
        MICPC=MICPC+1
```

```
                                RCVKE--HANDLE EVEN BYTES

                                        <JUMP!CCOND!<ICBA23-INIT&3000*4>!<ICBA23-INIT&777/2>>
                                RK3:    SP      BR,DECA,SP4                     ;DECREMENT THE COUNT OF BYTES
                                        MICPC=MICPC+1
                                        <MOVE!SPX!BR!DECA!SP4>
                                        C       RK6                             ;NO OVERFLOW
                                        MICPC=MICPC+1
                                        <JUMP!CCOND!<RK6-INIT&3000*4>!<RK6-INIT&777/2>>
                                        SP      BR,DECA,SP5                     ;DECREMENT HIGH BYTE OF COUNT
                                        MICPC=MICPC+1
                                        <MOVE!SPX!BR!DECA!SP5>
                                        Z       RL4                             ;BYTE COUNT ZERO
                                        MICPC=MICPC+1
                                        <JUMP!ZCOND!<RL4-INIT&3000*4>!<RL4-INIT&777/2>>
                                RK6:    STATE   RCVKO
                                        MICPC=MICPC+1
                                        <MOVE!WRTEBR!IMM!<RCVKO-INIT&777/2>>
                                        ALWAYS  REXIT
                                        MICPC=MICPC+1
                                        <JUMP!ALCOND!<REXIT-INIT&3000*4>!<REXIT-INIT&777/2>>
                                        ;
                                RKE1:   SP      IBUS,NPR,SPO                    ;READ NPR REGISTER
                                        MICPC=MICPC+1
                                        <MOVE!SPX!IBUS!NPR!SPO>
                                        BRWRTE  IMM,177                         ;MASK FOR ALL BUT CO
                                        MICPC=MICPC+1
                                        <MOVE!WRTEBR!IMM!<177>>
                                        OUT     BR,<AANDB!ONPR>                 ;TURN OFF ALL BUT CO
                                        MICPC=MICPC+1
                                        <MOVE!WROUTX!BR!<AANDB!ONPR>>
                                        ALWAYS  RK50
                                        MICPC=MICPC+1
                                        <JUMP!ALCOND!<RK50-INIT&3000*4>!<RK50-INIT&777/2>>
                                ;*****************END OF TIME CRITICAL PATH*********************************

                                RCVKE0: SP      IBUS,RCVDAT,SPO                 ;READ CHARACTER AND SAVE IN SPO
                                        MICPC=MICPC+1
                                        <MOVE!SPX!IBUS!RCVDAT!SPO>
                                        OUTPUT  BR,<SELA!OUTDA1>                ;SEND NONSENSE CHARACTER
                                        MICPC=MICPC+1
                                        <MOVE!WROUT!BR!<SELA!OUTDA1>>
                                        BRWRTE  BR,SELA!SP1                     ;READ STATUS BYTE
                                        MICPC=MICPC+1
                                        <MOVE!WRTEBR!BR!<SELA!SP1>>
                                        BR7     PASWRD                          ;MAINT MODE - SEE IF RLD MESSAGE
                                        MICPC=MICPC+1
                                        <JUMP!BR7CON!<PASWRD-INIT&3000*4>!<PASWRD-INIT&777/2>>
                                        ALWAYS  RK3                             ;OTHERWISE PROCESS NORMALLY
                                        MICPC=MICPC+1
                                        <JUMP!ALCOND!<RK3-INIT&3000*4>!<RK3-INIT&777/2>>
```

```
1154
1155  031462                        .SBTTL  RCVI--STORE UNNUMBERED MESSAGE TYPE
                                RCVI:   SP      IBUS,RCVDAT,SP13        ;STORE UNNUMBERED TYPE
                                        MICPC=MICPC+1
  (1)         000703                    <MOVE!SPX!IBUS!RCVDAT!SP13>
  (1)  031462 023213
1156  031464                            STATE   RCVJ                    ;NEXT STATE IS J
                                        MICPC=MICPC+1
  (1)         000704                    <MOVE!WRTEBR!IMM!<RCVJ-INIT&777/2>>
  (1)  031464 000706
1157  031466                            ALWAYS  REXIT
                                        MICPC=MICPC+1
  (1)         000705                    <JUMP!ALCOND!<REXIT-INIT&3000*4>!<REXIT-INIT&777/2>>
  (1)  031466 104425
1158                                    ;
1159                                    .SBTTL  RCVJ--ROUTINE TO HANDLE SUBTYPE FIELD,SELECT AND FINAL
1160  031470                        RCVJ:
1162  031470                            ALWAYS  SELQSY          ;"CALL" SELECT AND QSYNC SUBROUTINE
                                        MICPC=MICPC+1
  (1)         000706                    <JUMP!ALCOND!<SELQSY-INIT&3000*4>!<SELQSY-INIT&777/2>>
  (1)  031470 104756
1170  031472                            STATE   RCVR                    ;NEXT STATE IS N
                                        MICPC=MICPC+1
  (1)         000707                    <MOVE!WRTEBR!IMM!<RCVR-INIT&777/2>>
  (1)  031472 000711
1171  031474                            ALWAYS  REXIT
                                        MICPC=MICPC+1
  (1)         000710                    <JUMP!ALCOND!<REXIT-INIT&3000*4>!<REXIT-INIT&777/2>>
  (1)  031474 104425
1172                                    .SBTTL  RCVR--UNNUMBERED MESSAGE RESPONSE FIELD
1173                                    ;ENTERED FROM IDLE LOOP
1174                                    ;
1175  031476                        RCVR:   BRWRTE  IMM,3                   ;REP MESSAGE TYPE TO BR
                                        MICPC=MICPC+1
  (1)         000711                    <MOVE!WRTEBR!IMM!<3>>
  (1)  031476 000403
1176  031500                            NOP     BR,SUB,SP13             ;IS TYPE ACK OR NAK
                                        MICPC=MICPC+1
  (1)         000712                    <BR!SUB!SP13>
  (1)  031500 060353
1177  031502                            STATE   RCVQ                    ;NEXT STATE IS RCVQ
                                        MICPC=MICPC+1
  (1)         000713                    <MOVE!WRTEBR!IMM!<RCVQ-INIT&777/2>>
  (1)  031502 000716
1178                                                                    ;***NOTE THIS INSTR DOES NOT CLOCK "C"
1179  031504                            C       RCVF1                   ;IF NOT IGNORE
                                        MICPC=MICPC+1
  (1)         000714                    <JUMP!CCOND!<RCVF1-INIT&3000*4>!<RCVF1-INIT&777/2>>
  (1)  031504 105140
1180  031506                            ALWAYS  RD2                     ;DO RANGE CHECKS
                                        MICPC=MICPC+1
  (1)         000715                    <JUMP!ALCOND!<RD2-INIT&3000*4>!<RD2-INIT&777/2>>
  (1)  031506 104471
1181                                    .SBTTL  RCVQ--UNNUMBERED MESSAGE--NUMBER FIELD
1182                                    ;ENTER  FROM IDLE
1183                                    ;
1184  031510                        RCVQ:   STATE   RCVF                    ;NEXT STATE IS ADDRESS
                                        MICPC=MICPC+1
  (1)         000716                    <MOVE!WRTEBR!IMM!<RCVF-INIT&777/2>>
  (1)  031510 000534
1185  031512                            ALWAYS  RCVF1
                                        MICPC=MICPC+1
  (1)         000717                    <JUMP!ALCOND!<RCVF1-INIT&3000*4>!<RCVF1-INIT&777/2>>
  (1)  031512 104540
```

```
                        .SBTTL  RCVL--PROCESS CRC3
                        ;ENTERED FROM  IDLE LOOP
1187
1188
1189  031514           RCVL:  SPBR    IBUS,NPR,SPO                    ;READ NPR CONTROL
 (1)          000720          MICPC=MICPC+1
 (1)  031514  123600          <MOVE!SPBRX!IBUS!NPR!SPO>
1194  031516                 BRO     IDLE
 (1)          000721          MICPC=MICPC+1
 (1)  031516  102047          <JUMP!BROCON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
1196  031520           RL2:   BRWRTE  IMM,176                         ;MASK TO TURN OFF CO
 (1)          000722          MICPC=MICPC+1
 (1)  031520  000576          <MOVE!WRTEBR!IMM!<176>>
1197  031522                 OUT     BR,AANDB!ONPR
 (1)          000723          MICPC=MICPC+1
 (1)  031522  061270          <MOVE!WROUTX!BR!<AANDB!ONPR>>
1198                           ;
1199  031524           RL3:   STATE   RCVM
 (1)          000724          MICPC=MICPC+1
 (1)  031524  000726          <MOVE!WRTEBR!IMM!<RCVM-INIT&777/2>>
1200  031526                 ALWAYS  RCVF1
 (1)          000725          MICPC=MICPC+1
 (1)  031526  104540          <JUMP!ALCOND!<RCVF1-INIT&3000*4>!<RCVF1-INIT&777/2>>
1201                           ;
1206                           .SBTTL  RCVM--PROCESS CRC4--END OF DATA MESSAGE
1207                           ;ENTERED FROM IDLE LOOP
1208                           ;IF CRC CORRECT -- QUEUE INTERUPT AND UPDATE RESPONSE
1209                           ;
1210                           ;IF CRC WRONG SEND NAK
1211  031530           RCVM:  BRWRTE  IBUS,UBBR                       ;READ UNIBUS BR REGISTER
 (1)          000726          MICPC=MICPC+1
 (1)  031530  120620          <MOVE!WRTEBR!IBUS!<UBBR>>
1212  031532                 BRO     NXMERR                          ;NON-EXISTANT MEMORY
 (1)          000727          MICPC=MICPC+1
 (1)  031532  116051          <JUMP!BROCON!<NXMERR-INIT&3000*4>!<NXMERR-INIT&777/2>>
1213  031534                 SP      IBUS,RCVDAT,SPO                 ;READ CRC CHARACTER
 (1)          000730          MICPC=MICPC+1
 (1)  031534  023200          <MOVE!SPX!IBUS!RCVDAT!SPO>
1214  031536                 BRWRTE  IBUS,RCVCON                     ;READ RECEIVER CONTROL REGISTER
 (1)          000731          MICPC=MICPC+1
 (1)  031536  020640          <MOVE!WRTEBR!IBUS!<RCVCON>>
1215  031540                 BRO     RCVM1                           ;IF CRC GOOD -- PROCESS
 (1)          000732          MICPC=MICPC+1
 (1)  031540  116206          <JUMP!BROCON!<RCVM1-INIT&3000*4>!<RCVM1-INIT&777/2>>
1216  031542                 BRWRTE  BR,SELA!SP1                     ;READ STATUS BYTE
 (1)          000733          MICPC=MICPC+1
 (1)  031542  060601          <MOVE!WRTEBR!BR!<SELA!SP1>>
1217  031544                 BR7     RHX                             ;CRC ERROR IN BOOT MODE - FLUSH
 (1)          000734          MICPC=MICPC+1
 (1)  031544  107746          <JUMP!BR7CON!<RHX-INIT&3000*4>!<RHX-INIT&777/2>>
1218  031546                 BRWRTE  IMM,302                         ;DATA ERROR SUBTYPE
 (1)          000735          MICPC=MICPC+1
 (1)  031546  000702          <MOVE!WRTEBR!IMM!<302>>
1219  031550                 LDMA    IMM,NDATS                       ;ADDRESS CUM ERROR COUNTER
 (1)          000736          MICPC=MICPC+1
 (1)  031550  010014          <MOVE!LDMAR!IMM!<NDATS&377>>
1220  031552                 ALWAYS  RH5                             ;SEND NAK
 (1)          000737          MICPC=MICPC+1
```

```
                                <JUMP!ALCOND!<RH5-INIT&3000*4>!<RH5-INIT&777/2>>

1221
1222  031554                    RCVM0:  BRWRTE  IMM,10                  ;MAINT MESS ERROR
 (1)          000740                    MICPC=MICPC+1
 (1)  031554  000410                    <MOVE!WRTEBR!IMM!<10>>
1223  031556                    RCVM2:  LDMA    IMM,<<RTHRS+3>>         ;POINT TO ERROR WORD
 (1)          000741                    MICPC=MICPC+1
 (1)  031556  010177                    <MOVE!LDMAR!IMM!<<RTHRS+3>&377>>
1224  031560                            ALWAYS RCEXY                    ;GIVE FATAL ERROR
 (1)          000742                    MICPC=MICPC+1
 (1)  031560  114507                    <JUMP!ALCOND!<RCEXY-INIT&3000*4>!<RCEXY-INIT&777/2>>
1225                                    .SBTTL  EM2--PROCESS RLD MESSAGE
1226                                    ;ENTERED FROM IDLE LOOP
1227                                    ;IF RLD PASSWORD CHECKS TRIGGER THE BOOT ROM
1228
1229  031562                    EM2:    BRWRTE  IBUS,RCVDAT             ;READ THE CHAR
 (1)          000743                    MICPC=MICPC+1
 (1)  031562  020600                    <MOVE!WRTEBR!IBUS!<RCVDAT>>
1230  031564                            CMP     BR,SP13                 ;IS IT A MATCH
 (1)          000744                    MICPC=MICPC+1
 (1)  031564  060373                    <SUBTC!BR!SP13>
1231  031566                            Z       EM3
 (1)          000745                    MICPC=MICPC+1
 (1)  031566  105754                    <JUMP!ZCOND!<EM3-INIT&3000*4>!<EM3-INIT&777/2>>
1232                                                                    ;FALL INTO RHX
1233  031570                    RHX:    BRWRTE  BR,AA!SP1               ;READ STATUS BYTE SHIFTED LEFT
 (1)          000746                    MICPC=MICPC+1
 (1)  031570  060521                    <MOVE!WRTEBR!BR!<AA!SP1>>
1234  031572                            BR4     10$                     ;DLE RECEIVED IN NORMAL MODE
 (1)          000747                    MICPC=MICPC+1
 (1)  031572  107351                    <JUMP!BR4CON!<10$-INIT&3000*4>!<10$-INIT&777/2>>
1235  031574                            ALWAYS  FLUSH                   ;ALREADY IN MAINT MODE
 (1)          000750                    MICPC=MICPC+1
 (1)  031574  104415                    <JUMP!ALCOND!<FLUSH-INIT&3000*4>!<FLUSH-INIT&777/2>>
1236  031576                    10$:    BRWRTE  IMM,165                 ;MASK TO CLEAR ALL MAINT RELATED BITS
 (1)          000751                    MICPC=MICPC+1
 (1)  031576  000565                    <MOVE!WRTEBR!IMM!<165>>
1237  031600                            SP      BR,AANDB,SP1            ;CLEAR THEM
 (1)          000752                    MICPC=MICPC+1
 (1)  031600  063261                    <MOVE!SPX!BR!AANDB!SP1>
1238  031602                            ALWAYS  FLUSH
 (1)          000753                    MICPC=MICPC+1
 (1)  031602  104415                    <JUMP!ALCOND!<FLUSH-INIT&3000*4>!<FLUSH-INIT&777/2>>
1239  031604                    EM3:    SP      BR,DECA,SP4
 (1)          000754                    MICPC=MICPC+1
 (1)  031604  063164                    <MOVE!SPX!BR!DECA!SP4>
1240  031606                            ALWAYS  IDLE
 (1)          000755                    MICPC=MICPC+1
 (1)  031606  100447                    <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
```

```
1252
1253
1254  031610
 (1)            000756
 (1)  031610    023605
1255  031612
 (1)            000757
 (1)  031612    107763
1256  031614
 (1)            000760
 (1)  031614    060525
1257  031616
 (1)            000761
 (1)  031616    107766
1258  031620
 (1)            000762
 (1)  031620    164463
1259
1260  031622
 (1)            000763
 (1)  031622    000600
1261  031624
 (1)            000764
 (1)  031624    063310
1262  031626
 (1)            000765
 (1)  031626    104760
1263  031630
 (1)            000766
 (1)  031630    000420
1264  031632
 (1)            000767
 (1)  031632    063310
1265  031634
 (1)            000770
 (1)  031634    164463
1267
1275  031636
 (1)            000771
 (1)  031636    000402
1276  031640
 (1)            000772
 (1)  031640    114657
1277
1279  031642
 (1)            000773
 (1)  031642    060530
1280  031644
 (1)            000774
 (1)  031644    103365
1281  031646
 (1)            000775
 (1)  031646    107422
1282  031650
 (1)            000776
 (1)  031650    100447
```

```
        .SBTTL  SELQSY--ROUTINETOCHECK SELECT AND QSYNC AND DIDDLE LINE STATUS WORD
        ;USES SP5. ALWAYS CALLED BY FIRST INSTR IN A RSTATE
SELQSY: SPBR    IBUS,RCVDAT,SP5                 ;READCHARACTERINTO  SP5 AND THE BR
        MICPC=MICPC+1
        <MOVE!SPBRX!IBUS!RCVDAT!SP5>
        BR7     15$                             ;SELECT SET?--BRANCH
        MICPC=MICPC+1
        <JUMP!BR7CON!<15$-INIT&3000*4>!<15$-INIT&777/2>>
5$:     BRWRTE  BR,AA!SP5                       ;SHIFTBR LEFT
        MICPC=MICPC+1
        <MOVE!WRTEBR!BR!<AA!SP5>>
        BR7     20$                             ;FINAL SET?
        MICPC=MICPC+1
        <JUMP!BR7CON!<20$-INIT&3000*4>!<20$-INIT&777/2>>
        .ALWAY  BR,INCA,SP3!PAGE1
        MICPC=MICPC+1
        <JUMP!ALCOND!BR!INCA!SP3!PAGE1>
;
15$:    BRWRTE  IMM,200                         ;SET OK TO SEND
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<200>>
        SP      BR,AORB,SP10                    ;IN LINE STATUS WORD
        MICPC=MICPC+1
        <MOVE!SPX!BR!AORB!SP10>
        ALWAYS  5$
        MICPC=MICPC+1
        <JUMP!ALCOND!<5$-INIT&3000*4>!<5$-INIT&777/2>>
20$:    BRWRTE  IMM,20                          ;SETCLEARACTIVE
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<20>>
        SP      BR,AORB,SP10                    ;IN LINE STATUS WORD
        MICPC=MICPC+1
        <MOVE!SPX!BR!AORB!SP10>
        .ALWAY  BR,INCA,SP3!PAGE1
        MICPC=MICPC+1
        <JUMP!ALCOND!BR!INCA!SP3!PAGE1>
;
RTHRES: BRWRTE  IMM,2
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<2>>
        ALWAYS  ERRXX
        MICPC=MICPC+1
        <JUMP!ALCOND!<ERRXX-INIT&3000*4>!<ERRXX-INIT&777/2>>
;
ACK:    BRWRTE  BR,AA!SP10
        MICPC=MICPC+1
        <MOVE!WRTEBR!BR!<AA!SP10>>
        BR4     ACK1
        MICPC=MICPC+1
        <JUMP!BR4CON!<ACK1-INIT&3000*4>!<ACK1-INIT&777/2>>
        BR7     FLUSHA
        MICPC=MICPC+1
        <JUMP!BR7CON!<FLUSHA-INIT&3000*4>!<FLUSHA-INIT&777/2>>
        ALWAYS  IDLE
        MICPC=MICPC+1
        <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
```

```
1284                    ;
```

```
1286            031654
1287            000777
1288
1289
1290  031654
1291  031654
 (1)            001000
 (1)  031654    060530
1292  031656
 (1)            001001
 (1)  031656    107620
1293  031660
1294  031660
 (1)            001002
 (1)  031660    060610
1295  031662
 (1)            001003
 (1)  031662    113405
1296  031664
 (1)            001004
 (1)  031664    100451
1297  031666
 (1)            001005
 (1)  031666    112032
1298  031670
 (1)            001006
 (1)  031670    112410
1299  031672
 (1)            001007
 (1)  031672    100451
1300  031674
1301  031674
 (1)            001010
 (1)  031674    001620
1302  031676
 (1)            001011
 (1)  031676    103051
1304  031700
 (1)            001012
 (1)  031700    112432
1305
1316  031702
 (1)            001013
 (1)  031702    000450
 (1)            001014
 (1)  031704    063222
1317  031706
 (1)            001015
 (1)  031706    060521
1318  031710
 (1)            001016
 (1)  031710    113021
1319  031712
 (1)            001017
 (1)  031712    000620
1320  031714
```

```
        .=INIT+2000
        MICPC=777
        .SBTTL  TMTA--FIRST CHARACTER OF HEADER
        ;
TMTA:
        BRWRTE  BP,AA!SP10                      ;SHIFT LEFT
        MICPC=MICPC+1
        <MOVE!WRTEBR!BR!<AA!SP10>>
        BR7     RCVCK
        MICPC=MICPC+1
        <JUMP!BR7CON!<RCVCK-INIT&3000*4>!<RCVCK-INIT&777/2>>
TA1:
        BRWRTE  BR,SELA!SP10                    ;REREAD STATUS
        MICPC=MICPC+1
        <MOVE!WRTEBR!BR!<SELA!SP10>>
        BR7     TA2                             ;OK TO SEND?
        MICPC=MICPC+1
        <JUMP!BR7CON!<TA2-INIT&3000*4>!<TA2-INIT&777/2>>
TMTA6:  ALWAYS  I1                              ;NO
        MICPC=MICPC+1
        <JUMP!ALCOND!<I1-INIT&3000*4>!<I1-INIT&777/2>>
TA2:    BR0     NUMSYN                          ;IF UNNUMBPENDING  -- SEND IT
        MICPC=MICPC+1
        <JUMP!BR0CON!<NUMSYN-INIT&3000*4>!<NUMSYN-INIT&777/2>>
        BR1     10$
        MICPC=MICPC+1
        <JUMP!BR1CON!<10$-INIT&3000*4>!<10$-INIT&777/2>>
        ALWAYS  I1
        MICPC=MICPC+1
        <JUMP!ALCOND!<I1-INIT&3000*4>!<I1-INIT&777/2>>
10$:
        BRSHFT
        MICPC=MICPC+1
        <MOVE!SHFTBR!WRTEBR!SELB>
        BR4     I1                              ;IF START MODE EXIT
        MICPC=MICPC+1
        <JUMP!BR4CON!<I1-INIT&3000*4>!<I1-INIT&777/2>>
        BR1     NUMSYN                  ;IF LINE HAS GONE IDLE SEND SYN
        MICPC=MICPC+1
        <JUMP!BR1CON!<NUMSYN-INIT&3000*4>!<NUMSYN-INIT&777/2>>
                                                ;ELSE--START TO SEND MESSAGE
TMTEXT: TSTATE  TMTB                            ;STATE TO SEND COUNT
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<TMTB-INIT&777/2>>
        MICPC=MICPC+1
        <MOVE!SPX!BR!SELB!SP2>
        BRWRTE  BR,AA!SP1                       ;IN PORT STATUS
        MICPC=MICPC+1
        <MOVE!WRTEBR!BR!<AA!SP1>>
        BR4     10$                             ;DLE RCV'D SET-IGNORE MAINT MODE BIT
        MICPC=MICPC+1
        <JUMP!BR4CON!<10$-INIT&3000*4>!<10$-INIT&777/2>>
        BRWRTE  IMM,220                         ;WRITE DLE TO BR
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<220>>
        C       TMTA5                           ;MAINT MODE - SEND DLE
```

TMTA--FIRST CHARACTER OF HEADER

```
(1)             001020
(1)  031714     111024
1321 031716
(1)             001021
(1)  031716     060610
1322 031720
(1)             001022
(1)  031720     112026
1323 031722
(1)             001023
(1)  031722     000601
1324 031724
(1)             001024
(1)  031724     062230
1325 031726
(1)             001025
(1)  031726     100451
1326 031730
(1)             001026
(1)  031730     000601
(1)             001027
(1)  031732     063222
1327 031734
(1)             001030
(1)  031734     000405
1328 031736
(1)             001031
(1)  031736     110424
```

```
                MICPC=MICPC+1
                <JUMP!CCOND!<TMTA5-INIT&3000*4>!<TMTA5-INIT&777/2>>
10$:    BRWRTE  BR,SELA!SP10            ;NUMBERED MESSAGE?
                MICPC=MICPC+1
                <MOVE!WRTEBR!BR!<SELA!SP10>>
        BRO     TMTUN                   ;NO - BRANCH
                MICPC=MICPC+1
                <JUMP!BROCON!<TMTUN-INIT&3000*4>!<TMTUN-INIT&777/2>>
        BRWRTE  IMM,201                 ;OTHERWISE SEND AN SOH
                MICPC=MICPC+1
                <MOVE!WRTEBR!IMM!<201>>
TMTA5:  OUTPUT  BR,<SELB!TMTDAT>        ;IN TMT SILO
                MICPC=MICPC+1
                <MOVE!WROUT!BR!<SELB!TMTDAT>>
                ALWAYS  I1
                MICPC=MICPC+1
                <JUMP!ALCOND!<I1-INIT&3000*4>!<I1-INIT&777/2>>
TMTUN:  TSTATE  TMTI
                MICPC=MICPC+1
                <MOVE!WRTEBR!IMM!<TMTI-INIT&777/2>>
                MICPC=MICPC+1
                <MOVE!SPX!BR!SELB!SP2>
                BRWRTE  IMM,5           ;ENQ TO BR
                MICPC=MICPC+1
                <MOVE!WRTEBR!IMM!<5>>
                ALWAYS  TMTA5
                MICPC=MICPC+1
                <JUMP!ALCOND!<TMTA5-INIT&3000*4>!<TMTA5-INIT&777/2>>
```

TMTA--FIRST CHARACTER OF HEADER

```
1331  031740
 (1)            001032
 (1)  031740    020660
1332  031742
 (1)            001033
 (1)  031742    001620
1333  031744
 (1)            001034
 (1)  031744    103051
1334  031746
 (1)            001035
 (1)  031746    000773
1335  031750
 (1)            001036
 (1)  031750    063270
1336  031752
 (1)            001037
 (1)  031752    000443
 (1)            001040
 (1)  031754    063222
1337  031756
 (1)            001041
 (1)  031756    000410
1338  031760
 (1)            001042
 (1)  031760    063226
1339  031762
 (1)            001043
 (1)  031762    063166
1340  031764
 (1)            001044
 (1)  031764    111413
1341  031766
 (1)            001045
 (1)  031766    002011
1342  031770
 (1)            001046
 (1)  031770    000626
1343  031772
 (1)            001047
 (1)  031772    110424
```

```
NUMSYN: BRWRTE  IBUS,MODEM              ;ARE WE STILL SENDING?
        MICPC=MICPC+1
        <MOVE!WRTEBR!IBUS!<MODEM>>
        BRSHFT
        MICPC=MICPC+1
        <MOVE!SHFTBR!WRTEBR!SELB>
        BR4     I1                      ;RTS SET? IF SO WE ARE--STALL
        MICPC=MICPC+1
        <JUMP!BR4CON!<I1-INIT&3000*4>!<I1-INIT&777/2>>
        BRWRTE  IMM,373                 ;MASK TO TURN OFFLINE IDLE
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<373>>
        SP      BR,AANDB,SP10           ;IN LINE STATUS WORD
        MICPC=MICPC+1
        <MOVE!SPX!BR!AANDB!SP10>
        TSTATE  TMTA1
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<TMTA1-INIT&777/2>>
        MICPC=MICPC+1
        <MOVE!SPX!BR!SELB!SP2>
        BRWRTE  IMM,10
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<10>>
        SP      BR,SELB,SP6                     ;STORE IN SP6
        MICPC=MICPC+1
        <MOVE!SPX!BR!SELB!SP6>
TMTA1:  SP      BR,DECA,SP6                     ;DECREMENT SYN COUNT
        MICPC=MICPC+1
        <MOVE!SPX!BR!DECA!SP6>
        Z       TMTEXT
        MICPC=MICPC+1
        <JUMP!ZCOND!<TMTEXT-INIT&3000*4>!<TMTEXT-INIT&777/2>>
        OUTPUT  IMM,<1!OTMTCO>                  ;WRITE SOM TO TMTR CONTRL
        MICPC=MICPC+1
        <MOVE!WROUT!IMM!<1!OTMTCO>>
        BRWRTE  IMM,226                         ;SYNC CHAR
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<226>>
        ALWAYS  TMTA5
        MICPC=MICPC+1
        <JUMP!ALCOND!<TMTA5-INIT&3000*4>!<TMTA5-INIT&777/2>>
```

```
  1355                          .SBTTL  TMTB--OUTPUT FIRST CHAR OF COUNT
```

```
 1357
 1358  031774                    TMTB:   SPBR    BR,INCA,SP12            ;INCREMENT MSG NO
  (1)          001050                    MICPC=MICPC+1
  (1)  031774  063472                    <MOVE!SPBRX!BR!INCA!SP12>
```

```
1360  031776
 (1)          001051
 (1)  031776  062220
1361  032000
 (1)          001052
 (1)  032000  070216
1362  032002
 (1)          001053
 (1)  032002  016403
1363  032004
 (1)          001054
 (1)  032004  076612
1364  032006
 (1)          001055
 (1)  032006  000543
1365  032010
 (1)          001056
 (1)  032010  060376
1366  032012
 (1)          001057
 (1)  032012  111504
```

```
        OUTPUT  BR,SELB!OINDAT1
        MICPC=MICPC+1
        <MOVE!WROUT!BR!<SELB!OINDAT1>>
        LDMA    BR,SELA!SP16                    ;GETPOINTER TO NEXT TMT LINK
        MICPC=MICPC+1
        <MOVE!LDMAR!BR!<SELA!SP16>>
        MEMINC  IMM,3                           ;WRITE MSG TMTED TO FLAGS
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<3>>
        MEMINC  BR,SELA!SP12                    ;PICK UP MSGNO
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!BR!<SELA!SP12>>
        BRWRTE  IMM,TML8                        ;GET WRAPAROUND ADDRESS
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<TML8>>
        CMP     BR,SP16                         ;WRAPAROUND
        MICPC=MICPC+1
        <SUBTC!BR!SP16>
        Z       TB2
        MICPC=MICPC+1
        <JUMP!ZCOND!<TB2-INIT&3000*4>!<TB2-INIT&777/2>>
```

```
1368  032014
 (1)            001060
 (1)  032014    000406
1369  032016
 (1)            001061
 (1)  032016    063016
1370  032020
 (1)            001062
 (1)  032020    000506
1371  032022
 (1)            001063
 (1)  032022    063222
1372  032024
 (1)            001064
 (1)  032024    056224
1373  032026
 (1)            001065
 (1)  032026    056225
1374  032030
 (1)            001066
 (1)  032030    043227
1375
1376  032032
 (1)            001067
 (1)  032032    123200
1377  032034
 (1)            001070
 (1)  032034    000620
1378  032036
 (1)            001071
 (1)  032036    063260
1379  032040
 (1)            001072
 (1)  032040    003306
1380  032042
 (1)            001073
 (1)  032042    054666
1381  032044
 (1)            001074
 (1)  032044    042230
1382  032046
 (1)            001075
 (1)  032046    001620
1383  032050
 (1)            001076
 (1)  032050    001620
1384  032052
 (1)            001077
 (1)  032052    001620
1385  032054
 (1)            001100
 (1)  032054    001620
1386  032056
 (1)            001101
 (1)  032056    061310
1387  032060
```

```
        BRWRTE  IMM,6                           ;OFFSET TO NEXT LINK
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<6>>
TB1:    SP      BR,ADD,SP16                     ;UPDATE THE POINTER
        MICPC=MICPC+1
        <MOVE!SPX!BR!ADD!SP16>
        STATE   TMTC                            ;ADDRESS TMTR STATE
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<TMTC-INIT&777/2>>
TB0:    SP      BR,SELB,SP2                     ;UPDATE IT
        MICPC=MICPC+1
        <MOVE!SPX!BR!SELB!SP2>
        OUTPUT  <MEMX!INCMAR>,SELB!IBA1         ;WRITELOWBYTEOFADDRESS
        MICPC=MICPC+1
        <MOVE!WROUT!MEMX!INCMAR!<SELB!IBA1>>
        OUTPUT  <MEMX!INCMAR>,SELB!IBA2         ;WRITE HIGH BYTE OF ADDRESS
        MICPC=MICPC+1
        <MOVE!WROUT!MEMX!INCMAR!<SELB!IBA2>>
        SP      MEMX,SELB,SP7   ;HIGH BYTE OF COUNT TO SP7
        MICPC=MICPC+1
        <MOVE!SPX!MEMX!SELB!SP7>
                                ;WAIT TO MASK OFF MEM EXT. BITS
        SP      IBUS,NPR,SP0
        MICPC=MICPC+1
        <MOVE!SPX!IBUS!NPR!SP0>
        BRWRTE  IMM,220
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<220>>
        SP      BR,AANDB,SP0
        MICPC=MICPC+1
        <MOVE!SPX!BR!AANDB!SP0>
        SP      IMM,300,SP6                     ;MASK FOR MXT
        MICPC=MICPC+1
        <MOVE!SPX!IMM!300!SP6>
        BRWRTE  MEMX!INCMAR,AANDB!SP6           ;TURN OFF CC2
        MICPC=MICPC+1
        <MOVE!WRTEBR!MEMX!INCMAR!<AANDB!SP6>>
        OUTPUT  MEMX,SELB!TMTDAT                ;ALSO WRITE COUNT TO TMTR SILO
        MICPC=MICPC+1
        <MOVE!WROUT!MEMX!<SELB!TMTDAT>>
        BRSHFT                                  ;SHIFT BITS INTO CORRECT POSITION
        MICPC=MICPC+1
        <MOVE!SHFTBR!WRTEBR!SELB>
        BRSHFT
        MICPC=MICPC+1
        <MOVE!SHFTBR!WRTEBR!SELB>
        BRSHFT
        MICPC=MICPC+1
        <MOVE!SHFTBR!WRTEBR!SELB>
        BRSHFT
        MICPC=MICPC+1
        <MOVE!SHFTBR!WRTEBR!SELB>
        OUT     BR,AORB!ONPR
        MICPC=MICPC+1
        <MOVE!WROUTX!BR!<AORB!ONPR>>
        SPBR    MEMX,SELB,SP6           ;LOWBYTE OF COUNT TO SP6
```

TMTB--OUTPUT FIRST CHAR OF COUNT

```
(1)              001102
(1)  032060      043626
1389 032062
(1)              001103
(1)  032062      100447
1394 032064
(1)              001104
(1)  032064      000726
1395 032066
(1)              001105
(1)  032066      110461
1396
```

```
        MICPC=MICPC+1
        <MOVE!SPBRX!MEMX!SELB!SP6>
        ALWAYS  IDLE
        MICPC=MICPC+1
        <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
TB2:    BRWRTE  IMM,<<TML1-TML8>&377>           ;GO BACK TOFIRST LINK
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<<TML1-TML8>&377>>
        ALWAYS  TB1
        MICPC=MICPC+1
        <JUMP!ALCOND!<TB1-INIT&3000*4>!<TB1-INIT&777/2>>
        ;
```

```
1398
1399
1400  032070                        .SBTTL  TMTC--OUTPUT SECOND CHAR OF COUNT
                                    ;
TMTC:   032070  001106  BRWRTE  IMM,77                          ;MASK TO CLEAR MXT BITS
(1)     032070  000477  MICPC=MICPC+1
                        <MOVE!WRTEBR!IMM!<77>>
1401  032072            SPBR    BR,AANDB,SP7                    ;CLEAR THEM
(1)             001107  MICPC=MICPC+1
(1)     032072  063667  <MOVE!SPBRX!BR!AANDB!SP7>
1402  032074            OUTPUT  DP,<SELB!TMTDAT>                ;WRITE TO TMT SILO
(1)             001110  MICPC=MICPC+1
(1)     032074  062230  <MOVE!WROUT!DP!<SELB!TMTDAT>>
1404  032076            STATE   TMTD
(1)             001111  MICPC=MICPC+1
(1)     032076  000513  <MOVE!WRTEBR!IMM!<TMTD-INIT&777/2>>
1405  032100            ALWAYS  XEXIT
(1)             001112  MICPC=MICPC+1
(1)     032100  110554  <JUMP!ALCOND!<XEXIT-INIT&3000*4>!<XEXIT-INIT&777/2>>
1407                    ;
1408                    .SBTTL  TMTD--RESPONSE FIELD-NUMBERED MESSAGE
1409  032102    TMTD:   STATE   TMTE
(1)             001113  MICPC=MICPC+1
(1)     032102  000522  <MOVE!WRTEBR!IMM!<TMTE-INIT&777/2>>
1410  032104            SP      BR,DECA,SP6                     ;ADJUSRT COUNT FOR  TWO'S  COMPLEMENT
(1)             001114  MICPC=MICPC+1
(1)     032104  063166  <MOVE!SPX!BR!DECA!SP6>
1411  032106            C       TD2                             ;NO OVERFLOW
(1)             001115  MICPC=MICPC+1
(1)     032106  111117  <JUMP!CCOND!<TD2-INIT&3000*4>!<TD2-INIT&777/2>>
1412  032110            SP      BR,DECA,SP7                     ;DECREMENT HIGH BYTE OF COUNT
(1)             001116  MICPC=MICPC+1
(1)     032110  063167  <MOVE!SPX!BR!DECA!SP7>
1413  032112    TD2:    LDMA    IMM,ISP11                       ;RESP FIELD ADDR TO MAR
(1)             001117  MICPC=MICPC+1
(1)     032112  010171  <MOVE!LDMAR!IMM!<ISP11&377>>
1414  032114    TD3:    OUTPUT  MEMX,SELB!TMTDAT                ;WRITE IT TO SILO
(1)             001120  MICPC=MICPC+1
(1)     032114  042230  <MOVE!WROUT!MEMX!<SELB!TMTDAT>>
1415  032116            ALWAYS  XEXIT
(1)             001121  MICPC=MICPC+1
(1)     032116  110554  <JUMP!ALCOND!<XEXIT-INIT&3000*4>!<XEXIT-INIT&777/2>>
1416                    .SBTTL  TMTE--NUMBER FIELD--NUMBERED MESSAGE
1417  032120    TMTE:
1418  032120            SPBR    IBUS,NPR,SPO                    ;READ NPR CONTROL REGISTER
(1)             001122  MICPC=MICPC+1
(1)     032120  123600  <MOVE!SPBRX!IBUS!NPR!SPO>
1419  032122            BRO     I1                              ;BUSY - GET OUT
(1)             001123  MICPC=MICPC+1
(1)     032122  102051  <JUMP!BROCON!<I1-INIT&3000*4>!<I1-INIT&777/2>>
1420  032124            OUTPUT  IBUS,<INDAT1!TMTDAT>            ;WRITE IT TO THE SILO
(1)             001124  MICPC=MICPC+1
(1)     032124  022010  <MOVE!WROUT!IBUS!<INDAT1!TMTDAT>>
1421  032126            STATE   TMTF
(1)             001125  MICPC=MICPC+1
(1)     032126  000527  <MOVE!WRTEBR!IMM!<TMTF-INIT&777/2>>
1422  032130            ALWAYS  TH3
(1)             001126  MICPC=MICPC+1
```

```
  (1)  032130  110574
1423
1424
1425  032132
  (1)        001127
  (1)  032132  000533
1426  032134
  (1)        001130
  (1)  032134  063222
1427  032136
  (1)        001131
  (1)  032136  000401
1433  032140
  (1)        001132
  (1)  032140  110424
1435
1436  032142
  (1)        001133
  (1)  032142  000402
1437  032144
  (1)        001134
  (1)  032144  062231
1438  032146
  (1)        001135
  (1)  032146  062230
1439  032150
  (1)        001136
  (1)  032150  020500
1440  032152
  (1)        001137
  (1)  032152  112153
1441  032154
  (1)        001140
  (1)  032154  000542
1442  032156
  (1)        001141
  (1)  032156  110554
```

TMTE--NUMBER FIELD--NUMBERED MESSAGE

```
        <JUMP!ALCOND!<TH3-INIT&3000*4>!<TH3-INIT&777/2>>
        .SBTTL  TMTF--NUMBERED MSG ADDRESS FIELD
        ;
TMTF:   STATE   TF1
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<TF1-INIT&777/2>>
TF2:    SP      BR,SELB,SP2
        MICPC=MICPC+1
        <MOVE!SPX!BR!SELB!SP2>
        BRWRTE  IMM,1                           ;LOAD ADDRESS
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<1>>
        ALWAYS  TMTA5
        MICPC=MICPC+1
        <JUMP!ALCOND!<TMTA5-INIT&3000*4>!<TMTA5-INIT&777/2>>
        .SBTTL  TF1-NUMBERED MSG HEADER EOM
TF1:    BRWRTE  IMM,2                           ;EOM  MASK TO BR
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<2>>
        OUTPUT  BR,<SELB!OTMTCO>                ;UPDATE TMTR CONTROL REGISTER
        MICPC=MICPC+1
        <MOVE!WROUT!BR!<SELB!OTMTCO>>
        OUTPUT  BR,<SELB!TMTDAT>                ;OUTPUT A GARBAGE CHAR
        MICPC=MICPC+1
        <MOVE!WROUT!BR!<SELB!TMTDAT>>
        BRWRTE  IBUS,IIBA1                      ;READ LOW ORDER FROM INBA
        MICPC=MICPC+1
        <MOVE!WRTEBR!IBUS!<IIBA1>>
        BRO     TMTF1                           ;IF ODD BYTE--BRANCH
        MICPC=MICPC+1
        <JUMP!BROCON!<TMTF1-INIT&3000*4>!<TMTF1-INIT&777/2>>
        STATE   TMTH
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<TMTH-INIT&777/2>>
        ALWAYS  XEXIT
        MICPC=MICPC+1
        <JUMP!ALCOND!<XEXIT-INIT&3000*4>!<XEXIT-INIT&777/2>>
```

```
1444
1445
1446  032160
 (1)                001142
 (1)   032160  123600
1450  032162
 (1)                001143
 (1)   032162  102051
1451  032164
 (1)                001144
 (1)   032164  022010
1452  032166
 (1)                001145
 (1)   032166  023100
1453  032170
 (1)                001146
 (1)   032170  062064
1454  032172
 (1)                001147
 (1)   032172  063166
1455  032174
 (1)                001150
 (1)   032174  111153
1456  032176
 (1)                001151
 (1)   032176  063167
1457  032200
 (1)                001152
 (1)   032200  101763
1458  032202
1463  032202
 (1)                001153
 (1)   032202  000556
1464  032204
 (1)                001154
 (1)   032204  063222
1465  032206
 (1)                001155
 (1)   032206  100451
1466  032210
1468  032210
 (1)                001156
 (1)   032210  123600
1469  032212
 (1)                001157
 (1)   032212  102051
1471  032214
 (1)                001160
 (1)   032214  022030
1472  032216
 (1)                001161
 (1)   032216  023100
1473  032220
 (1)                001162
 (1)   032220  062064
1475  032222
```

```
;************TIME CRITICAL PATH--MODIFY WITH GREAT CARE
        .SBTTL  TMTH--ROUTINE TO OUTPUT DATA CHARACTERS
TMTH:   SPBR    IBUS,NPR,SPO                    ;READ NPR CONTROL
        MICPC=MICPC+1
        <MOVE!SPBRX!IBUS!NPR!SPO>
        BRO     I1                              ;IF NPR IN PROGRESS --BRANCH
        MICPC=MICPC+1
        <JUMP!BROCON!<I1-INIT&3000*4>!<I1-INIT&777/2>>
5$:     OUTPUT  IBUS,<INDAT1!TMTDAT>            ;WRITE THE EVEN CHAR TO TMT SILO
        MICPC=MICPC+1
        <MOVE!WROUT!IBUS!<INDAT1!TMTDAT>>
        SP      IBUS,IIBA1,SPO                  ;READ LOW BYTE OF BA TO SP
        MICPC=MICPC+1
        <MOVE!SPX!IBUS!IIBA1!SPO>
        OUTPUT  BR,<INCA!IBA1>                  ;OUTPUT INCREMENTED BA
        MICPC=MICPC+1
        <MOVE!WROUT!BR!<INCA!IBA1>>
        SP      BR,DECA,SP6                     ;DECREMENT CHARACTER  COUNT
        MICPC=MICPC+1
        <MOVE!SPX!BR!DECA!SP6>
        C       TH6                             ;NO OVERFLOW
        MICPC=MICPC+1
        <JUMP!CCOND!<TH6-INIT&3000*4>!<TH6-INIT&777/2>>
        SP      BR,DECA,SP7                     ;DECREMENT HIGH BYTE OF COUNT
        MICPC=MICPC+1
        <MOVE!SPX!BR!DECA!SP7>
        Z       HEH1                            ;BYTE  COUNT ZERO
        MICPC=MICPC+1
        <JUMP!ZCOND!<HEH1-INIT&3000*4>!<HEH1-INIT&777/2>>
TH6:
TMTF1:  STATE   TMTHO
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<TMTHO-INIT&777/2>>
XEXIT:  SP      BR,SELB,SP2                     ;STORE NEW TRANSMIT STATE
        MICPC=MICPC+1
        <MOVE!SPX!BR!SELB!SP2>
        ALWAYS  I1
        MICPC=MICPC+1
        <JUMP!ALCOND!<I1-INIT&3000*4>!<I1-INIT&777/2>>
TMTHO:
        SPBR    IBUS,NPR,SPO                    ;NPR BUSY
        MICPC=MICPC+1
        <MOVE!SPBRX!IBUS!NPR!SPO>
        BRO     I1
        MICPC=MICPC+1
        <JUMP!BROCON!<I1-INIT&3000*4>!<I1-INIT&777/2>>
TH9:    OUTPUT  IBUS,<INDAT2!TMTDAT>            ;ODD CHAR TO SILO
        MICPC=MICPC+1
        <MOVE!WROUT!IBUS!<INDAT2!TMTDAT>>
        SP      IBUS,IIBA1,SPO                  ;READ LOW BYTE TO BA
        MICPC=MICPC+1
        <MOVE!SPX!IBUS!IIBA1!SPO>
        OUTPUT  BR,<INCA!IBA1>                  ;OUTPUT THE INCREMENTED BA
        MICPC=MICPC+1
        <MOVE!WROUT!BR!<INCA!IBA1>>
        SP      IBUS,IIBA2,SPO                  ;GET HIGH BYTE
```

```
                                TMTH--ROUTINE TO OUTPUT DATA CHARACTERS

(1)     001163                          MICPC=MICPC+1
(1)  032222 023120                      <MOVE!SPX!IBUS!IIBA2!SPO>
1476 032224                             OUTPUT  BR,<AC!IBA2>                    ;ADD CARRY
(1)     001164                          MICPC=MICPC+1
(1)  032224 062105                      <MOVE!WROUT!BR!<AC!IBA2>>
1478 032226                             C       HOINCH
(1)     001165                          MICPC=MICPC+1
(1)  032226 111376                      <JUMP!CCOND!<HOINCH-INIT&3000*4>!<HOINCH-INIT&777/2>>
1479 032230                     TH8:    SP      BR,DECA,SP6                     ;DECREMENT CHARACTERCOUNT
(1)     001166                          MICPC=MICPC+1
(1)  032230 063166                      <MOVE!SPX!BR!DECA!SP6>
1480 032232                             C       TH7                             ;NO OVERFLOW
(1)     001167                          MICPC=MICPC+1
(1)  032232 111172                      <JUMP!CCOND!<TH7-INIT&3000*4>!<TH7-INIT&777/2>>
1481 032234                             SP      BR,DECA,SP7                     ;DECREMENT HIGH BYTE OF  COUNT
(1)     001170                          MICPC=MICPC+1
(1)  032234 063167                      <MOVE!SPX!BR!DECA!SP7>
1482 032236                             Z       HEH1                            ;BYTE COUNT ZERO
(1)     001171                          MICPC=MICPC+1
(1)  032236 101763                      <JUMP!ZCOND!<HEH1-INIT&3000*4>!<HEH1-INIT&777/2>>
1483 032240                     TH7:    SPBR    IBUS,NPR,SPO                    ;READ NPR REGISTER
(1)     001172                          MICPC=MICPC+1
(1)  032240 123600                      <MOVE!SPBRX!IBUS!NPR!SPO>
1487 032242                             STATE   TMTH
(1)     001173                          MICPC=MICPC+1
(1)  032242 000542                      <MOVE!WRTEBR!IMM!<TMTH-INIT&777/2>>
1488 032244                     TH3:    SP      BR,SELB,SP2                     ;SAVE TSTATE
(1)     001174                          MICPC=MICPC+1
(1)  032244 063222                      <MOVE!SPX!BR!SELB!SP2>
1489 032246                     TH3X:   BRWRTE  IMM,156                         ;CLEAR CO AND C1
(1)     001175                          MICPC=MICPC+1
(1)  032246 000556                      <MOVE!WRTEBR!IMM!<156>>
1490 032250                             SP      BR,AANDB,SPO                    ;CLEAR THE BITS
(1)     001176                          MICPC=MICPC+1
(1)  032250 063260                      <MOVE!SPX!BR!AANDB!SPO>
1491 032252                             OUT     BR,<INCA!ONPR>
(1)     001177                          MICPC=MICPC+1
(1)  032252 061070                      <MOVE!WROUTX!BR!<INCA!ONPR>>
1493 032254                             ALWAYS  RD6
(1)     001200                          MICPC=MICPC+1
(1)  032254 104506                      <JUMP!ALCOND!<RD6-INIT&3000*4>!<RD6-INIT&777/2>>
1500                            ;***************END TIME CRITICAL PATH*****************************
```

```
1502
1503  032256
 (1)                001201
 (1)   032256       010151
1504  032260
 (1)                001202
 (1)   032260       043226
1505  032262
 (1)                001203
 (1)   032262       000605
1506  032264
 (1)                001204
 (1)   032264       110520
1507
1508
1509  032266
 (1)                001205
 (1)   032266       010152
1510  032270
 (1)                001206
 (1)   032270       000610
1511  032272
 (1)                001207
 (1)   032272       110520
1512
1513
1514  032274
 (1)                001210
 (1)   032274       000403
1515  032276
 (1)                001211
 (1)   032276       060346
1516  032300
 (1)                001212
 (1)   032300       000616
 (1)                001213
 (1)   032302       063222
1517  032304
 (1)                001214
 (1)   032304       111223
1518  032306
 (1)                001215
 (1)   032306       110517
1519
1520
1521  032310
 (1)                001216
 (1)   032310       000627
 (1)                001217
 (1)   032312       063222
1522  032314
 (1)                001220
 (1)   032314       000403
1523  032316
 (1)                001221
 (1)   032316       060366
```

```
                .SBTTL  TMTI--SEND UNNUMBERED TYPE FIELD
TMTI:   LDMA    IMM,T                           ;ADDRESS OF TYPE FIELD TO MAR
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<T&377>>
        SP      MEMX,SELB,SP6                   ;COPY IT TO SP6
        MICPC=MICPC+1
        <MOVE!SPX!MEMX!SELB!SP6>
        STATE   TMTJ
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<TMTJ-INIT&777/2>>
        ALWAYS  TD3
        MICPC=MICPC+1
        <JUMP!ALCOND!<TD3-INIT&3000*4>!<TD3-INIT&777/2>>
        ;
                .SBTTL  TMTJ--SEND SUB-TYPE FIELD
TMTJ:   LDMA    IMM,ST                          ;ADDRESS OF SUB-TYPE FIELD TO MAR
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<ST&377>>
        STATE   TMTK
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<TMTK-INIT&777/2>>
        ALWAYS  TD3
        MICPC=MICPC+1
        <JUMP!ALCOND!<TD3-INIT&3000*4>!<TD3-INIT&777/2>>
                .SBTTL  TMTK--OUTPUT RESPONSE FIELD (UNNUMB MSG)
        ;
TMTK:   BRWRTE  IMM,3                           ;WRITE A 3 TO BR
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<3>>
        NOP     BR,SUB,SP6                      ;IF TYPE LESS THAN 3
        MICPC=MICPC+1
        <BR!SUB!SP6>
        TSTATE  TMTL
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<TMTL-INIT&777/2>>
        MICPC=MICPC+1
        <MOVE!SPX!BR!SELB!SP2>
        C       TMTLO
        MICPC=MICPC+1
        <JUMP!CCOND!<TMTLO-INIT&3000*4>!<TMTLO-INIT&777/2>>
        ALWAYS  TD2
        MICPC=MICPC+1
        <JUMP!ALCOND!<TD2-INIT&3000*4>!<TD2-INIT&777/2>>
        ;
                .SBTTL  TMTL--UNNUMB MSG NUMBER FIELD
TMTL:   TSTATE  TMTM
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<TMTM-INIT&777/2>>
        MICPC=MICPC+1
        <MOVE!SPX!BR!SELB!SP2>
        BRWRTE  IMM,3
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<3>>
        CMP     BR,SP6                          ;IS MESSAGE REP
        MICPC=MICPC+1
        <SUBTC!BR!SP6>
```

TMTL--UNNUMB MSG NUMBER FIELD

```
1524  032320
 (1)            001222
 (1)  032320    111625          Z       TMTL1                           ;YES
                                MICPC=MICPC+1
                                <JUMP!ZCOND!<TMTL1-INIT&3000*4>!<TMTL1-INIT&777/2>>
1525  032322           TMTLO:   BRWRTE  IMM,0                           ;ADDRESS CONTNAT OF ZERO
 (1)            001223          MICPC=MICPC+1
 (1)  032322    000400          <MOVE!WRTEBR!IMM!<0>>
1527  032324                   ALWAYS  TMTA5
 (1)            001224          MICPC=MICPC+1
 (1)  032324    110424          <JUMP!ALCOND!<TMTA5-INIT&3000*4>!<TMTA5-INIT&777/2>>
1533                            ;
1534  032326           TMTL1:   BRWRTE  BR,SELA!SP12                    ;WRITE A RESPONSE
 (1)            001225          MICPC=MICPC+1
 (1)  032326    060612          <MOVE!WRTEBR!BR!<SELA!SP12>>
1535  032330                   ALWAYS  TMTA5
 (1)            001226          MICPC=MICPC+1
 (1)  032330    110424          <JUMP!ALCOND!<TMTA5-INIT&3000*4>!<TMTA5-INIT&777/2>>
1536
1537                            ;SBTTL  TMTM--UNNUMB MSG--STATION ADDRESS
1538  032332           TMTM:    STATE   TNEOM
 (1)            001227          MICPC=MICPC+1
 (1)  032332    000631          <MOVE!WRTEBR!IMM!<TNEOM-INIT&777/2>>
1539  032334                   ALWAYS  TF2
 (1)            001230          MICPC=MICPC+1
 (1)  032334    110530          <JUMP!ALCOND!<TF2-INIT&3000*4>!<TF2-INIT&777/2>>
1540  032336           TNEOM:   BRWRTE  IMM,2                           ;END OF MESSAGE TO BR
 (1)            001231          MICPC=MICPC+1
 (1)  032336    000402          <MOVE!WRTEBR!IMM!<2>>
1541  032340                   OUTPUT  BR,<SELB!OTMTCO>
 (1)            001232          MICPC=MICPC+1
 (1)  032340    062231          <MOVE!WROUT!BR!<SELB!OTMTCO>>
1542  032342                   OUTPUT  BR,<SELB!TMTDAT>                 ;OUTPUT A GARBAGE CHARACTER
 (1)            001233          MICPC=MICPC+1
 (1)  032342    062230          <MOVE!WROUT!BR!<SELB!TMTDAT>>
1544  032344                   BRWRTE  IMM,4                            ;SET UP LINE HAS GONE IDLE MASK
 (1)            001234          MICPC=MICPC+1
 (1)  032344    000404          <MOVE!WRTEBR!IMM!<4>>
1545  032346                   SPBR    BR,AORB,SP10                     ;UPDATE LINE STATUS WORD
 (1)            001235          MICPC=MICPC+1
 (1)  032346    063710          <MOVE!SPBRX!BR!AORB!SP10>
1547  032350                   BRWRTE  BR,AA!SP10                       ;SHIFT STATUS LEFT
 (1)            001236          MICPC=MICPC+1
 (1)  032350    060530          <MOVE!WRTEBR!BR!<AA!SP10>>
1548  032352                   BR7     TNEOM2                           ;IF HDX SET---BRANCH TO CLEAR OK TO SEND
 (1)            001237          MICPC=MICPC+1
 (1)  032352    113644          <JUMP!BR7CON!<TNEOM2-INIT&3000*4>!<TNEOM2-INIT&777/2>>
1549  032354                   BRWRTE  IMM,376                          ;MASK TO TURN OFF UNNUMB PENDDING
 (1)            001240          MICPC=MICPC+1
 (1)  032354    000776          <MOVE!WRTEBR!IMM!<376>>
1550  032356           TNEOM1:  SP      BR,AANDB,SP10                   ;MASK TO LINE STATUS WORD
 (1)            001241          MICPC=MICPC+1
 (1)  032356    063270          <MOVE!SPX!BR!AANDB!SP10>
1551  032360           TEOM2:   STATE   TMTA
 (1)            001242          MICPC=MICPC+1
 (1)  032360    000400          <MOVE!WRTEBR!IMM!<TMTA-INIT&777/2>>
1552  032362                   ALWAYS  XEXIT
 (1)            001243          MICPC=MICPC+1
```

```
  (1)  032362  110554
1553   032364
  (1)          001244
  (1)  032364  000576
1554   032366
  (1)          001245
  (1)  032366  110641
```

```
        <JUMP!ALCOND!<XEXIT-INIT&3000*4>!<XEXIT-INIT&777/2>>
TNEOM2: BRWRTE  IMM,176                          ;CLEAR OK TO SEND AND UNNUMB PENPENDING
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<176>>
        ALWAYS  TNEOM1
        MICPC=MICPC+1
        <JUMP!ALCOND!<TNEOM1-INIT&3000*4>!<TNEOM1-INIT&777/2>>
```

```
                        .SBTTL  TIMSRV--TIMEOUT ROUTINE--SENDS  REP
                        ;
                        .ENABLE LSB
1559  032370            TIMSRV: BRWRTE  IMM,177                         ;MASK OFF  BR REQ
 (1)        001246              MICPC=MICPC+1
 (1)  032370  000577           <MOVE!WRTEBR!IMM!<177>>
                                                                        ;RESET TIMER---SLICK MOVE
                                                                        ;SINCE TIMER IS RESET BY WRITING
                                                                        ;A 1 AND THE EXPIRATION  LOOKS
                                                                        ;LIKE 1--VOILA
1564  032372                    OUT     BR,<AANDB!OBR>                  ;AND THE BIT ON
 (1)        001247              MICPC=MICPC+1
 (1)  032372  061271           <MOVE!WROUTX!BR!<AANDB!OBR>>
1565  032374                    LDMA    IMM,236
 (1)        001250              MICPC=MICPC+1
 (1)  032374  010236           <MOVE!LDMAR!IMM!<236&377>>
1566  032376                    SP      MEMX,SELB,SP0
 (1)        001251              MICPC=MICPC+1
 (1)  032376  043220           <MOVE!SPX!MEMX!SELB!SP0>
1567  032400                    SPBR    BR,INCA,SP0
 (1)        001252              MICPC=MICPC+1
 (1)  032400  063460           <MOVE!SPBRX!BR!INCA!SP0>
1568  032402                    MEMINC  BR,SELB
 (1)        001253              MICPC=MICPC+1
 (1)  032402  076620           <MOVE!WRMEM!INCMAR!BR!<SELB>>
1569  032404                    C       5$
 (1)        001254              MICPC=MICPC+1
 (1)  032404  111256           <JUMP!CCOND!<5$-INIT&3000*4>!<5$-INIT&777/2>>
1570  032406                    ALWAYS  IDLE1
 (1)        001255              MICPC=MICPC+1
 (1)  032406  100457           <JUMP!ALCOND!<IDLE1-INIT&3000*4>!<IDLE1-INIT&777/2>>
1571  032410            5$:     SP      MEMX,SELB,SP0
 (1)        001256              MICPC=MICPC+1
 (1)  032410  043220           <MOVE!SPX!MEMX!SELB!SP0>
1572  032412                    SPBR    BR,INCA,SP0
 (1)        001257              MICPC=MICPC+1
 (1)  032412  063460           <MOVE!SPBRX!BR!INCA!SP0>
1573  032414                    MEM     BR,SELB
 (1)        001260              MICPC=MICPC+1
 (1)  032414  062620           <MOVE!WRMEM!BR!<SELB>>
1574  032416                    C       10$
 (1)        001261              MICPC=MICPC+1
 (1)  032416  111263           <JUMP!CCOND!<10$-INIT&3000*4>!<10$-INIT&777/2>>
1575  032420                    ALWAYS  IDLE1
 (1)        001262              MICPC=MICPC+1
 (1)  032420  100457           <JUMP!ALCOND!<IDLE1-INIT&3000*4>!<IDLE1-INIT&777/2>>
1576  032422            10$:    MEM     IMM,250
 (1)        001263              MICPC=MICPC+1
 (1)  032422  002650           <MOVE!WRMEM!IMM!<250>>
1577  032424                    SP      BR,INCA,SP14                    ;INCREMENT THE COUNT
 (1)        001264              MICPC=MICPC+1
 (1)  032424  063074           <MOVE!SPX!BR!INCA!SP14>
                                                                        ;NOTE: STATE OF "C" BIT
                                                                        ;MUST BE PRESERVED FROM THIS
                                                                        ;POINT UNTIL IT IS TESTED
1581  032426            TIMEO:
```

```
1582  032426
 (1)            001265
 (1)  032426    060601          BRWRTE  BR,SELA!SP1                     ;READ STATUS BYTE
                                MICPC=MICPC+1
                                <MOVE!WRTEBR!BR!<SELA!SP1>>
1583  032430                    BRO     IDLE
 (1)            001266          MICPC=MICPC+1
 (1)  032430    102047          <JUMP!BROCON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
1584  032432                    BR7     IDLE                             ;IF IN MAINT. MODE DISABLE TIMER
 (1)            001267          MICPC=MICPC+1
 (1)  032432    103447          <JUMP!BR7CON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
1585  032434                    BRWRTE  BR,SELA!SP10                     ;READ LINE STATUS REGISTER
 (1)            001270          MICPC=MICPC+1
 (1)  032434    060610          <MOVE!WRTEBR!BR!<SELA!SP10>>
1586  032436                    BSHFTB                                   ;ROTATE IT RIGHT ONE BIT
 (1)            001271          MICPC=MICPC+1
 (1)  032436    061620          <MOVE!SHFTBR!SELB!BR>
1587  032440                    BR4     BS1                              ;START MODE - RESEND A START
 (1)            001272          MICPC=MICPC+1
 (1)  032440    103164          <JUMP!BR4CON!<BS1-INIT&3000*4>!<BS1-INIT&777/2>>
1588                                                                    ;TEST STATE OF C BIT FROM INC SP14
1589  032442                    C       20$                              ;IF CARRY SET COUNT HAS EXPIRED
 (1)            001273          MICPC=MICPC+1
 (1)  032442    111277          <JUMP!CCOND!<20$-INIT&3000*4>!<20$-INIT&777/2>>
1590  032444                    BRO     TABUPD                           ;NUMBERED MESSAGE IN PROGRESS
 (1)            001274          MICPC=MICPC+1
 (1)  032444    116306          <JUMP!BROCON!<TABUPD-INIT&3000*4>!<TABUPD-INIT&777/2>>
1591  032446                    BR7     TABUPD                           ;UNNUMBMSGIN PROGRESS
 (1)            001275          MICPC=MICPC+1
 (1)  032446    117706          <JUMP!BR7CON!<TABUPD-INIT&3000*4>!<TABUPD-INIT&777/2>>
1592  032450                    ALWAYS  SNDACK                           ;ELSE TRY TO SEND AN ACK
 (1)            001276          MICPC=MICPC+1
 (1)  032450    110737          <JUMP!ALCOND!<SNDACK-INIT&3000*4>!<SNDACK-INIT&777/2>>
1593  032452    
1594  032452    
1595  032452                    BRWRTE  IMM,205                          ;SET OK TO SEND,UNNUMB PEND,AND
 (1)            001277          MICPC=MICPC+1
 (1)  032452    000605          <MOVE!WRTEBR!IMM!<205>>
1596                                                                    ;LINE GONE IDLE
1597  032454                    SP      BR,AORB,SP10
 (1)            001300          MICPC=MICPC+1
 (1)  032454    063310          <MOVE!SPX!BR!AORB!SP10>
1598  032456                    SP      IMM,374,SP14                     ;RESET THE TIMER TICK COUNT
 (1)            001301          MICPC=MICPC+1
 (1)  032456    003374          <MOVE!SPX!IMM!374!SP14>
1599  032460                    BRWRTE  BR,SELA!SP12                     ;GET LAST NUMBER SENT
 (1)            001302          MICPC=MICPC+1
 (1)  032460    060612          <MOVE!WRTEBR!BR!<SELA!SP12>>
1600  032462                    CMP     BR,SP17                          ;COMPARE TO LAST ACKED
 (1)            001303          MICPC=MICPC+1
 (1)  032462    060377          <SUBTC!BR!SP17>
1601  032464                    Z       SNDACK                  ;IF EQ --SEND ACK
 (1)            001304          MICPC=MICPC+1
 (1)  032464    111737          <JUMP!ZCOND!<SNDACK-INIT&3000*4>!<SNDACK-INIT&777/2>>
1602  032466    
 (1)            001305          MICPC=MICPC+1
 (1)  032466    010015          <MOVE!LDMAR!IMM!<REPCS&377>>
1603  032470                    SP      MEMX,SELB,SPO                    ;COPY IT TO SPO

TIME1:
20$:
TIME2:  LDMA    IMM,REPCS                                               ;CUMULATIVE REPS SENT
```

```
  (1)           001306
  (1)  032470   043220          MICPC=MICPC+1
                                <MOVE!SPX!MEMX!SELB!SPO>
1604   032472                   MEM     BR,INCA!SPO                     ;INCREMENT IT
  (1)           001307          MICPC=MICPC+1
  (1)  032472   062460          <MOVE!WRMEM!BR!<INCA!SPO>>
1605   032474                   LDMA    IMM,REPST                       ;ADDRESS DYNAMIC REP COUNTER
  (1)           001310          MICPC=MICPC+1
  (1)  032474   010003          <MOVE!LDMAR!IMM!<REPST&377>>
1606   032476                   BRWRTE  MEMX,SELB                       ;COPY IT TO THE BR
  (1)           001311          MICPC=MICPC+1
  (1)  032476   040620          <MOVE!WRTEBR!MEMX!<SELB>>
1607   032500                   BSHFTB
  (1)           001312          MICPC=MICPC+1
  (1)  032500   061620          <MOVE!SHFTBR!SELB!BR>
1608   032502                   MEM     BR,SELB
  (1)           001313          MICPC=MICPC+1
  (1)  032502   062620          <MOVE!WRMEM!BR!<SELB>>
1609   032504                   BRO     RTHRES
  (1)           001314          MICPC=MICPC+1
  (1)  032504   106371          <JUMP!BROCON!<RTHRES-INIT&3000*4>!<RTHRES-INIT&777/2>>
1610   032506                   LDMA    IMM,T                           ;LOAD ADDRESS OF TYPE FIELD IN UNNUMB SK
  (1)           001315          MICPC=MICPC+1
  (1)  032506   010151          <MOVE!LDMAR!IMM!<T&377>>
1611   032510                   MEMINC  IMM,3                           ;LOAD REP TYPE
  (1)           001316          MICPC=MICPC+1
  (1)  032510   016403          <MOVE!WRMEM!INCMAR!IMM!<3>>
1612   032512                   ALWAYS  SA1
  (1)           001317          MICPC=MICPC+1
  (1)  032512   110741          <JUMP!ALCOND!<SA1-INIT&3000*4>!<SA1-INIT&777/2>>
1613                            .DSABLE LSB
1614                            ;
```

```
1616  032514
 (1)            001320
 (1)  032514    120620  TEOM:   BRWRTE  IBUS,UBBR
                        MICPC=MICPC+1
                        <MOVE!WRTEBR!IBUS!<UBBR>>
1617  032516
 (1)            001321         BRO     NXMERR                          ;NON-EXISTANT MEMORY
 (1)  032516    116051         MICPC=MICPC+1
                        <JUMP!BROCON!<NXMERR-INIT&3000*4>!<NXMERR-INIT&777/2>>
1618  032520                   BRWRTE  IMM,2                           ;EOM TO BR
 (1)            001322         MICPC=MICPC+1
 (1)  032520    000402         <MOVE!WRTEBR!IMM!<2>>
1619  032522                   OUTPUT  BR,<SELB!OTMTCO>                ;WRITE TMTR CONTROL
 (1)            001323         MICPC=MICPC+1
 (1)  032522    062231         <MOVE!WROUT!BR!<SELB!OTMTCO>>
1620  032524                   OUTPUT  BR,<SELB!TMTDAT>                ;WRITE GARBAGE DATA
 (1)            001324         MICPC=MICPC+1
 (1)  032524    062230         <MOVE!WROUT!BR!<SELB!TMTDAT>>
1621  032526                   BRWRTE  BR,SELA!SP1                     ;CKECK FOR BOOT MODE
 (1)            001325         MICPC=MICPC+1
 (1)  032526    060601         <MOVE!WRTEBR!BR!<SELA!SP1>>
1622  032530                   BR7     BTEOM                           ;---IF SET IS MAINT MSG
 (1)            001326         MICPC=MICPC+1
 (1)  032530    113764         <JUMP!BR7CON!<BTEOM-INIT&3000*4>!<BTEOM-INIT&777/2>>
1623  032532    TEOM1:  LDMA    BR,SELA!SP16                            ;ADDRESS LAST TMT LINK
 (1)            001327         MICPC=MICPC+1
 (1)  032532    070216         <MOVE!LDMAR!BR!<SELA!SP16>>
1624  032534                   BRWRTE  MEMX,SELB
 (1)            001330         MICPC=MICPC+1
 (1)  032534    040620         <MOVE!WRTEBR!MEMX!<SELB>>
1625  032536                   BRO     TEOM2
 (1)            001331         MICPC=MICPC+1
 (1)  032536    112242         <JUMP!BROCON!<TEOM2-INIT&3000*4>!<TEOM2-INIT&777/2>>
1626  032540    TEOM3:  BRWRTE  IMM,375                         ;TURN OFF MESSAGE PENDING
 (1)            001332         MICPC=MICPC+1
 (1)  032540    000775         <MOVE!WRTEBR!IMM!<375>>
1627  032542                   SPBR    BR,AANDB,SP10           ;
 (1)            001333         MICPC=MICPC+1
 (1)  032542    063670         <MOVE!SPBRX!BR!AANDB!SP10>
1628  032544                   BRO     TEOM2                           ;IF UNNUMB PENDING--GO AWAY
 (1)            001334         MICPC=MICPC+1
 (1)  032544    112242         <JUMP!BROCON!<TEOM2-INIT&3000*4>!<TEOM2-INIT&777/2>>
1629  032546                   TSTATE  TMTA
 (1)            001335         MICPC=MICPC+1
 (1)  032546    000400         <MOVE!WRTEBR!IMM!<TMTA-INIT&777/2>>
 (1)            001336         MICPC=MICPC+1
 (1)  032550    063222         <MOVE!SPX!BR!SELB!SP2>
1630                     .SBTTL  SNDACK--ROUTINE TO SEND AN ACK
1631  032552    SNDACK: LDMA    IMM,T
 (1)            001337         MICPC=MICPC+1
 (1)  032552    010151         <MOVE!LDMAR!IMM!<T&377>>
1632  032554                   MEMINC  IMM,1
 (1)            001340         MICPC=MICPC+1
 (1)  032554    016401         <MOVE!WRMEM!INCMAR!IMM!<1>>
1633  032556    SA1:    BRWRTE  IMM,5
 (1)            001341         MICPC=MICPC+1
 (1)  032556    000405         <MOVE!WRTEBR!IMM!<5>>
1634  032560    SA2:    MEMINC  IMM,300
 (1)            001342         MICPC=MICPC+1
```

SNDACK--ROUTINE TO SEND AN ACK

```
  (1)  032560  016700
1635   032562
  (1)          001343
  (1)  032562  063310
1636   032564
  (1)          001344
  (1)  032564  100447
1637
1638   032566
  (1)          001345
  (1)  032566  120600
1639   032570
  (1)          001346
  (1)  032570  102047
1640   032572
  (1)          001347
  (1)  032572  070604
1641   032574
  (1)          001350
  (1)  032574  103574
1642   032576
  (1)          001351
  (1)  032576  036400
1643   032600
  (1)          001352
  (1)  032600  036420
1644   032602
  (1)          001353
  (1)  032602  000402
1645   032604
  (1)          001354
  (1)  032604  063004
1646   032606
  (1)          001355
  (1)  032606  023100
1647   032610
  (1)          001356
  (1)  032610  062004
1648   032612
  (1)          001357
  (1)  032612  023120
1649   032614
  (1)          001360
  (1)  032614  062105
1650   032616
  (1)          001361
  (1)  032616  123200
1651   032620
  (1)          001362
  (1)  032620  115166
1652   032622
  (1)          001363
  (1)  032622  110575
```

```
        <MOVE!WRMEM!INCMAR!IMM!<300>>
SA3:    SP      BR,AORB,SP10
        MICPC=MICPC+1
        <MOVE!SPX!BR!AORB!SP10>
        ALWAYS  IDLE
        MICPC=MICPC+1
        <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
        ;
FUDGE:  BRWRTE  IBUS,NPR                ;READ NPR CONTROL
        MICPC=MICPC+1
        <MOVE!WRTEBR!IBUS!<NPR>>
        BRO     IDLE                    ;IF NPR GOING---LEAVE
        MICPC=MICPC+1
        <JUMP!BROCON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
        BRWRTE  BR!LDMAR,SELA!SP4       ;LOAD THE MAR
        MICPC=MICPC+1
        <MOVE!WRTEBR!BR!LDMAR!<SELA!SP4>>
        BR7     BS2                     ;IF SET - READ BACK ALL 200
        MICPC=MICPC+1
        <JUMP!BR7CON!<BS2-INIT&3000*4>!<BS2-INIT&777/2>>
        MEMINC  IBUS,INDAT1             ;OTHERWISE RESTORE TWO BYTES
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IBUS!<INDAT1>>
        MEMINC  IBUS,INDAT2             ;..
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IBUS!<INDAT2>>
        BRWRTE  IMM,2                   ;UPDATE---UNIBUS ADDRESS
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<2>>
        SP      BR,ADD,SP4              ;UPDATE NPR COUNTER
        MICPC=MICPC+1
        <MOVE!SPX!BR!ADD!SP4>
        SP      IBUS,IIBA1,SPO          ;UPDATE ADDRESS LOW
        MICPC=MICPC+1
        <MOVE!SPX!IBUS!IIBA1!SPO>
        OUTPUT  BR,ADD!IBA1
        MICPC=MICPC+1
        <MOVE!WROUT!BR!<ADD!IBA1>>
        SP      IBUS,IIBA2,SPO          ;READ HIGH ADDRESS
        MICPC=MICPC+1
        <MOVE!SPX!IBUS!IIBA2!SPO>
        OUTPUT  BR,AC!IBA2              ;UPDATE HIGH
        MICPC=MICPC+1
        <MOVE!WROUT!BR!<AC!IBA2>>
        SP      IBUS,NPR,SPO            ;READ NPR REGISTER
        MICPC=MICPC+1
        <MOVE!SPX!IBUS!NPR!SPO>
        C       RESEXT                  ;IF CARRY---UPDATE MXT
        MICPC=MICPC+1
        <JUMP!CCOND!<RESEXT-INIT&3000*4>!<RESEXT-INIT&777/2>>
        ALWAYS  TH3X                    ;GO DO ANOTHER NPR
        MICPC=MICPC+1
        <JUMP!ALCOND!<TH3X-INIT&3000*4>!<TH3X-INIT&777/2>>
```

```
1654  032624
 (1)            001364
 (1)  032624    000774
1655  032626
 (1)            001365
 (1)  032626    063270
1656  032630
 (1)            001366
 (1)  032630    063233
1657
1658  032632
 (1)            001367
 (1)  032632    010067
1659  032634
 (1)            001370
 (1)  032634    043220
1661  032636
 (1)            001371
 (1)  032636    000404
1662  032640
 (1)            001372
 (1)  032640    063310
1664  032642
 (1)            001373
 (1)  032642    000400
 (1)            001374
 (1)  032644    063222
1665  032646
 (1)            001375
 (1)  032646    114531
1666
1667  032650
1675  032650
 (1)            001376
 (1)  032650    123200
1676  032652
 (1)            001377
 (1)  032652    000404
1677  032654
 (1)            001400
 (1)  032654    061010
1678  032656
 (1)            001401
 (1)  032656    110566
1679
1680  032660
 (1)            001402
 (1)  032660    010016
1681  032662
 (1)            001403
 (1)  032662    043220
1682  032664
 (1)            001404
 (1)  032664    062460
1683  032666
 (1)            001405
```

SNDACK--ROUTINE TO SEND AN ACK

```
BTEOM:  BRWRTE  IMM,374                 ;MASK FOR CLEAR MSG PENDING
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<374>>
        SP      BR,AANDB,SP10           ;TURN THEM OFF IN LINE STATUS WORD
        MICPC=MICPC+1
        <MOVE!SPX!BR!AANDB!SP10>
        SP      BR,SELB,SP13            ;STORE UNRECOGNIZABLE VALUE INTO SP13
        MICPC=MICPC+1
        <MOVE!SPX!BR!SELB!SP13>
                                        ;SO "RH3" WILL EXIT BACK TO IDLE LOOP
        LDMA    IMM,STC                 ;ADDRESS START OF TMT  CHAIN
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<STC&377>>
        SP      MEMX,SELB,SP0           ;COPY LINK ADDRESS
        MICPC=MICPC+1
        <MOVE!SPX!MEMX!SELB!SP0>
        BRWRTE  IMM,4
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<4>>
        SP      BR,AORB,SP10
        MICPC=MICPC+1
        <MOVE!SPX!BR!AORB!SP10>
        TSTATE  TMTA                    ;CHANGE XMIT STATE TO LINE IS IDLE
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<TMTA-INIT&777/2>>
        MICPC=MICPC+1
        <MOVE!SPX!BR!SELB!SP2>
        ALWAYS  TDON2                   ;POST A DONE
        MICPC=MICPC+1
        <JUMP!ALCOND!<TDON2-INIT&3000*4>!<TDON2-INIT&777/2>>
        ;INCREMENT MXT BITS
HOINCH:
        SP      IBUS,NPR,SP0            ;READ NPR REG IWTH CURRENT MXT BITS
        MICPC=MICPC+1
        <MOVE!SPX!IBUS!NPR!SP0>
        BRWRTE  IMM,4                   ;WRITE BIT TO ADD
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<4>>
        OUT     BR,<ADD!ONPR>           ;TURN ON PROPER MXT BITS
        MICPC=MICPC+1
        <MOVE!WROUTX!BR!<ADD!ONPR>>
        ALWAYS  TH8
        MICPC=MICPC+1
        <JUMP!ALCOND!<TH8-INIT&3000*4>!<TH8-INIT&777/2>>
        .SBTTL  REP HANDLER
REP:    LDMA    IMM,REPCR               ;LOAD MAR ADDRESS WITH POINTER TO REPS RECC
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<REPCR&377>>
        SP      MEMX,SELB,SP0           ;READ NUMBER OF REPS RECD
        MICPC=MICPC+1
        <MOVE!SPX!MEMX!SELB!SP0>
        MEM     DP,<INCA!SP0>           ;INCREMNT REPS RECD
        MICPC=MICPC+1
        <MOVE!WRMEM!DP!<INCA!SP0>>
        LDMA    IMM,T                   ;LOAD ADDRESS OF TYPE FIELD
        MICPC=MICPC+1
```

```
  (1)  032666  010151
1684   032670
  (1)          001406
  (1)  032670  016402
1685   032672
  (1)          001407
  (1)  032672  016703
1686   032674
  (1)          001410
  (1)  032674  114700
1687
1688   032676
  (1)          001411
  (1)  032676  117013
1689   032700
  (1)          001412
  (1)  032700  100447
1690   032702
  (1)          001413
  (1)  032702  000727
1691   032704
  (1)          001414
  (1)  032704  063270
1692   032706
  (1)          001415
  (1)  032706  110737
```

```
        <MOVE!LDMAR!IMM!<T&377>>
        MEMINC  IMM,2                   ; LOAD NAK TYPE
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<2>>
        MEMINC  IMM,303                 ;LOAD REP RESPONSE SUB-TYPE
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<303>>
        ALWAYS  SNAK                    ;SEND AN UNNUMB MSG
        MICPC=MICPC+1
        <JUMP!ALCOND!<SNAK-INIT&3000*4>!<SNAK-INIT&777/2>>
        .
STACK:  BR4     5$                      ;IF START MODE ACCEPT STACK
        MICPC=MICPC+1
        <JUMP!BR4CON!<5$-INIT&3000*4>!<5$-INIT&777/2>>
        ALWAYS  IDLE                    ;IGNORE STACK
        MICPC=MICPC+1
        <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
5$:     BRWRTE  IMM,327                 ;MASK TO CLEAR START
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<327>>
        SP      BR,AANDB,SP10           ;CLEAR START MODE
        MICPC=MICPC+1
        <MOVE!SPX!BR!AANDB!SP10>
        ALWAYS  SNDACK          ;RESET TIMER AND IDLE
        MICPC=MICPC+1
        <JUMP!ALCOND!<SNDACK-INIT&3000*4>!<SNDACK-INIT&777/2>>
```

REP HANDLER

```
1694  032710
1702  032710
 (1)            001416
 (1)  032710    123220
1703  032712
 (1)            001417
 (1)  032712    000515
1704  032714
 (1)            001420
 (1)  032714    063260
1705  032716
 (1)            001421
 (1)  032716    000404
1706  032720
 (1)            001422
 (1)  032720    061011
1707  032722
 (1)            001423
 (1)  032722    104664
1709  032724
 (1)            001424
 (1)  032724    002212
1710  032726
 (1)            001425
 (1)  032726    104601
1712  032730
 (1)            001426
 (1)  032730    010011
1713  032732
 (1)            001427
 (1)  032732    043220
1714  032734
 (1)            001430
 (1)  032734    042460
1715  032736
 (1)            001431
 (1)  032736    060617
1716  032740
 (1)            001432
 (1)  032740    063232
1717  032742
 (1)            001433
 (1)  032742    000406
1718
1719  032744
 (1)            001434
 (1)  032744    010067
1720  032746
 (1)            001435
 (1)  032746    053236
1721  032750
 (1)            001436
 (1)  032750    063310
1722  032752
 (1)            001437
 (1)  032752    003374
```

```
ICBA23:
        SP      IBUS,UBBR,SPO
        MICPC=MICPC+1
        <MOVE!SPX!IBUS!UBBR!SPO>
        BRWRTE  IMM,115
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<115>>
        SP      BR,AANDB,SPO
        MICPC=MICPC+1
        <MOVE!SPX!BR!AANDB!SPO>
        BRWRTE  IMM,4
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<4>>
        OUT     BR,<ADD!OBR>
        MICPC=MICPC+1
        <MOVE!WROUTX!BR!<ADD!OBR>>
        ALWAYS  RK3
        MICPC=MICPC+1
        <JUMP!ALCOND!<RK3-INIT&3000*4>!<RK3-INIT&777/2>>
FLUSH1: OUTPUT  IMM,<200!ORCVCO>                ;FLUSH THE RECVR
        MICPC=MICPC+1
        <MOVE!WROUT!IMM!<200!ORCVCO>>
        ALWAYS  CG1
        MICPC=MICPC+1
        <JUMP!ALCOND!<CG1-INIT&3000*4>!<CG1-INIT&777/2>>
NAK:    LDMA    IMM,NDATR                       ;CUMMULATIVE NAK COUNTER
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<NDATR&377>>
        SP      MEMX,SELB,SPO                   ;READ IT
        MICPC=MICPC+1
        <MOVE!SPX!MEMX!SELB!SPO>
        MEM     MEMX,INCA!SPO                   ;INCREMENT THE COUNTER
        MICPC=MICPC+1
        <MOVE!WRMEM!MEMX!<INCA!SPO>>
        BRWRTE  BR,SELA!SP17                            ;GETLASTMESSAGE ACKED
        MICPC=MICPC+1
        <MOVE!WRTEBR!BR!<SELA!SP17>>
        SP      BR,SELB,SP12                            ;COPY TO CURRENT NUMBER
        MICPC=MICPC+1
        <MOVE!SPX!BR!SELB!SP12>
        BRWRTE  IMM,6                           ;WRITE NUMBERED MSG PENDING
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<6>>
                                                ;AND LINE HAS GONE IDLE
NAK1:   LDMA    IMM,STC                         ;ADDRESS START OF XMIT CHAIN
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<STC&377>>
        SP      MEMX!LDMAR,SELB,SP16                    ;COPY START OF CHAIN TO LAST XMIT
        MICPC=MICPC+1
        <MOVE!SPX!MEMX!LDMAR!SELB!SP16>
        SP      BR,AORB,SP10            ;SET IT IN LINE STATUS WORD
        MICPC=MICPC+1
        <MOVE!SPX!BR!AORB!SP10>
        SP      IMM,374,SP14            ;RESET TIMER COUNT
        MICPC=MICPC+1
        <MOVE!SPX!IMM!374!SP14>
```

```
1723  032754
 (1)              001440
 (1)  032754      113727          BR7     TEOM1
                                  MICPC=MICPC+1
                                  <JUMP!BR7CON!<TEOM1-INIT&3000*4>!<TEOM1-INIT&777/2>>
1724  032756                      BRWRTE  MEMX,SELB
 (1)              001441          MICPC=MICPC+1
 (1)  032756      040620          <MOVE!WRTEBR!MEMX!<SELB>>
1725  032760                      BRO     I1
 (1)              001442          MICPC=MICPC+1
 (1)  032760      102051          <JUMP!BROCON!<I1-INIT&3000*4>!<I1-INIT&777/2>>
1726  032762                      ALWAYS  TEOM3
 (1)              001443          MICPC=MICPC+1
 (1)  032762      110732          <JUMP!ALCOND!<TEOM3-INIT&3000*4>!<TEOM3-INIT&777/2>>
1727  032764  INWAT1: BRWRTE  IMM,200                 ;MASK FOR RDYI
 (1)              001444          MICPC=MICPC+1
 (1)  032764      000600          <MOVE!WRTEBR!IMM!<200>>
1728  032766                      SP      IBUS,INCON,SPO          ;READ INPUT CONTROL CSR
 (1)              001445          MICPC=MICPC+1
 (1)  032766      123000          <MOVE!SPX!IBUS!INCON!SPO>
1729  032770                      OUT     BR,AORB!OINCON          ;SET RDYI
 (1)              001446          MICPC=MICPC+1
 (1)  032770      061300          <MOVE!WROUTX!BR!<AORB!OINCON>>
1730  032772                      PSTATE  IEIWAT                  ;CHANGE STATE TO WAIT FOR IEI OR RQI
 (1)  032772                      MEM     IMM,<<IEIWAT-INIT&777/2>>
 (2)              001447          MICPC=MICPC+1
 (2)  032772      002507          <MOVE!WRMEM!IMM!<<IEIWAT-INIT&777/2>>>
1731  032774                      ALWAYS IDLE
 (1)              001450          MICPC=MICPC+1
 (1)  032774      100447          <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
1732                              ;
1734                              .SBTTL NXMERR ---NON EXISTANT MEMORY HANDLER
1735  032776  NXMERR: LDMA    IMM,<<RTHRS+3>>          ;ADDRESS ERROR LINK
 (1)              001451          MICPC=MICPC+1
 (1)  032776      010177          <MOVE!LDMAR!IMM!<<RTHRS+3>&377>>
1736  033000                      MEMINC  IMM,1
 (1)              001452          MICPC=MICPC+1
 (1)  033000      016401          <MOVE!WRMEM!INCMAR!IMM!<1>>
1737  033002                      MEM     IMM,0                   ;NXM ERROR BIT
 (1)              001453          MICPC=MICPC+1
 (1)  033002      002400          <MOVE!WRMEM!IMM!<0>>
1738  033004                      SP      MEMX,SELB,SP10          ;CLEAR STATUS
 (1)              001454          MICPC=MICPC+1
 (1)  033004      043230          <MOVE!SPX!MEMX!SELB!SP10>
1739  033006                      ALWAYS  RCEXX
 (1)              001455          MICPC=MICPC+1
 (1)  033006      114511          <JUMP!ALCOND!<RCEXX-INIT&3000*4>!<RCEXX-INIT&777/2>>
```

```
1742
1743  033010                    ;FUGITIVE RECEIVE ROUTINES---DON'T FIT IN PAGE
                        RH1:    LDMA    BR,<INCA!SP15>          ;LD MAR WITH ADDR OF BA
 (1)          001456            MICPC=MICPC+1
 (1)  033010  070075            <MOVE!LDMAR!BR!<INCA!SP15>>
1744  033012                    BRWRTE  BR,SELA!SP1             ;READ STATUS BYTE
 (1)          001457            MICPC=MICPC+1
 (1)  033012  060601            <MOVE!WRTEBR!BR!<SELA!SP1>>
1745  033014                    BR1     RH4                     ;NO BUFF ASSIGNED IN MAINT MODE
 (1)          001460            MICPC=MICPC+1
 (1)  033014  116472            <JUMP!BR1CON!<RH4-INIT&3000*4>!<RH4-INIT&777/2>>
1746  033016                    BRWRTE  IMM!INCMAR,77
 (1)          001461            MICPC=MICPC+1
 (1)  033016  014477            <MOVE!WRTEBR!IMM!INCMAR!<77>>
1747  033020                    SP      BR,AANDB,SP5
 (1)          001462            MICPC=MICPC+1
 (1)  033020  063265            <MOVE!SPX!BR!AANDB!SP5>
1748  033022                    SP      BR!INCMAR,SELB,SP0      ;SAVE BYTE COUNT MASK
 (1)          001463            MICPC=MICPC+1
 (1)  033022  077220            <MOVE!SPX!BR!INCMAR!SELB!SP0>
1749  033024                    BRWRTE  MEMX!INCMAR,AANDB!SP0   ;GET HIGH BYTE COUNT BITS
 (1)          001464            MICPC=MICPC+1
 (1)  033024  054660            <MOVE!WRTEBR!MEMX!INCMAR!<AANDB!SP0>>
1750  033026                    CMP     BR,SP5                  ;COMPARE HIGH ORDER BITS OF COUNT
 (1)          001465            MICPC=MICPC+1
 (1)  033026  060365            <SUBTC!BR!SP5>
1751  033030                    Z       RCLOW                   ;IF EQUAL COMPARE LOW ORDER BITS OF COUNT
 (1)          001466            MICPC=MICPC+1
 (1)  033030  115476            <JUMP!ZCOND!<RCLOW-INIT&3000*4>!<RCLOW-INIT&777/2>>
1752  033032                    C       RCFATL                  ;IF CARRY--TOO BIG ERROR
 (1)          001467            MICPC=MICPC+1
 (1)  033032  115100            <JUMP!CCOND!<RCFATL-INIT&3000*4>!<RCFATL-INIT&777/2>>
1753  033034                    BRWRTE  IBUS,IOBA1                      ;READ LOW BYTE OF IN BA
 (1)          001470            MICPC=MICPC+1
 (1)  033034  020540            <MOVE!WRTEBR!IBUS!<IOBA1>>
1754  033036                    BRO     RCVODD                          ;IF SET IS ODD TRANSFER
 (1)          001471            MICPC=MICPC+1
 (1)  033036  116074            <JUMP!BROCON!<RCVODD-INIT&3000*4>!<RCVODD-INIT&777/2>>
1755  033040                    STATE   RCVKE0          ;
                        RH4:    MICPC=MICPC+1
 (1)          001472            <MOVE!WRTEBR!IMM!<RCVKE0-INIT&777/2>>
 (1)  033040  000676            ALWAYS  REXIT
1756  033042                    MICPC=MICPC+1
 (1)          001473            <JUMP!ALCOND!<REXIT-INIT&3000*4>!<REXIT-INIT&777/2>>
 (1)  033042  104425            STATE   RCVKO1          ;
1757  033044              RCVODD: MICPC=MICPC+1
 (1)          001474            <MOVE!WRTEBR!IMM!<RCVKO1-INIT&777/2>>
 (1)  033044  000624            ALWAYS  REXIT
1758  033046                    MICPC=MICPC+1
 (1)          001475            <JUMP!ALCOND!<REXIT-INIT&3000*4>!<REXIT-INIT&777/2>>
 (1)  033046  104425
1759
1760  033050             RCLOW: CMP     MEMX,SP4                        ;COMPARE LOW ORDER BITS OF COUNT
 (1)          001476            MICPC=MICPC+1
 (1)  033050  040364            <SUBTC!MEMX!SP4>
1761  033052                    ALWAYS  RH6
 (1)          001477            MICPC=MICPC+1
 (1)  033052  114467            <JUMP!ALCOND!<RH6-INIT&3000*4>!<RH6-INIT&777/2>>
```

```
1762  033054
 (1)            001500
 (1)  033054    010151
1763  033056
 (1)            001501
 (1)  033056    016402
1764  033060
 (1)            001502
 (1)  033060    002711
1765  033062
 (1)            001503
 (1)  033062    010175
1766  033064
 (1)            001504
 (1)  033064    036540
1767  033066
 (1)            001505
 (1)  033066    036560
1768  033070
 (1)            001506
 (1)  033070    000420
1769  033072
 (1)            001507
 (1)  033072    016400
1770  033074
 (1)            001510
 (1)  033074    062620
1771  033076
 (1)            001511
 (1)  033076    002212
1772  033100
 (1)            001512
 (1)  033100    000404
 (1)            001513
 (1)  033102    063222
1773  033104
 (1)            001514
 (1)  033104    003001
1774  033106
 (1)            001515
 (1)  033106    114662
1775
1776
1777  033110
 (1)            001516
 (1)  033110    060610
1778  033112
 (1)            001517
 (1)  033112    001620
1779  033114
 (1)            001520
 (1)  033114    117123
1780
1781  033116
 (1)            001521
 (1)  033116    000600
```

```
RCFATL: LDMA    IMM,T
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<T&377>>
        MEMINC  IMM,2
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<2>>
        MEM     IMM,311
        MICPC=MICPC+1
        <MOVE!WRMEM!IMM!<311>>
        LDMA    IMM,<<RTHRS+1>>          ;ADDRESS ERROR LINK
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<<RTHRS+1>&377>>
        MEMINC  IBUS,IOBA1
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IBUS!<IOBA1>>
        MEMINC  IBUS,IOBA2
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IBUS!<IOBA2>>
        BRWRTE  IMM,20
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<20>>
RCEXY:  MEMINC  IMM,0
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<0>>
        MEM     BR,SELB
        MICPC=MICPC+1
        <MOVE!WRMEM!BR!<SELB>>
RCEXX:  OUTPUT  IMM,<200!ORCVCO>         ;FLUSH INPUT SILO
        MICPC=MICPC+1
        <MOVE!WROUT!IMM!<200!ORCVCO>>
        TSTATE  TMTA6
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<TMTA6-INIT&777/2>>
        MICPC=MICPC+1
        <MOVE!SPX!BR!SELB!SP2>
        SP      IMM,1,SP1        ;SET INIT MODE IN PORT STATUS WORD
        MICPC=MICPC+1
        <MOVE!SPX!IMM!1!SP1>
        ALWAYS  NTRS1
        MICPC=MICPC+1
        <JUMP!ALCOND!<NTRS1-INIT&3000*4>!<NTRS1-INIT&777/2>>
;
        .SBTTL  START HANDLER
START:  BRWRTE  DP,<SELA!SP10>           ;READ LINE STATUS WORD
        MICPC=MICPC+1
        <MOVE!WRTEBR!DP!<SELA!SP10>>
        BRSHFT                           ;GET START MODE BIT IN POSITION
        MICPC=MICPC+1
        <MOVE!SHFTBR!WRTEBR!SELB>
        BR4     10$                      ;IF IN START MODE SET STACK
        MICPC=MICPC+1
        <JUMP!BR4CON!<10$-INIT&3000*4>!<10$-INIT&777/2>>
                                         ;ELSE SET UP START ERROR
        BRWRTE  IMM,200
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<200>>
```

```
1782  033120
 (1)          001522
 (1)  033120  104741
1783  033122
 (1)          001523
 (1)  033122  010151
1784  033124
 (1)          001524
 (1)  033124  016407
1785  033126
 (1)          001525
 (1)  033126  000411
1786  033130
 (1)          001526
 (1)  033130  110742
```

```
        ALWAYS  RCVM2
        MICPC=MICPC+1
        <JUMP!ALCOND!<RCVM2-INIT&3000*4>!<RCVM2-INIT&777/2>>
10$:    LDMA    IMM,T                   ;SET UP ADDR OF TYPE FIELD
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<T&377>>
        MEMINC  IMM,7                   ;WRITE STACK TYPE
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<7>>
        BRWRTE  IMM,11                  ;SEND THE UNNUMB MESS
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<11>>
        ALWAYS  SA2
        MICPC=MICPC+1
        <JUMP!ALCOND!<SA2-INIT&3000*4>!<SA2-INIT&777/2>>
```

```
1788  033132
 (1)            001527
 (1)  033132    040757
1789  033134
 (1)            001530
 (1)  033134    107573
1790  033136
 (1)            001531
 (1)  033136    070200
1791  033140
 (1)            001532
 (1)  033140    002400
1792  033142
 (1)            001533
 (1)  033142    010067
1793  033144
 (1)            001534
 (1)  033144    002471
1794  033146
 (1)            001535
 (1)  033146    000543
1795  033150
 (1)            001536
 (1)  033150    060360
1796  033152
 (1)            001537
 (1)  033152    115542
1797  033154
 (1)            001540
 (1)  033154    000406
1798  033156
 (1)            001541
 (1)  033156    062400
1799  033160
 (1)            001542
 (1)  033160    010241
1800  033162
 (1)            001543
 (1)  033162    053223
1801
1802  033164
 (1)            001544
 (1)  033164    016600
1803  033166
 (1)            001545
 (1)  033166    062460
1804  033170
 (1)            001546
 (1)  033170    010241
1805  033172
 (1)            001547
 (1)  033172    002642
1806  033174
 (1)            001550
 (1)  033174    000776
1807  033176
```

```
TDON3:  BRWRTE  MEMX,SUB!SP17                   ;COMPARE RESPONSE TO MSG NO
        MICPC=MICPC+1
        <MOVE!WRTEBR!MEMX!<SUB!SP17>>
        BR7     RH3                             ;IF NEGATIVE EXIT
        MICPC=MICPC+1
        <JUMP!BR7CON!<RH3-INIT&3000*4>!<RH3-INIT&777/2>>
TDON2:  LDMA    BR,SELA!SPO                     ;ADDRESS THE TRANSMITLINK
        MICPC=MICPC+1
        <MOVE!LDMAR!BR!<SELA!SPO>>
        MEM     IMM,0                           ;TURN OF ASSIGNEDAND TMTED BITS IN FLAG
        MICPC=MICPC+1
        <MOVE!WRMEM!IMM!<0>>
        LDMA    IMM,STC
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<STC&377>>
        MEM     IMM,TML1                        ;ASSUME WRAPAROUND
        MICPC=MICPC+1
        <MOVE!WRMEM!IMM!<TML1>>
        BRWRTE  IMM,TML8                        ;WRAPAROUND?
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<TML8>>
        CMP     BR,SPO
        MICPC=MICPC+1
        <SUBTC!BR!SPO>
        Z       TDON4                           ;YES
        MICPC=MICPC+1
        <JUMP!ZCOND!<TDON4-INIT&3000*4>!<TDON4-INIT&777/2>>
        BRWRTE  IMM,6                           ;OFFSET FOR NEXT TMT LINK
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<6>>
        MEM     BR,ADD!SPO                      ;UPDATE THE POINTER
        MICPC=MICPC+1
        <MOVE!WRMEM!BR!<ADD!SPO>>
TDON4:  LDMA    IMM,NXTSP                       ;ADDRESS DONE LINK
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<NXTSP&377>>
        LDMA    MEMX,SELB!SPX!SP3               ;ADDRESS THE LINK,COPYING
        MICPC=MICPC+1
        <MOVE!LDMAR!MEMX!<SELB!SPX!SP3>>

                                                ;ITS ADDRESS TO SPO
MEMINC  IMM,200                                 ;WRITE THE INTERRUPT TYPE
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<200>>
        MEM     BR,INCA!SPO                     ;COPY ACTUAL LINK ADDRESS
        MICPC=MICPC+1
        <MOVE!WRMEM!BR!<INCA!SPO>>
        LDMA    IMM,NXTSP                       ;ADDRESS PTR INT STACK
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<NXTSP&377>>
        MEM     IMM,INTSTK                      ;ASSUME WRAP AROUND
        MICPC=MICPC+1
        <MOVE!WRMEM!IMM!<INTSTK>>
        BRWRTE  IMM,<<MMEND-2>>                 ;ADDRESS ENDOFINT STACK
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<<MMEND-2>>>
        CMP     BR,SP3                          ;WRAPAROUND?
```

```
  (1)           001551                MICPC=MICPC+1
  (1)   033176  060363                <SUBTC!BR!SP3>
 1808   033200                        Z       TDON40                          ;YES---BRANCH
  (1)           001552                MICPC=MICPC+1
  (1)   033200  115555                <JUMP!ZCOND!<TDON40-INIT&3000*4>!<TDON40-INIT&777/2>>
 1809   033202                        BRWRTE  IMM,2                           ;OFFSET TO NEXT PAIR
  (1)           001553                MICPC=MICPC+1
  (1)   033202  000402                <MOVE!WRTEBR!IMM!<2>>
 1810   033204                        MEM     BR,ADD!SP3                      ;UPDATE POINTER
  (1)           001554                MICPC=MICPC+1
  (1)   033204  062403                <MOVE!WRMEM!BR!<ADD!SP3>>
 1811   033206               TDON40:  BRWRTE  IMM,20                          ;WRITE INTERUPT PEMDING
  (1)           001555                MICPC=MICPC+1
  (1)   033206  000420                <MOVE!WRTEBR!IMM!<20>>
 1812   033210                        SP      BR,AORB,SP1                     ;IN PORT STATUS WORD
  (1)           001556                MICPC=MICPC+1
  (1)   033210  063301                <MOVE!SPX!BR!AORB!SP1>
 1813
 1814   033212               TDON1:   LDMA    IMM,ISP17                       ;GET LAST ACKED
  (1)           001557                MICPC=MICPC+1
  (1)   033212  010153                <MOVE!LDMAR!IMM!<ISP17&377>>
 1815   033214                        SP      MEMX,SELB,SP17                  ;STORE IT IN SP17
  (1)           001560                MICPC=MICPC+1
  (1)   033214  043237                <MOVE!SPX!MEMX!SELB!SP17>
 1816   033216                        LDMA    IMM,STC                         ;GET START OF TMT CHAIN
  (1)           001561                MICPC=MICPC+1
  (1)   033216  010067                <MOVE!LDMAR!IMM!<STC&377>>
 1817   033220                        LDMA    MEMX,SELB!SPBRX!SP0             ;ADDRESS THE LINK
  (1)           001562                MICPC=MICPC+1
  (1)   033220  053620                <MOVE!LDMAR!MEMX!<SELB!SPBRX!SP0>>
 1818   033222                        BRWRTE  MEMX!INCMAR,SELB                ;GET THE FLAGS
  (1)           001563                MICPC=MICPC+1
  (1)   033222  054620                <MOVE!WRTEBR!MEMX!INCMAR!<SELB>>
 1819   033224                        BR1     TDON3                           ;IF BUFFER ASSIGNED PROCEED
  (1)           001564                MICPC=MICPC+1
  (1)   033224  116527                <JUMP!BR1CON!<TDON3-INIT&3000*4>!<TDON3-INIT&777/2>>
 1820   033226                        ALWAYS  RH3                             ;ELSE---EXIT
  (1)           001565                MICPC=MICPC+1
  (1)   033226  104573                <JUMP!ALCOND!<RH3-INIT&3000*4>!<RH3-INIT&777/2>>
 1821
 1822                                 .
 1823   033230               RESEXT:  BRWRTE  IMM,4                   ;ADD TO MXT BITS
  (1)           001566                MICPC=MICPC+1
  (1)   033230  000404                <MOVE!WRTEBR!IMM!<4>>
 1824   033232                        SP      BR,ADD,SP0
  (1)           001567                MICPC=MICPC+1
  (1)   033232  063000                <MOVE!SPX!BR!ADD!SP0>
 1825   033234                        ALWAYS  TH3X
  (1)           001570                MICPC=MICPC+1
  (1)   033234  110575                <JUMP!ALCOND!<TH3X-INIT&3000*4>!<TH3X-INIT&777/2>>
```

```
1827
1828                                    ;
1829                                    ; INPUTS:
1830                                    ;       SPO = RECEIVE CHARACTER
1831  033236                            PASWRD: SP      IBUS,LNOSW,SP13         ;READ PASSWD SWITCH
 (1)            001571                          MICPC=MICPC+1
 (1)  033236    023333                          <MOVE!SPX!IBUS!LNOSW!SP13>
1832  033240                                    Z       10$                     ;IF ALL ONES NO RLD ENABLED
 (1)            001572                          MICPC=MICPC+1
 (1)  033240    115576                          <JUMP!ZCOND!<10$-INIT&3000*4>!<10$-INIT&777/2>>
1833  033242                                    BRWRTE  IMM,6                   ;CHECK FOR ENTER MOP MODE
 (1)            001573                          MICPC=MICPC+1
 (1)  033242    000406                          <MOVE!WRTEBR!IMM!<6>>
1834  033244                                    CMP     BR,SPO
 (1)            001574                          MICPC=MICPC+1
 (1)  033244    060360                          <SUBTC!BR!SPO>
1835  033246                                    Z       20$                     ;IF EQUAL   ENTER MOP
 (1)            001575                          MICPC=MICPC+1
 (1)  033246    115603                          <JUMP!ZCOND!<20$-INIT&3000*4>!<20$-INIT&777/2>>
1836  033250                            10$:    BRWRTE  BR,SELA!SP1             ;READ STATUS BYTE
 (1)            001576                          MICPC=MICPC+1
 (1)  033250    060601                          <MOVE!WRTEBR!BR!<SELA!SP1>>
1837  033252                                    BR1     RHX                     ;MESSAGE WITH NO BUFFER ASSIGNED
 (1)            001577                          MICPC=MICPC+1
 (1)  033252    106746                          <JUMP!BR1CON!<RHX-INIT&3000*4>!<RHX-INIT&777/2>>
1838  033254                                    BRWRTE  BR,AA!SP1
 (1)            001600                          MICPC=MICPC+1
 (1)  033254    060521                          <MOVE!WRTEBR!BR!<AA!SP1>>
1839  033256                                    BR4     RCVMO                   ;DLE RECEIVED IN NORMAL MODE
 (1)            001601                          MICPC=MICPC+1
 (1)  033256    107340                          <JUMP!BR4CON!<RCVMO-INIT&3000*4>!<RCVMO-INIT&777/2>>
1840  033260                                    ALWAYS  RK3                     ;HANDLE MAINT MODE MESSAGE
 (1)            001602                          MICPC=MICPC+1
 (1)  033260    104664                          <JUMP!ALCOND!<RK3-INIT&3000*4>!<RK3-INIT&777/2>>
1841  033262                            20$:    SP      BR,DECA,SP4             ;COUNT FOR NUMB OF COMPARES
 (1)            001603                          MICPC=MICPC+1
 (1)  033262    063164                          <MOVE!SPX!BR!DECA!SP4>
1842  033264                                    STATE   EM2
 (1)            001604                          MICPC=MICPC+1
 (1)  033264    000743                          <MOVE!WRTEBR!IMM!<EM2-INIT&777/2>>
1843  033266                                    ALWAYS  REXIT
 (1)            001605                          MICPC=MICPC+1
 (1)  033266    104425                          <JUMP!ALCOND!<REXIT-INIT&3000*4>!<REXIT-INIT&777/2>>
1844
1845                                    ;       .ENABL  LSB
1846                                    ;
1847  033270                            RCVM1:  LDMA    IMM,NAKST               ;RESET NAKS SENT
 (1)            001606                          MICPC=MICPC+1
 (1)  033270    010001                          <MOVE!LDMAR!IMM!<NAKST&377>>
1848  033272                                    MEM     IMM,1                   ;..
 (1)            001607                          MICPC=MICPC+1
 (1)  033272    002401                          <MOVE!WRMEM!IMM!<1>>
1849  033274                                    LDMA    IMM,BC                  ;ADDRESS ORIGINAL RECV BYTE COUNT
 (1)            001610                          MICPC=MICPC+1
 (1)  033274    010167                          <MOVE!LDMAR!IMM!<BC&377>>
1850  033276                                    SP      MEMX!INCMAR,SELB,SP4    ;MOVE BYTE COUNT TO SP4
```

```
                001611
 (1)
 (1)   033276   057224
                        MICPC=MICPC+1
                        <MOVE!SPX!MEMX!INCMAR!SELB!SP4>
1851   033300           SP      MEMX!INCMAR,SELB,SP5    ;AND SP5
                001612  MICPC=MICPC+1
 (1)
 (1)   033300   057225  <MOVE!SPX!MEMX!INCMAR!SELB!SP5>
1852   033302           MEM     BR,DECA!SP11                    ;COPY SP11 TO MEMORY
                001613  MICPC=MICPC+1
 (1)
 (1)   033302   062571  <MOVE!WRMEM!BR!<DECA!SP11>>
1853   033304           LDMA    IMM,NXTSP
                001614  MICPC=MICPC+1
 (1)
 (1)   033304   010241  <MOVE!LDMAR!IMM!<NXTSP&377>>
1854   033306           SP      MEMX!LDMAR,SELB,SP3             ;COPY TO SP3
                001615  MICPC=MICPC+1
 (1)
 (1)   033306   053223  <MOVE!SPX!MEMX!LDMAR!SELB!SP3>
1855   033310           MEMINC  IMM,204                 ;RECEIVE DONE IMAGE
                001616  MICPC=MICPC+1
 (1)
 (1)   033310   016604  <MOVE!WRMEM!INCMAR!IMM!<204>>
1856   033312           MEM     BR!LDMAR,SELA!SP15              ;COPY LINK ADDRESS TO NEXT INTER
                001617  MICPC=MICPC+1
 (1)
 (1)   033312   072615  <MOVE!WRMEM!BR!LDMAR!<SELA!SP15>>
1857   033314           MEMINC  IMM,0                   ;ZERO THE FLAGS
                001620  MICPC=MICPC+1
 (1)
 (1)   033314   016400  <MOVE!WRMEM!INCMAR!IMM!<0>>
1858   033316           SP      IMM!INCMAR,SP0,300      ;WRITE A 300 TO SP0
                001621  MICPC=MICPC+1
 (1)
 (1)   033316   017300  <MOVE!SPX!IMM!INCMAR!SP0!300>
1859   033320           BRWRTE  IMM!INCMAR,2            ;PREPARE TO ADDRESS NEXT
                001622  MICPC=MICPC+1
 (1)
 (1)   033320   014402  <MOVE!WRTEBR!IMM!INCMAR!<2>>
1860                                                    ;INTERRUPT STACK AND INCREMENT
1861                                                    ;THE MAR
1862   033322           MEM     MEMX,AANDB!SP0          ;MASK OFF ORIGINAL HIGH BYTE
                001623  MICPC=MICPC+1
 (1)
 (1)   033322   042660  <MOVE!WRMEM!MEMX!<AANDB!SP0>>
1863                                                    ;OF COUNT SAVING EXTENDED MEM BITS
1864   033324           MEMINC  MEMX,AORB!SP5           ;COPY TO MEMORY LINK
                001624  MICPC=MICPC+1
 (1)
 (1)   033324   056705  <MOVE!WRMEM!INCMAR!MEMX!<AORB!SP5>>
1865   033326           MEMINC  BR,SELA!SP4
                001625  MICPC=MICPC+1
 (1)
 (1)   033326   076604  <MOVE!WRMEM!INCMAR!BR!<SELA!SP4>>
1866   033330           LDMA    IMM,NXTSP               ;ADDRESS NEXT INT STACK
                001626  MICPC=MICPC+1
 (1)
 (1)   033330   010241  <MOVE!LDMAR!IMM!<NXTSP&377>>
1867   033332           MEM     BR,ADD!SP3
                001627  MICPC=MICPC+1
 (1)
 (1)   033332   062403  <MOVE!WRMEM!BR!<ADD!SP3>>
1868   033334           BRWRTE  IMM,<<MMEND-2>>         ;ADDRESSEND OF INT STACK
                001630  MICPC=MICPC+1
 (1)
 (1)   033334   000776  <MOVE!WRTEBR!IMM!<<MMEND-2>>>
1869   033336           CMP     BR,SP3                  ;WRAP AROUND
                001631  MICPC=MICPC+1
 (1)
 (1)   033336   060363  <SUBTC!BR!SP3>
1870   033340           Z       40$                     ;IF YES-- BRANCH
                001632  MICPC=MICPC+1
 (1)
 (1)   033340   115646  <JUMP!ZCOND!<40$-INIT&3000*4>!<40$-INIT&777/2>>
```

```
 1871  033342
 1872  033342
  (1)              001633
  (1)  033342      000462
 1873  033344
  (1)              001634
  (1)  033344      060375
 1874  033346
  (1)              001635
  (1)  033346      115650
 1875  033350
  (1)              001636
  (1)  033350      000405
 1876  033352
  (1)              001637
  (1)  033352      063015
 1877  033354
  (1)              001640
  (1)  033354      000420
 1878  033356
  (1)              001641
  (1)  033356      063301
 1880  033360
  (1)              001642
  (1)  033360      060610
 1881  033362
  (1)              001643
  (1)  033362      107015
```

```
20$:
        BRWRTE  IMM,RCL7
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<RCL7>>
        CMP     BR,SP15
        MICPC=MICPC+1
        <SUBTC!BR!SP15>
        Z       50$
        MICPC=MICPC+1
        <JUMP!ZCOND!<50$-INIT&3000*4>!<50$-INIT&777/2>>
        BRWRTE  IMM,5
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<5>>
30$:    SP      BR,ADD,SP15
        MICPC=MICPC+1
        <MOVE!SPX!BR!ADD!SP15>
        BRWRTE  IMM,20                          ;MASK FOR INTERUPT PENDING
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<20>>
        SP      DP,AORB,SP1                     ;UPDATE PORT STATUS WORD
        MICPC=MICPC+1
        <MOVE!SPX!DP!AORB!SP1>
        BRWRTE  DP,<SELA!SP10>                  ;READ LINE STATUS WORD
        MICPC=MICPC+1
        <MOVE!WRTEBR!DP!<SELA!SP10>>
        BR4     FLUSH                           ;IF CLEAR ACTIVE SET---FLUSH
        MICPC=MICPC+1
        <JUMP!BR4CON!<FLUSH-INIT&3000*4>!<FLUSH-INIT&777/2>>
```

```
1883  033364
 (1)            001644
 (1)  033364    000400
1884  033366
 (1)            001645
 (1)  033366    104425
1889  033370
 (1)            001646
 (1)  033370    002642
1890  033372
 (1)            001647
 (1)  033372    114633
1891  033374
 (1)            001650
 (1)  033374    000742
1892  033376
 (1)            001651
 (1)  033376    114637
1893
1894  033400
 (1)            001652
 (1)  033400    010152
1895  033402
 (1)            001653
 (1)  033402    043620
1896  033404
 (1)            001654
 (1)  033404    060400
1897  033406
 (1)            001655
 (1)  033406    103361
1898  033410
 (1)            001656
 (1)  033410    000401
1899  033412
1900  033412
 (1)            001657
 (1)  033412    010177
1901  033414
 (1)            001660
 (1)  033414    016400
1902  033416
 (1)            001661
 (1)  033416    062620
1903  033420
 (1)            001662
 (1)  033420    010241
1904  033422
 (1)            001663
 (1)  033422    053220
1905  033424
 (1)            001664
 (1)  033424    016601
1906  033426
 (1)            001665
 (1)  033426    002574
```

```
RM1:     STATE   RCVA
         MICPC=MICPC+1
         <MOVE!WRTEBR!IMM!<RCVA-INIT&777/2>>
         ALWAYS  REXIT
         MICPC=MICPC+1
         <JUMP!ALCOND!<REXIT-INIT&3000*4>!<REXIT-INIT&777/2>>
40$:     MEM     IMM,INTSTK              ;POINT TO START OF INTERRUPT STACK
         MICPC=MICPC+1
         <MOVE!WRMEM!IMM!<INTSTK>>
         ALWAYS  20$
         MICPC=MICPC+1
         <JUMP!ALCOND!<20$-INIT&3000*4>!<20$-INIT&777/2>>
50$:     BRWRTE  IMM,<<RCL1-RCL7>&377>   ;POINT TO START OF QUEUE
         MICPC=MICPC+1
         <MOVE!WRTEBR!IMM!<<RCL1-RCL7>&377>>
         ALWAYS  30$
         MICPC=MICPC+1
         <JUMP!ALCOND!<30$-INIT&3000*4>!<30$-INIT&777/2>>
         .DSABL  LSB
NTHRES:  LDMA    IMM,ST
         MICPC=MICPC+1
         <MOVE!LDMAR!IMM!<ST&377>>
         SPBR    MEMX,SELB,SPO
         MICPC=MICPC+1
         <MOVE!SPBRX!MEMX!SELB!SPO>
         BRWRTE  BR,ADD!SPO                      ;SHIFT LEFT
         MICPC=MICPC+1
         <MOVE!WRTEBR!BR!<ADD!SPO>>
         BR4     OVRRUN
         MICPC=MICPC+1
         <JUMP!BR4CON!<OVRRUN-INIT&3000*4>!<OVRRUN-INIT&777/2>>
         BRWRTE  IMM,1
         MICPC=MICPC+1
         <MOVE!WRTEBR!IMM!<1>>
ERRXX:
NTRSO:   LDMA    IMM,<<RTHRS+3>>
         MICPC=MICPC+1
         <MOVE!LDMAR!IMM!<<RTHRS+3>&377>>
         MEMINC  IMM,0
         MICPC=MICPC+1
         <MOVE!WRMEM!INCMAR!IMM!<0>>
         MEM     BR,SELB
         MICPC=MICPC+1
         <MOVE!WRMEM!BR!<SELB>>
NTRS1:   LDMA    IMM,NXTSP
         MICPC=MICPC+1
         <MOVE!LDMAR!IMM!<NXTSP&377>>
         LDMA    MEMX,SELB!SPX!SPO
         MICPC=MICPC+1
         <MOVE!LDMAR!MEMX!<SELB!SPX!SPO>>
         MEMINC  IMM,201
         MICPC=MICPC+1
         <MOVE!WRMEM!INCMAR!IMM!<201>>
         MEM     IMM,<<RTHRS>>
         MICPC=MICPC+1
         <MOVE!WRMEM!IMM!<<RTHRS>>>
```

```
1907  033430
 (1)          001666
 (1)  033430  010241          LDMA    IMM,NXTSP
                              MICPC=MICPC+1
                              <MOVE!LDMAR!IMM!<NXTSP&377>>
1908  033432
 (1)          001667
 (1)  033432  002642          MEM     IMM,INTSTK                    ;ASSUME QUEUE WRAP AROUND
                              MICPC=MICPC+1
                              <MOVE!WRMEM!IMM!<INTSTK>>
1909  033434
 (1)          001670
 (1)  033434  000776          BRWRTE  IMM,<<MMEND-2>>
                              MICPC=MICPC+1
                              <MOVE!WRTEBR!IMM!<<MMEND-2>>>
1910  033436
 (1)          001671
 (1)  033436  060360          CMP     BR,SPO
                              MICPC=MICPC+1
                              <SUBTC!BR!SPO>
1911  033440
 (1)          001672
 (1)  033440  115675          Z       NTRS2                         ;IT DID WRAP AROUND
                              MICPC=MICPC+1
                              <JUMP!ZCOND!<NTRS2-INIT&3000*4>!<NTRS2-INIT&777/2>>
1912  033442
 (1)          001673
 (1)  033442  000402          BRWRTE  IMM,2                         ;OFFSET TO NEXT PAIR
                              MICPC=MICPC+1
                              <MOVE!WRTEBR!IMM!<2>>
1913  033444
 (1)          001674
 (1)  033444  062400          MEM     BR,ADD!SPO                    ;UPDATE QUEUE POINTER
                              MICPC=MICPC+1
                              <MOVE!WRMEM!BR!<ADD!SPO>>
1914  033446          NTRS2:  BRWRTE  IMM,20
 (1)          001675          MICPC=MICPC+1
 (1)  033446  000420          <MOVE!WRTEBR!IMM!<20>>
1915  033450                  SPBR    BR,AORB,SP1
 (1)          001676          MICPC=MICPC+1
 (1)  033450  063701          <MOVE!SPBRX!BR!AORB!SP1>
1916  033452                  BRO     TAB1                  ;FLAGGED BY ERROR TYPE
 (1)          001677          MICPC=MICPC+1
 (1)  033452  116311          <JUMP!BROCON!<TAB1-INIT&3000*4>!<TAB1-INIT&777/2>>
1917  033454          SNAK:   LDMA    IMM,ISP11
 (1)          001700          MICPC=MICPC+1
 (1)  033454  010171          <MOVE!LDMAR!IMM!<ISP11&377>>
1918  033456                  SP      MEMX,SELB,SP11
 (1)          001701          MICPC=MICPC+1
 (1)  033456  043231          <MOVE!SPX!MEMX!SELB!SP11>
1919  033460                  SP      BR,INCA,SP11                  ;INCREMENT MSG EXPECTED
 (1)          001702          MICPC=MICPC+1
 (1)  033460  063071          <MOVE!SPX!BR!INCA!SP11>
1920  033462          SNAK1:  BRWRTE  IMM,1                         ;UNNUMB PENING BIT TO BR
 (1)          001703          MICPC=MICPC+1
 (1)  033462  000401          <MOVE!WRTEBR!IMM!<1>>
1921  033464          SNAK2:  SP      BR,AORB,SP10                  ;UPDATE LINE STATUS WORD
 (1)          001704          MICPC=MICPC+1
 (1)  033464  063310          <MOVE!SPX!BR!AORB!SP10>
1922  033466                  ALWAYS  FLUSH
 (1)          001705          MICPC=MICPC+1
 (1)  033466  104415          <JUMP!ALCOND!<FLUSH-INIT&3000*4>!<FLUSH-INIT&777/2>>
```

START HANDLER

```
1924  033470
 (1)            001706
 (1)  033470    023640
1925  033472
 (1)            001707
 (1)  033472    060400
1926  033474
 (1)            001710
 (1)  033474    103447
1927  033476
 (1)            001711
 (1)  033476    010154
1928  033500
 (1)            001712
 (1)  033500    000442
1929  033502
 (1)            001713
 (1)  033502    076670
1930  033504
 (1)            001714
 (1)  033504    076611
1931  033506
 (1)            001715
 (1)  033506    076615
1932  033510
 (1)            001716
 (1)  033510    076617
1933
1934  033512
 (1)            001717
 (1)  033512    000461
1935  033514
 (1)            001720
 (1)  033514    010210
1936  033516
 (1)  033516
 (2)            001721
 (2)  033516    002756
1937  033520
 (1)            001722
 (1)  033520    003004
1938
1939
1940  033522
 (1)            001723
 (1)  033522    010017
1941  033524
 (1)            001724
 (1)  033524    104443
```

```
TABUPD: SPBR    IBUS,RCVCON,SPO          ;READ RECEIVER CONTROL REG
        MICPC=MICPC+1
        <MOVE!SPBRX!IBUS!RCVCON!SPO>
        BRWRTE  BR,ADD!SPO               ;SHIFT LEFT
        MICPC=MICPC+1
        <MOVE!WRTEBR!BR!<ADD!SPO>>
        BR7     IDLE                     ;RECEIVE ACTIVE--IDLE
        MICPC=MICPC+1
        <JUMP!BR7CON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
TAB1:   LDMA    IMM,IMG10
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<IMG10&377>>
        BRWRTE  IMM,42
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<42>>
        MEMINC  BR,AANDB!SP10            ;SAVE BITS 1 AND 5 OF SP10
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!BR!<AANDB!SP10>>
        MEMINC  BR,SELA!SP11             ;SAVE SP11
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!BR!<SELA!SP11>>
        MEMINC  BR,SELA!SP15             ;SAVE SP15
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!BR!<SELA!SP15>>
        MEMINC  BR,SELA!SP17             ;SAVE SP17
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!BR!<SELA!SP17>>
                                         ;MAR NOW POINTS TO BASE
        STATE   RB2             ;DO NOT CHANGE BR UNTIL RB0
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<RB2-INIT&777/2>>
        LDMA    IMM,TABST        ;POINT TO TABLE UPDATE STATE
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<TABST&377>>
        PSTATE  TBU1                     ;NEW PORT STATE ADDRESS
        MEM     IMM,<<TBU1-INIT&777/2>>
        MICPC=MICPC+1
        <MOVE!WRMEM!IMM!<<TBU1-INIT&777/2>>>
        SP      IMM,4,SP4                ;INITIALIZE COUNT
        MICPC=MICPC+1
        <MOVE!SPX!IMM!4!SP4>
                                         ;NOTE: FIRST 6 RAM LOCATIONS ARE NOT WRITTEN
                                         ;TO CORE TABLE.
        LDMA    IMM,BASE
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<BASE&377>>
        ALWAYS  RB0
        MICPC=MICPC+1
        <JUMP!ALCOND!<RB0-INIT&3000*4>!<RB0-INIT&777/2>>
```

START HANDLER

```
1943  033526
  (1)           001725
  (1)  033526   000402
1944  033530
  (1)           001726
  (1)  033530   063004
1945  033532
  (1)           001727
  (1)  033532   023140
1946  033534
  (1)           001730
  (1)  033534   062006
1947  033536
  (1)           001731
  (1)  033536   023160
1948  033540
  (1)           001732
  (1)  033540   062107
1949  033542
  (1)           001733
  (1)  033542   101271
1950  033544
  (1)           001734
  (1)  033544   060601
1951  033546
  (1)           001735
  (1)  033546   116340
1952
1953  033550
  (1)           001736
  (1)  033550   070604
1954  033552
  (1)           001737
  (1)  033552   117346
1955  033554
  (1)           001740
  (1)  033554   070604
1956  033556
  (1)           001741
  (1)  033556   117746
1957  033560
  (1)           001742
  (1)  033560   056222
1958  033562
  (1)           001743
  (1)  033562   056223
1959  033564
  (1)           001744
  (1)  033564   123200
1960  033566
  (1)           001745
  (1)  033566   104640
1961  033570
  (1)           001746
  (1)  033570   010210
1962  033572
```

```
EC2:    BRWRTE  IMM,2                           ;INCREMENT COUNT/TEST
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<2>>
        SP      BR,ADD,SP4
        MICPC=MICPC+1
        <MOVE!SPX!BR!ADD!SP4>
        SP      IBUS,IOBA1,SPO                  ;POINT TO NEXT ADDRESS
        MICPC=MICPC+1
        <MOVE!SPX!IBUS!IOBA1!SPO>
        OUTPUT  BR,ADD!OBA1
        MICPC=MICPC+1
        <MOVE!WROUT!BR!<ADD!OBA1>>
        SP      IBUS,IOBA2,SPO
        MICPC=MICPC+1
        <MOVE!SPX!IBUS!IOBA2!SPO>
        OUTPUT  BR,AC!OBA2
        MICPC=MICPC+1
        <MOVE!WROUT!BR!<AC!OBA2>>
        C       TABMXT
        MICPC=MICPC+1
        <JUMP!CCOND!<TABMXT-INIT&3000*4>!<TABMXT-INIT&777/2>>
ECX:    BRWRTE  BR,SELA!SP1                     ;READ PORT STATUS
        MICPC=MICPC+1
        <MOVE!WRTEBR!BR!<SELA!SP1>>
        BRO     30$                             ;INIT MODE, WRITE OUT 200 BYTES
        MICPC=MICPC+1
        <JUMP!BROCON!<30$-INIT&3000*4>!<30$-INIT&777/2>>
                                                ;OTHERWISE ONLY WRITE OUT ERROR COUNTERS
        BRWRTE  BR!LDMAR,SELA!SP4               ;READ COUNTER
        MICPC=MICPC+1
        <MOVE!WRTEBR!BR!LDMAR!<SELA!SP4>>
        BR4     20$                             ;ALL DONE
        MICPC=MICPC+1
        <JUMP!BR4CON!<20$-INIT&3000*4>!<20$-INIT&777/2>>
30$:    BRWRTE  BR!LDMAR,SELA!SP4               ;READ CTR
        MICPC=MICPC+1
        <MOVE!WRTEBR!BR!LDMAR!<SELA!SP4>>
        BR7     20$                             ;ALL DONE
        MICPC=MICPC+1
        <JUMP!BR7CON!<20$-INIT&3000*4>!<20$-INIT&777/2>>
        OUTPUT  MEMX!INCMAR,SELB!OUTDA1         ;STORE COUNTS OF ERRORS
        MICPC=MICPC+1
        <MOVE!WROUT!MEMX!INCMAR!<SELB!OUTDA1>>
        OUTPUT  MEMX!INCMAR,SELB!OUTDA2
        MICPC=MICPC+1
        <MOVE!WROUT!MEMX!INCMAR!<SELB!OUTDA2>>
        SP      IBUS,NPR,SPO
        MICPC=MICPC+1
        <MOVE!SPX!IBUS!NPR!SPO>
        ALWAYS  RK8
        MICPC=MICPC+1
        <JUMP!ALCOND!<RK8-INIT&3000*4>!<RK8-INIT&777/2>>
20$:    LDMA    IMM,TABST
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<TABST&377>>
        PSTATE  I3
```

```
  (1)  033572
  (2)          001747
  (2)  033572  002455
1964   033574
  (1)          001750
  (1)  033574  114644
1976   033576
  (1)          001751
  (1)  033576  000600
1977   033600
  (1)          001752
  (1)  033600  063301
1978   033602
  (1)          001753
  (1)  033602  000604
1979   033604
  (1)          001754
  (1)  033604  063230
1980   033606
  (1)          001755
  (1)  033606  000400
  (1)          001756
  (1)  033610  063222
1981   033612
  (1)          001757
  (1)  033612  100663
1983   033614
  (1)          001760
  (1)  033614  060601
1984   033616
  (1)          001761
  (1)  033616  103746
1985   033620
  (1)          001762
  (1)  033620  000610
1986   033622
  (1)          001763
  (1)  033622  063301
1987   033624
  (1)          001764
  (1)  033624  100746
1995           000001

. ABS.  033626      000

 ERRORS DETECTED:  0

 ,DDCMP/CRF/DS:CRF←DMCNEW,LOW1,DMCGAL
RUN-TIME: 6 11 0 SECONDS
RUN-TIME RATIO: 366/17=21.1
CORE USED:  6K  (11 PAGES)


  1668
```

```
        MEM     IMM,<<I3-INIT&777/2>>
        MICPC=MICPC+1
        <MOVE!WRMEM!IMM!<<I3-INIT&777/2>>>
        ALWAYS  RM1
        MICPC=MICPC+1
        <JUMP!ALCOND!<RM1-INIT&3000*4>!<RM1-INIT&777/2>>
CBOOT:  BRWRTE  IMM,200                         ;MASK FOR BOOT MODE
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<200>>
        SP      BR,AORB,SP1                     ;IN PORT STATUS WORD
        MICPC=MICPC+1
        <MOVE!SPX!BR!AORB!SP1>
        BRWRTE  IMM,204                         ;MASK FOR OK TOSEND AND LINE IDLE
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<204>>
        SP      BR,SELB,SP10                    ;IN LINE STATUS
        MICPC=MICPC+1
        <MOVE!SPX!BR!SELB!SP10>
        TSTATE  TMTA
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<TMTA-INIT&777/2>>
        MICPC=MICPC+1
        <MOVE!SPX!BR!SELB!SP2>
        ALWAYS  INS12
        MICPC=MICPC+1
        <JUMP!ALCOND!<INS12-INIT&3000*4>!<INS12-INIT&777/2>>
BOOT:   BRWRTE  BR,SELA!SP1
        MICPC=MICPC+1
        <MOVE!WRTEBR!BR!<SELA!SP1>>
        BR7     RA3
        MICPC=MICPC+1
        <JUMP!BR7CON!<RA3-INIT&3000*4>!<RA3-INIT&777/2>>
        BRWRTE  IMM,210
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<210>>
        SP      BR,AORB,SP1
        MICPC=MICPC+1
        <MOVE!SPX!BR!AORB!SP1>
        ALWAYS  RA3
        MICPC=MICPC+1
        <JUMP!ALCOND!<RA3-INIT&3000*4>!<RA3-INIT&777/2>>
        .END
```

```
1669
1670  016324                HIMAP:                  ;HIGH SPEED (LOCAL) MICRO-CODE
```

DMCNEW.MAC 26-OCT-77 13:33 TABLE OF CONTENTS

DDCGAH.MAC 14-DEC-77 15:07 TABLE OF CONTENTS

```
                        .TITLE  DMC-11 MICROPROCESSOR INSTRUCTIONS
                        .SBTTL  MACRO DEFINITIONS
                        ;
                        .SBTTL REVISION 00
                        .SBTTL FEBRUARY 25, 1975
                        .SBTTL
                        .SBTTL REVISION 01
                        .SBTTL MARCH 18,1975
                        .SBTTL NEW CSR BOARD CHANGES
                        .SBTTL
                        .SBTTL HARVEY M. SCHLESINGER
                        ;
                        .SBTTL COPYRIGHT 1975 DIGITAL EQUIPMENT CORPORATION
                        ;
```

```
                        NEW=0
                        ;MICROPROCESSOR INSTRUCTION WORD DEFINITIONS
000000          MOVE=0                  ;OPCODE MOVE
100000          JUMP=100000             ;OPCODE JUMP
020000          IBUS=20000              ;SOURCE IBUS
000000          IMM=0                   ;SOURCVE IMMEDIATE
040000          MEMX=40000              ;SOURCE MEMORY
060000          BRX=60000               ;SOURCE BR
060000          BR=60000                ;SOURCE BR

060000          DP=60000                ;SOURCE BR
010000          LDMAR=10000             ;MA-LOAD MAR LO
014000          INCMAR=14000            ;MA-INCREMENT MAR
000400          WRTEBR=400              ;DEST-WRITE BR
001000          WROUTX=1000             ;DEST-EXTENDED IBUS
001400          SHFTBR=1400             ;DEST-SHIFT BR LEFT
002000          WROUT=2000              ;DEST-WRITE OUTPUT
002400          WRMEM=2400              ;DEST-WRITE MEMORY
003000          SPX=3000                ;DEST-WRITE SP
003400          SPBRX=3400              ;DEST-WRITE SP AND BR
                        ;FUNCTIONS
000200          SELA=200                ;FUNCTION-SELECT A
000220          SELB=220                ;FUNCTION-SELECT B
000240          AORNB=240               ;FUNCTION-A OR NOT B
000260          AANDB=260               ;FUNCTION A  AND B
000300          AORB=300                ;FUNCTION-A OR B
000320          AXORB=320               ;FUNCTION A XOR B
000340          SUB=340                 ;SUBTRACT
000360             SUBTC=360            ;FUNCTION- TWOS COMPLEMENT SUBTRACT
000000          ADD=0                   ;ADD A+B
000020          ADDC=20                 ;A+B+CARRY
000040          SUBC=40                 ;A-B-C
000060          INCA=60                 ;INCREMENT A
000100          AC=100                  ;A PLUS CARRY
000120          AA=120                  ;A PLUS A
000140          AAC=140                 ;A PLUS A PLUS C
000160          DECA=160                ;DECREMENT A
                        ;END FUNCTIONS
                        PAGE1=4000
                        PAGE2=10000
                        PAGE3=14000
001000          CCOND=1000              ;CONDITION C
001400          ZCOND=1400              ;CONDITION Z
000400          ALCOND=400              ;ALWAYS
002000          BR0CON=2000             ;CONDITION BR0
002400          BR1CON=2400             ;CONDITION BR1
003000          BR4CON=3000             ;CONDITION BR4
003400          BR7CON=3400             ;CONDITION BR7
```

```
                              .SBTTL  MICRO INSTRUCTION DEFINITIONS
                              .SBTTL          BRANCH INSTRUCTIONS
                        ;
        100000          JUMP=100000           ;JUMP OP CODE
                        ;
                        ;
                        ;
                        ;
                        ;
                        ;
                        ;
                        ;
                              .SBTTL          INDEXED BRANCH INSTRUCTIONS
                        ;
                        ;
                        ;
                        ;
                        ;
                        ;
                        ;
                              .SBTTL          MOVE INSTRUCTIONS
                        ;
        000000          MOVE=0          ;MOVE OPCODE
                        ;
                        ;
                        ;
                        ;
                        ;
                        ;
                        ;
                        ;
                        ;
                        ;
                        ;
                        ;
                        ;
                        ;
                        ;
                        ;
```

```
                                .SBTTL  INPUT/OUTPUT ASSIGNMENTS
                                ;IBUS   ASSIGNMENTS
100000                          INCON=0+100000          ;IN CONTROL CSR
100020                          MAIN=20+100000          ;MAINTAINENCE REGISTER
100040                          OCON=40+100000          ;OUT CONTROL CSR
100060                          UBADDR=60+100000        ;UNUSED
100100                          PORT1=100+100000        ;CSR4
100120                          PORT2=120+100000        ;CSR5
100140                          PORT3=140+100000        ;CSR6
100160                          PORT4=160+100000        ;CSR7
100200                          NPR=200+100000          ;NPR CONTROL
100220                          UBBR=220+100000         ;BR(INTERRUPT)CONTROL
000000                          INDAT1=0                ;INPUT DATA LOW BYTE
000020                          INDAT2=20               ;INPUT DATA HIGH BYTE
000140                          IOBA1=140               ;OUTPUT BA LOW BYTE
000160                          IOBA2=160               ;OUTPUT BA HIGH BYTE
000100                          IIBA1=100               ;INPUT BA LOW BYTE
000120                          IIBA2=120               ;INPUT BA HIGH BYTE
000200                          RCVDAT=200              ;RECEIVE DATA
000220                          TMTCON=220              ;TMTR CONTROL
000240                          RCVCON=240              ;RCVR CONTROL
000260                          MODEM=260               ;MODEM CONTROL
000300                          SYNREG=300              ;SYN REGISTER
000320                          LNOSW=320               ;LINE NUMBER SWITCH
000340                          BM873=340               ;BM873 ADDRESS
000360                          LUMAIN=360              ;LINE UNIT MAINTAINENCE
                                ;OBUS   ASSIGNMENTS
                                ;EXTENDED OBUS
000000                          OINCON=0                ;IN CONTROL CSR
000001                          OMAIN=1                 ;MAINT
000002                          OOCON=2                 ;OUT CONTROL CSR
000003                          OUBADD=3                ;UNUSED
000004                          OPORT1=4                ;CSR4
000005                          OPORT2=5                ;CSR5
000006                          OPORT3=6                ;CSR6
000007                          OPORT4=7                ;CSR7
000010                          ONPR=10                 ;NPR CONTROL
000011                          OBR=11                  ;BR CONTROL
                                ;UNEXTENDED OBUS
000000                          OINDAT1=0               ;OUTPUT OF.
000002                          OUTDA1=2                ;OUTPUT DATA LOW BYTE
000003                          OUTDA2=3                ;OUTPUT DATA HIGH BYTE
000006                          OBA1=6                  ;OUTPUT BA LOW BYTE
000007                          OBA2=7                  ;OUTPUT BA HIGH BYTE
000004                          IBA1=4                  ;INPUT BA LOW BYTE
000005                          IBA2=5                  ;INPUT BA HIGH BYTE
000010                          TMTDAT=10               ;TMTR DATA
000011                          OTMTCO=11               ;TMTR CONTROL
000012                          ORCVCO=12               ;RCVR CONTROL
000013                          OMODEM=13               ;MODEM CONTROL
000014                          SYNC=14                 ;SYN REGISTER
000017                          OLUMAN=17               ;LINE UNIT MAINT.
```

```
                        .SBTTL PROTOCOL DEPENDANT MACROS
                        ;
                        ;
                        ;
                        ;
                        ;
                        ;
177777                  MICPC=177777    ;INIT MICRO PC
```

```
354
355             000000          .SBTTL  DMC11 DDCMP MICRO CODE ASSEMBLED FOR USE  WITH THE M8201 LINE UNIT
                                LOW=0
```

```
                        .TITLE DMC11 DDCMP PROTOCOL IMPLEMENTATION

;
; COPYRIGHT (C) 1977
; DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.
;
; THIS SOFTWARE IS FURNISHED UNDER A LICENSE FOR USE ONLY ON A
; SINGLE COMPUTER SYSTEM AND MAY BE COPIED ONLY WITH THE
; INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE, OR
; ANY OTHER COPIES THEREOF, MAY NOT BE PROVIDED OR OTHERWISE
; MADE AVAILABLE TO ANY OTHER PERSON EXCEPT FOR USE ON SUCH
; SYSTEM AND TO ONE WHO AGREES TO THESE LICENSE TERMS. TITLE
; TO AND OWNERSHIP OF THE SOFTWARE SHALL AT ALL TIMES REMAIN
; IN DEC.
;
; THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT
; NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL
; EQUIPMENT CORPORATION.
;
; DEC ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF
; ITS SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DEC.
;
                        .SBTTL VERSION 00A FEBRUARY 26,1975
                        .SBTTL
                        .SBTTL HARVEY M. SCHLESINGER
                        .SBTTL
                        .SBTTL
                        .SBTTL  VERSION 00B     MARCH 17,1975
                        .SBTTL  CSR AND MICROPROCESSOR CHANGES
                        .SBTTL
                        .SBTTL  VERSION 00C     NOVEMBER 6, 1975
                        .SBTTL  RETRANSMISSION CHANGES
                        .SBTTL
                        .SBTTL  VERSION 00D     DECEMBER 3,1975
                        .SBTTL  TRANSMIT DONE CHANGES
                        .SBTTL
                        .SBTTL  VERSION 01A     6-MAY-77
                        .SBTTL  ACKNOWLEDGEMENT CHANGES - ACK A MESSAGE EVEN IF NO DATA TO RETURN
                        .SBTTL
                        .SBTTL  VERSION 01B     01-DEC-77
                        .SBTTL  MANY CHANGES
                        .SBTTL
                        .SBTTL  THE LATEST MODIFICATIONS WERE ADDED ON:
                        .SBTTL          01-DEC-77
```

```
                        .SBTTL  MICROPROCESSOR MAIN MEMORY ASSIGNMENTS
                        ;ALLOCATION OF MICROPROCESSOR MAIN MEMORY
000000                  NAKSR=0             ;NAKS RECD--DYNAMIC
000001                  NAKST=NAKSR+1       ;NAKS TMTED--DYNAMIC
000002                  REPSR=NAKST+1       ;REPS RECD--DYNAMIC
000003                  REPST=REPSR+1       ;REPS TMTED--DYNAMIC
000006                  NP=REPST+3          ;CONSTANT 0
000007                  NTLR=NP+1           ;NAKS-MSG NO BUFFERS CUMUL.
000010                  NHDR=NTLR+1         ;NAKS-MSG HEADER BAD
000011                  NDATR=NHDR+1        ;NAKS-DATA BAD
000012                  NTLS=NDATR+1        ;NAK SENT --NO BUFFERS
000013                  NHDS=NTLS+1         ;NAK SENT BAD HEADER
000014                  NDATS=NHDS+1        ;NAK SENT BAD DATA
000015                  REPCS=NDATS+1       ;REPS SENT CUMUL
000016                  REPCR=REPCS+1       ;REPS RECD CUMUL
000017                  BASE=REPCR+1                ;CORE TABLE BASE ADDRESS
000022                  SRC=BASE+3                  ;START OF INPUT CHAIN--NEXT RECV DONE
000023                  ERC=SRC+1                   ;END OF INPUT CHAIN
000024                  RCL1=ERC+1                  ;RECEIVE LINK #1
000031                  RCL2=RCL1+5                 ;   "       "   #2
000036                  RCL3=RCL2+5                 ;   "       "   #3
000043                  RCL4=RCL3+5
000050                  RCL5=RCL4+5
000055                  RCL6=RCL5+5
000062                  RCL7=RCL6+5
000067                  STC=RCL7+5                  ;START OF OUTPUT CHAIN---NEXT TMT DONE
000070                  ETC=STC+1                   ;END OF TRANSMIT CHAIN
000071                  TML1=ETC+1                  ;TRANSMIT LINK #1
000077                  TML2=TML1+6                 ;   "         "   #2
000105                  TML3=TML2+6                 ;   "         "   #3
000113                  TML4=TML3+6
000121                  TML5=TML4+6
000127                  TML6=TML5+6
000135                  TML7=TML6+6
000143                  TML8=TML7+6
000151                  T=TML8+6                    ;TYPE FIELD
000152                  ST=T+1              ;SUBTYPE FIELD
000153                  ISP17=ST+1          ;MSG ACKED IMAGE
000154                  IMG10=ISP17+1       ;IMAGE OF BIT 1 OF SP10
000155                  IMG11=IMG10+1       ;IMAGE OF SP11
000156                  IMG12=IMG11+1       ;IMAGE OF SP12
000157                  IMG14=IMG12+1       ;IMAGE OF SP14
000160                  IMG16=IMG14+1       ;IMAGE OF SP16
000161                  IMG17=IMG16+1       ;IMAGE OF SP17
000162                  TYPTAB=IMG17+1      ;TYPE TABLE---
                                            ;72 TYPE TABLE REP
                                            ;73  "      "   NAK
000164                  TYPSTT=TYPTAB+2     ;74  "      "   START
                                            ;75  "      "   STACK
                                            ;
                                            ;
000167                  BC=TYPSTT+3         ;RECEIVE BYTE COUNT
000171                  ISP11=BC+2          ;SP11 IMAGE
000172                  ISP12=ISP11+1       ;SP12 IMAGE
000173                  INCONS=ISP12+1      ;IN CONTROL CSR IMAGE
000174                  RTHRS=INCONS+1      ;RECV THRESHOLD LINK
```

```
                ;ALL LOCATIONS FROM 200 ON ARE NOT WRITTEN OUT DURING A TABLE UPDATE

000210                  TABST=210       ;TABLE UPDATE STATE
000211                  PRTST=TABST+1   ;PORT STATE
000240                  NXTINT=240      ;NEXT INTERRUPT POSITION
000241                  NXTSP=NXTINT+1  ;END OF INTERUPT CHAIN
000242                  INTSTK=NXTSF+1  ;STACK OF INTERUPTS
000400                  MMEND=400       ;MAIN MEMORY END
```

```
                                .SBTTL  SCRATCH PAD ASSIGNMENTS
000000                          SP0=0   ;SP0---SCRATCH REGISTER
000001                          SP1=1   ;SP1---PORT STATUS WORD
                                        ;BIT ASSIGNMENTS
                                        ;BIT0--INIT MODE
                                        ;BIT1--NO BUFFER ASSIGNED IN BOOT MODE
                                        ;BIT2--UNUSED
                                        ;BIT3--DLE RECEIVED WHILE NOT IN MAINT MODE
                                        ;BIT4--INTERRUPT PENDING
                                        ;BIT6--DISCONNECT ERROR
                                        ;BIT7--BOOT MODE
000002                          SP2=2   ;SP2---TRANSMIT STATE POINTER
000003                          SP3=3   ;SP3---RECEIVE STATE POINTER
000004                          SP4=4   ;SP4---END RECV ADDRESS
000005                          SP5=5   ;SP5---END RECEIVE ADDRESS
000006                          SP6=6   ;SP6---END TRANSMIT ADDRESS
000007                          SP7=7   ;SP7---END TRANSMIT ADDRESS
000010                          SP10=10 ;SP10---LINE STATUS WORD
                                        ;BIT ASSIGNMENTS
                                        ;BIT0--UNNUMB PENDING
                                        ;BIT1--MESSAGE IN PROGRESS
                                        ;BIT2--LINE HAS GONE IDLE
                                        ;BIT3--START RECEIVED
                                        ;BIT4--CLEAR ACTIVE ON END
                                        ;BIT5--START MODE
                                        ;BIT6--HALF DUPLEX
                                        ;BIT7--OK TO SEND
000011                          SP11=11 ;SP11---R FIELD
000012                          SP12=12 ;SP12---N FIELD
000013                          SP13=13 ;SP13---TYPE
000014                          SP14=14 ;SP14---TIMER ENTRY---NUMBER OF ONE SECOND TICKS
000015                          SP15=15 ;SP15---RECEIVE LINK IMAGE
000016                          SP16=16 ;SP16---POINTER TO TMT LINK COPY IN MAIN MEM
000017                          SP17=17 ;SP17---LAST MESSAGE ACKNOWLEDGED
```

```
                                        .SBTTL  INIT--INITIALIZATION ROUTINE
                                        ;ZEROS MAIN MEMORY
                                        ;LOOPS WAITING FOR RECEIVE DATA(BOOT?)
                                        ;OR FOR RQI TO BE SET
                                        ;WILL ACCEPT ONLY BASE FORMAT. ALL OTHERS WILL RETURN A PROCEDURE ERROR
                                        ;
                                        ;AT INITIALIZATION --- THE HARDWARE CLEARS THE BR AND MAR
                 016322                 .=16322
016322           016322 INIT:   SP       BR,SELB,SP0             ;CLEAR SP0
(1)                                     MICPC=MICPC+1
(1)              016322  063220         <MOVE!SPX!BR!SELB!SP0>
016324                                  SP       BR,SELB,SP3            ;PAGE ONE TRANSFER ADDRESS
(1)                                     MICPC=MICPC+1
(1)              016324  063223         <MOVE!SPX!BR!SELB!SP3>
016326                                  SP       BR,SELB,SP17           ;CLEAR SP17
(1)                                     MICPC=MICPC+1
(1)              016326  063237         <MOVE!SPX!BR!SELB!SP17>
016330                                  SP       BR,SELB,SP12           ;CLEAR SP12
(1)                                     MICPC=MICPC+1
(1)              016330  063232         <MOVE!SPX!BR!SELB!SP12>
016332                                  OUT      BR,<SELA!0INCON>       ;ZERO THE IN CONTROL CSR
(1)                                     MICPC=MICPC+1
(1)              016332  061200         <MOVE!WROUTX!BR!<SELA!0INCON>>
016334                                  OUT      BR,<SELA!0OCON>        ;ZERO THE OUT CONTROL CSR
(1)                                     MICPC=MICPC+1
(1)              016334  061202         <MOVE!WROUTX!BR!<SELA!0OCON>>
016336                                  SP       IMM,370,SP10           ;WRITE 5 ONE BITS TO THE HIGH ORDER
(1)                                     MICPC=MICPC+1
(1)              016336  003370         <MOVE!SPX!IMM!370!SP10>
                                                                       ;BITS OF SP10
016340           016340 5$:     SP       BR,AA,SP10             ;SHIFT SP10 LEFT SETTING CARRY THE
(1)                                     MICPC=MICPC+1
(1)              016340  063130         <MOVE!SPX!BR!AA!SP10>
                                                                       ;FIRST 5 TIMES THRU THE LOOP
016342                                  MEMINC   BR,ADDC!SP3            ;WRITE A ONE TO THE FIRST 5 MEMORY
(1)                                     MICPC=MICPC+1
(1)              016342  076423         <MOVE!WRMEM!INCMAR!BR!<ADDC!SP3>>
                                                                       ;LOCATIONS AND ZERO THE REST
016344                                  SP       BR,INCA,SP0            ;INCREMENT COUNTER
(1)                                     MICPC=MICPC+1
(1)              016344  063060         <MOVE!SPX!BR!INCA!SP0>
016346                                  Z        10$                    ;ALL DONE
(1)                                     MICPC=MICPC+1
(1)              016346  101414         <JUMP!ZCOND!<10$-INIT&3000*4>!<10$-INIT&777/2>>
016350                                  ALWAYS   5$                     ;KEEP GOING
(1)                                     MICPC=MICPC+1
(1)              016350  100407         <JUMP!ALCOND!<5$-INIT&3000*4>!<5$-INIT&777/2>>
016352           016352 10$:    SPBR     IMM,1,SP1              ;WRITE A 1 TO THE BR AND SP1
(1)                                     MICPC=MICPC+1
(1)              016352  003401         <MOVE!SPBRX!IMM!1!SP1>
016354                                  SP       BR,SELB,SP11           ;WRITE A 1 TO SP11
(1)                                     MICPC=MICPC+1
(1)              016354  063231         <MOVE!SPX!BR!SELB!SP11>
016356                                  LDMA     IMM,TYPTAB             ;POINT MAR TO TYPE TABLE
(1)                                     MICPC=MICPC+1
(1)              016356  010162         <MOVE!LDMAR!IMM!<TYPTAB&377>>
```

```
531  016360
(1)            000017
(1)  016360    000626
532  016362
(1)            000020
(1)  016362    062234
533  016364
(1)            000021
(1)  016364    016403
534  016366
(1)            000022
(1)  016366    016402
535  016370
(1)            000023
(1)  016370    016406
536  016372
(1)            000024
(1)  016372    016407
537  016374
(1)            000025
(1)  016374    016401
538  016376
(1)            000026
(1)  016376    010210
539  016400
(1)  016400
(2)            000027
(2)  016400    016455
540  016402
(1)  016402
(2)            000030
(2)  016402    016472
541  016404
(1)            000031
(1)  016404    010067
542  016406
(1)            000032
(1)  016406    016471
543  016410
(1)            000033
(1)  016410    002471
544  016412
(1)            000034
(1)  016412    043236
545  016414
(1)            000035
(1)  016414    010022
546  016416
(1)            000036
(1)  016416    016424
547  016420
(1)            000037
(1)  016420    002424
548  016422
(1)            000040
(1)  016422    043235
```

```
        BRWRTE  IMM,226                 ;WRITE SYNC TO MEMORY
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<226>>
        OUTPUT  BR,SELB!SYNC    ;LOAD THE SYNC REGISTER
        MICPC=MICPC+1
        <MOVE!WROUT!BR!<SELB!SYNC>>
        MEMINC  IMM,3                   ;REP
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<3>>
        MEMINC  IMM,2                   ;NAK
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<2>>
        MEMINC  IMM,6                   ;START
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<6>>
        MEMINC  IMM,7                   ;STACK
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<7>>
        MEMINC  IMM,1                   ;ACK
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<1>>
        LDMA    IMM,TABST               ;POINT TO TABLE UPDATE STATE
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<TABST&377>>
        PSTATI  I3                      ;INITIALIZE IT
        MEMINC  IMM,<<I3-INIT&777/2>>
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<<I3-INIT&777/2>>>
        PSTATI  NIDLE2                  ;INITIALIZE PORT STATUS
        MEMINC  IMM,<<NIDLE2-INIT&777/2>>
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<<NIDLE2-INIT&777/2>>>
        LDMA    IMM,STC                 ;LOAD ADDRESS OF LAST TMT CHAIN
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<STC&377>>
        MEMINC  IMM,TML1                ;STORE ADDRESS OF FIRST TMT LINK
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<TML1>>
        MEM     IMM,TML1
        MICPC=MICPC+1
        <MOVE!WRMEM!IMM!<TML1>>
        SP      MEMX,SELB,SP16          ;INITIALIZE LAST XMIT POINTER
        MICPC=MICPC+1
        <MOVE!SPX!MEMX!SELB!SP16>
        LDMA    IMM,SRC                 ;LOAD ADDRESS OF LAST RECV CHAIN
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<SRC&377>>
        MEMINC  IMM,RCL1                ;SET UP ADDRESS OF FIRST RECV LINK
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<RCL1>>
        MEM     IMM,RCL1
        MICPC=MICPC+1
        <MOVE!WRMEM!IMM!<RCL1>>
        SP      MEMX,SELB,SP15
        MICPC=MICPC+1
        <MOVE!SPX!MEMX!SELB!SP15>
```

```
549  016424
(1)            000041
(1)  016424    010240
550  016426
(1)            000042
(1)  016426    016642
551  016430
(1)            000043
(1)  016430    002642
552  016432
(1)            000044
(1)  016432    000600
553  016434
(1)            000045
(1)  016434    061221
554
555  016436
(1)            000046
(1)  016436    110642
```

```
        LDMA    IMM,NXTINT                      ;ADDRESS OF NEXT INTERRUPT POINTER TO MAR
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<NXTINT&377>>
        MEMINC  IMM,INTSTK                              ;INITIALIZE NEXT INTERRUPT POINTER
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<INTSTK>>
        MEM     IMM,INTSTK                              ;INITIALIZE INSERTION POINTER
        MICPC=MICPC+1
        <MOVE!WRMEM!IMM!<INTSTK>>
        BRWRTE  IMM,200                         ;MASK TO SET RUN BIT
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<200>>
        OUT     BR,<SELB!OMAIN>                 ;WRITE  THE RUN BIT TO MAINT CSR
        MICPC=MICPC+1
        <MOVE!WROUTX!BR!<SELB!OMAIN>>
                                                ;FALL INTO IDLE LOOP
        ALWAYS  TEOM2
        MICPC=MICPC+1
        <JUMP!ALCOND!<TEOM2-INIT&3000*4>!<TEOM2-INIT&777/2>>
```

```
                                        .SBTTL  IDLE--PROGRAM IDLE LOOP
                                        ;PROGRAM IDLE LOOP
                                        ;DISPATCHES TO APPROPRIATE SERVICE ROUTINES
                                        ;USES STATE POINTERS FOR TMT,RCV,CSR ACTIVITY
                                        ;
                                IDLE:
                                        BRWRTE  IBUS,TMTCON
                                        MICPC=MICPC+1
                                        <MOVE!WRTEBR!IBUS!<TMTCON>>
                                        .BR4    DP,SELA,<2!PAGE2>
                                        MICPC=MICPC+1
                                        <JUMP!  BR4CON!DP!SELA!2!PAGE2>
                                I1:     BRWRTE  IBUS,RCVCON             ;READ LINE UNIT RECEIVE CONTROL WORD
                                        MICPC=MICPC+1
                                        <MOVE!WRTEBR!IBUS!<RCVCON>>
                                        .BR4    BR,SELA,SP3!PAGE1       ;BRANCH BASED UPON RECV STATE
                                        MICPC=MICPC+1
                                        <JUMP!  BR4CON!BR!SELA!SP3!PAGE1>
                                I2:     LDMA    IMM,TABST               ;POINT TO TABLE UPDATE STATE
                                        MICPC=MICPC+1
                                        <MOVE!LDMAR!IMM!<TABST&377>>
                                        .ALWAY  MEMX,SELB,0
                                        MICPC=MICPC+1
                                        <JUMP!ALCOND!MEMX!SELB!0>
                                I3:
                                        BRWRTE  IBUS,TMTCON
                                        MICPC=MICPC+1
                                        <MOVE!WRTEBR!IBUS!<TMTCON>>
                                        BR4     I4
                                        MICPC=MICPC+1
                                        <JUMP!BR4CON!<I4-INIT&3000*4>!<I4-INIT&777/2>>
                                        ALWAYS  IDLE0
                                        MICPC=MICPC+1
                                        <JUMP!ALCOND!<IDLE0-INIT&3000*4>!<IDLE0-INIT&777/2>>
                                I4:     STATE   TMTA5
                                        MICPC=MICPC+1
                                        <MOVE!WRTEBR!IMM!<TMTA5-INIT&777/2>>
                                        NOP     BR,SUB,SP2
                                        MICPC=MICPC+1
                                        <BR!SUB!SP2>
                                        C       IDLE
                                        MICPC=MICPC+1
                                        <JUMP!CCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
                                IDLE0:  SPBR    IBUS,UBBR,SP0           ;TIMER EXPIRES?
                                        MICPC=MICPC+1
                                        <MOVE!SPBRX!IBUS!UBBR!SP0>
                                        BR4     TIMSRV
                                        MICPC=MICPC+1
                                        <JUMP!BR4CON!<TIMSRV-INIT&3000*4>!<TIMSRV-INIT&777/2>>
                                        SP      IBUS,RCVCON,SP0         ;READ RCV CONTROL REGISTER
                                        MICPC=MICPC+1
                                        <MOVE!SPX!IBUS!RCVCON!SP0>
                                        BRWRTE  BR,AA!SP0               ;SHIFT LEFT
                                        MICPC=MICPC+1
                                        <MOVE!WRTEBR!BR!<AA!SP0>>
                                        BR7     I1                      ;RCV ACTIVE, DON'T DO PORT STATUS
```

```
(1)            000067
(1)  016500    103451
585  016502
(1)            000070
(1)  016502    010211
586  016504
(1)            000071
(1)  016504    140620

        MICPC=MICPC+1
        <JUMP!BR7CON!<I1-INIT&3000*4>!<I1-INIT&777/2>>
        LDMA    IMM,PRTST                       ;ADDRESS PORT STATE
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<PRTST&377>>
        .ALWAY  MEMX,SELB,0                     ;INDEX
        MICPC=MICPC+1
        <JUMP!ALCOND!MEMX!SELB!0>
```

M12


```
                        .SBTTL  NIDLE2---NO CSR ACTIVITY STATE
NIDLE2: BRWRTE  BR,SELA!SP1             ;READ PORT STATUS WORD
        MICPC=MICPC+1
        <MOVE!WRTEBR!BR!<SELA!SP1>>
        BR4     NIDLE5                  ;INTERRUPT PENDING?---BRANCH
        MICPC=MICPC+1
        <JUMP!BR4CON!<NIDLE5-INIT&3000*4>!<NIDLE5-INIT&777/2>>
        SPBR    IBUS,INCON,SPO          ;READ INPUT CONTROL CSR
        MICPC=MICPC+1
        <MOVE!SPBRX!IBUS!INCON!SPO>
        BRSHFT                          ;SHIFT IT RIGHT
        MICPC=MICPC+1
        <MOVE!SHFTBR!WRTEBR!SELB>
        BR4     INWAT1                  ;IF RQI SET -- BRANCH
        MICPC=MICPC+1
        <JUMP!BR4CON!<INWAT1-INIT&3000*4>!<INWAT1-INIT&777/2>>
                                        ;TO RE-READ THE IN CNTRL REGISTER TO AVOID
                                        ;A RACE IN MICRO-P READ/UNIBUS WRITE
        ALWAYS  IDLE
        MICPC=MICPC+1
        <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
NIDLE6:
NIDLE5: PSTATE  OUTINT                  ;SET STATE FOR INTERRUPT PROCESSING
        MEM     IMM,<<OUTINT-INIT&777/2>>
        MICPC=MICPC+1
        <MOVE!WRMEM!IMM!<<OUTINT-INIT&777/2>>>
        ALWAYS  IDLE
        MICPC=MICPC+1
        <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
```

```
589
590  016506
(1)          000072
(1)  016506  060601
592  016510
(1)          000073
(1)  016510  103100
597  016512
(1)          000074
(1)  016512  123400
598  016514
(1)          000075
(1)  016514  001620
599  016516
(1)          000076
(1)  016516  117034
600
601
603  016520
(1)          000077
(1)  016520  100447
605  016522
626  016522
(1)  016522
(2)          000100
(2)  016522  002611
627  016524
(1)          000101
(1)  016524  100447
```

```
630  016526
(1)            000102
(1)  016526    123400
631  016530
(1)            000103
(1)  016530    103505
632  016532
(1)            000104
(1)  016532    114434
633  016534
(1)            000105
(1)  016534    060520
634  016536
(1)            000106
(1)  016536    103603
635  016540
(1)            000107
(1)  016540    120400
636  016542
(1)            000110
(1)  016542    001620
641  016544
(1)            000111
(1)  016544    103047
642  016546
(1)  016546
(2)            000112
(2)  016546    002514
643  016550
(1)            000113
(1)  016550    100447
645  016552
(1)            000114
(1)  016552    123400
646  016554
(1)            000115
(1)  016554    102527
647  016556
(1)  016556
(2)            000116
(2)  016556    002654
648  016560
(1)            000117
(1)  016560    102124
649  016562
(1)            000120
(1)  016562    001620
650  016564
(1)  016564
(2)            000121
(2)  016564    002722
651  016566
(1)            000122
(1)  016566    102524
652  016570
(1)  016570
```

```
IEIWAT: SPBR    IBUS,INCON,SPO          ;READ INPUT CONTROL CSR
        MICPC=MICPC+1
        <MOVE!SPBRX!IBUS!INCON!SPO>
        BR7     10$                     ;RDYI STILL SET
        MICPC=MICPC+1
        <JUMP!BR7CON!<10$-INIT&3000*4>!<10$-INIT&777/2>>
        ALWAYS  INWAT1                  ;GOT CLEARED BY 11 RE-SET IT
        MICPC=MICPC+1
        <JUMP!ALCOND!<INWAT1-INIT&3000*4>!<INWAT1-INIT&777/2>>
10$:    BRWRTE  BR,<AA!SPO>             ;CHECK IEI
        MICPC=MICPC+1
        <MOVE!WRTEBR!BR!<AA!SPO>>
        BR7     ININT                   ;IEI SET, GENERATE INTERRUPT
        MICPC=MICPC+1
        <JUMP!BR7CON!<ININT-INIT&3000*4>!<ININT-INIT&777/2>>
INWAT2: BRWRTE  IBUS,INCON              ;READ INPUT CONTROL CSR
        MICPC=MICPC+1
        <MOVE!WRTEBR!IBUS!<INCON>>
        BRSHFT
        MICPC=MICPC+1
        <MOVE!SHFTBR!WRTEBR!SELB>
        BR4     IDLE                    ;RQI STILL SET... GO AWAY
        MICPC=MICPC+1
        <JUMP!BR4CON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
        PSTATE  INSRV                   ;SET NEXT STATE TO INPUT SERVICE
        MEM     IMM,<<INSRV-INIT&777/2>>
        MICPC=MICPC+1
        <MOVE!WRMEM!IMM!<<INSRV-INIT&777/2>>>
        ALWAYS  IDLE
        MICPC=MICPC+1
        <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
INSRV:  SPBR    IBUS,INCON,SPO          ;READ THE INPUT CONTROL CSR
        MICPC=MICPC+1
        <MOVE!SPBRX!IBUS!INCON!SPO>
        BR1     30$                     ;--SENSE OR BASE
        MICPC=MICPC+1
        <JUMP!BR1CON!<30$-INIT&3000*4>!<30$-INIT&777/2>>
        PSTATE  CTLSRV                  ;ASSUME CNTL I
        MEM     IMM,<<CTLSRV-INIT&777/2>>
        MICPC=MICPC+1
        <MOVE!WRMEM!IMM!<<CTLSRV-INIT&777/2>>>
        BR0     20$                     ;CNTL I
        MICPC=MICPC+1
        <JUMP!BR0CON!<20$-INIT&3000*4>!<20$-INIT&777/2>>
        BRSHFT                          ;MUST BE BA/CC
        MICPC=MICPC+1
        <MOVE!SHFTBR!WRTEBR!SELB>
        PSTATE  RBASRV                  ;ASSUME IN
        MEM     IMM,<<RBASRV-INIT&777/2>>
        MICPC=MICPC+1
        <MOVE!WRMEM!IMM!<<RBASRV-INIT&777/2>>>
        BR1     20$                     ;REC BA/CC
        MICPC=MICPC+1
        <JUMP!BR1CON!<20$-INIT&3000*4>!<20$-INIT&777/2>>
        PSTATE  TBASRV                  ;XMIT BA/CC
        MEM     IMM,<<TBASRV-INIT&777/2>>
```

```
(2)             000123
(2)  016570  002701
653  016572
(1)             000124
(1)  016572  060601
654  016574
(1)             000125
(1)  016574  102130
655  016576
(1)             000126
(1)  016576  100447
656  016600
(1)             000127
(1)  016600  102140
657  016602
(1)  016602
(2)             000130
(2)  016602  002472
658  016604
(1)             000131
(1)  016604  000500
659  016606
(1)             000132
(1)  016606  061260
660  016610
(1)             000133
(1)  016610  010177
661  016612
(1)             000134
(1)  016612  016402
662  016614
(1)             000135
(1)  016614  002400
663  016616
(1)             000136
(1)  016616  042233
664  016620
(1)             000137
(1)  016620  114474
665  016622
(1)             000140
(1)  016622  060721
666  016624
(1)             000141
(1)  016624  102130
```

```
        MICPC=MICPC+1
        <MOVE!WRMEM!IMM!<<TBASRV-INIT&777/2>>>
20$:    BRWRTE  BR,SELA!SP1             ;INIT MODE
        MICPC=MICPC+1
        <MOVE!WRTEBR!BR!<SELA!SP1>>
        BRO     PROCER                  ;IF INIT MODE--ERROR
        MICPC=MICPC+1
        <JUMP!BROCON!<PROCER-INIT&3000*4>!<PROCER-INIT&777/2>>
        ALWAYS  IDLE
        MICPC=MICPC+1
        <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
30$:    BRO     INSRV1                          ;IF BASE---PROCESS
        MICPC=MICPC+1
        <JUMP!BROCON!<INSRV1-INIT&3000*4>!<INSRV1-INIT&777/2>>
PROCER: PSTATE  NIDLE2                  ;RESET PORT STATUS
        MEM     IMM,<<NIDLE2-INIT&777/2>>
        MICPC=MICPC+1
        <MOVE!WRMEM!IMM!<<NIDLE2-INIT&777/2>>>
        BRWRTE  IMM,100                 ;CLEAR INPUT CONTROL CSR
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<100>>
        OUT     BR,AANDB!OINCON         ;;
        MICPC=MICPC+1
        <MOVE!WROUTX!BR!<AANDB!OINCON>>
        LDMA    IMM,<<RTHRS+3>>         ;ADDRESS ERROR LINK
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<<RTHRS+3>&377>>
        MEMINC  IMM,2
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<2>>
        MEM     IMM,0
        MICPC=MICPC+1
        <MOVE!WRMEM!IMM!<0>>
        OUTPUT  MEMX,SELB!OMODEM        ;CLEAR DATA TERMINAL READY
        MICPC=MICPC+1
        <MOVE!WROUT!MEMX!<SELB!OMODEM>>
        ALWAYS  RCEXX                   ;POST THE ERROR - FATAL
        MICPC=MICPC+1
        <JUMP!ALCOND!<RCEXX-INIT&3000*4>!<RCEXX-INIT&777/2>>
INSRV1: BRWRTE  BR,AXORB!SP1                    ;INIT MODE?
        MICPC=MICPC+1
        <MOVE!WRTEBR!BR!<AXORB!SP1>>
        BRO     PROCER
        MICPC=MICPC+1
        <JUMP!BROCON!<PROCER-INIT&3000*4>!<PROCER-INIT&777/2>>
```

```
668
669  016626
(1)  016626
(2)            000142
(2)  016626    002472
670  016630
(1)            000143
(1)  016630    010017
671  016632
(1)            000144
(1)  016632    136500
672  016634
(1)            000145
(1)  016634    136520
673  016636
(1)            000146
(1)  016636    122560
674  016640
(1)            000147
(1)  016640    123000
675  016642
(1)            000150
(1)  016642    000500
676  016644
(1)            000151
(1)  016644    061260
677  016646
(1)            000152
(1)  016646    002133
678  016650
(1)            000153
(1)  016650    040620
679  016652
(1)            000154
(1)  016652    103164
680  016654
(1)            000155
(1)  016654    010151
681  016656
(1)            000156
(1)  016656    016406
682  016660
(1)            000157
(1)  016660    002700
683  016662
(1)            000160
(1)  016662    063161
684  016664
(1)            000161
(1)  016664    000641
685  016666
(1)            000162
(1)  016666    003374
686  016670
(1)            000163
(1)  016670    110727
```

```
        .SBTTL  BASSRV---- BASE SERVICE ROUTINE
BASSRV: PSTATE  NIDLE2
        MEM     IMM,<<NIDLE2-INIT&777/2>>
        MICPC=MICPC+1
        <MOVE!WRMEM!IMM!<<NIDLE2-INIT&777/2>>>
        LDMA    IMM,BASE                        ;CLEAR TO MAR SO IT POINTS TO BASE POINT
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<BASE&377>>
        MEMINC  IBUS,PORT1                      ;READ CSR4
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IBUS!<PORT1>>
        MEMINC  IBUS,PORT2                      ;READ CSR5
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IBUS!<PORT2>>
        MEM     IBUS,PORT4
        MICPC=MICPC+1
        <MOVE!WRMEM!IBUS!<PORT4>>
        SP      IBUS,INCON,SP0                  ;READ INPUT CONTROL CSR
        MICPC=MICPC+1
        <MOVE!SPX!IBUS!INCON!SP0>
        BRWRTE  IMM,100                         ;CLEAR THE BR
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<100>>
        OUT     BR,<AANDB!0INCON>               ;CLEAR THE INCONTROL CSR
        MICPC=MICPC+1
        <MOVE!WROUTX!BR!<AANDB!0INCON>>
        OUTPUT  IMM,<120!0MODEM>                ;MASK FOR HDX AND DTR
        MICPC=MICPC+1
        <MOVE!WROUT!IMM!<120!0MODEM>>
        BRWRTE  MEMX,SELB                       ;READ SEL6
        MICPC=MICPC+1
        <MOVE!WRTEBR!MEMX!<SELB>>
        BR4     RESUME                          ;IF SET RESUME
        MICPC=MICPC+1
        <JUMP!BR4CON!<RESUME-INIT&3000*4>!<RESUME-INIT&777/2>>
        LDMA    IMM,T                           ;LOAD ADDRESS OF TYPE FIELD
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<T&377>>
        MEMINC  IMM,6                           ;WRITE START TYPE TO MEMORY
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<6>>
        MEM     IMM,300                         ;WRITE SELECT AND FINAL TO MEMORY
        MICPC=MICPC+1
        <MOVE!WRMEM!IMM!<300>>
        SP      BR,DECA,SP1     ;TURN OFF INIT MODE
        MICPC=MICPC+1
        <MOVE!SPX!BR!DECA!SP1>
BS1:    BRWRTE  IMM,241                         ;SET OK TO SEND,STARTMODE AND UNNUM PENDING
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<241>>
BS0:    SP      IMM,374,SP14    ;RESET REP COUNTER
        MICPC=MICPC+1
        <MOVE!SPX!IMM!374!SP14>
        ALWAYS  SA3
        MICPC=MICPC+1
        <JUMP!ALCOND!<SA3-INIT&3000*4>!<SA3-INIT&777/2>>
```

BASSRV---- BASE SERVICE ROUTINE

```
687  016672
(1)            000164
(1)  016672    003004
688  016674
(1)            000165
(1)  016674    063070
689
690  016676
(1)            000166
(1)  016676    010017
691  016700
(1)            000167
(1)  016700    000731
692  016702
(1)            000170
(1)  016702    110463
693  016704
(1)            000171
(1)  016704    010154
694  016706
(1)            000172
(1)  016706    057310
695  016710
(1)            000173
(1)  016710    057231
696  016712
(1)            000174
(1)  016712    057235
697  016714
(1)            000175
(1)  016714    043237
698  016716
(1)            000176
(1)  016716    043232
699  016720
(1)            000177
(1)  016720    063170
700
701  016722
(1)            000200
(1)  016722    063161
702  016724
(1)            000201
(1)  016724    000606
703  016726
(1)            000202
(1)  016726    114424
704  016730
(1)  016730
(2)            000203
(2)  016730    002507
705  016732
(1)            000204
(1)  016732    000415
706  016734
(1)            000205
```

```
RESUME: SP      IMM,SP4,4               ;SET UP SP4 FOR COUNTING NPRS
        MICPC=MICPC+1
        <MOVE!SPX!IMM!SP4!4>
        SP      BR,INCA,SP10            ;SET UNNUMB MESSAGE PENDING TO
        MICPC=MICPC+1
        <MOVE!SPX!BR!INCA!SP10>
                                        ;TRICK TRANSMITTER CODE
        LDMA    IMM,BASE                ;ADDRESS BASE TABLE ADDRESS
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<BASE&377>>
        STATE   FUDGE                   ;SET TMTR STATE TO ENTER TABLE UPDATE
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<FUDGE-INIT&777/2>>
        ALWAYS  TBO                     ;GO SET UP MXT BITS AND ADRESS OF BASE FOR NPRS
        MICPC=MICPC+1
        <JUMP!ALCOND!<TBO-INIT&3000*4>!<TBO-INIT&777/2>>
BS2:    LDMA    IMM,IMG10
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<IMG10&377>>
        SP      MEMX!INCMAR,AORB,SP10           ;RESTORE BIT 1 OF SP10
        MICPC=MICPC+1
        <MOVE!SPX!MEMX!INCMAR!AORB!SP10>
        SP      MEMX!INCMAR,SELB,SP11           ;RESTORE SP11
        MICPC=MICPC+1
        <MOVE!SPX!MEMX!INCMAR!SELB!SP11>
        SP      MEMX!INCMAR,SELB,SP15           ;RESTORE SP15
        MICPC=MICPC+1
        <MOVE!SPX!MEMX!INCMAR!SELB!SP15>
        SP      MEMX,SELB,SP17          ;RESTORE        SP17
        MICPC=MICPC+1
        <MOVE!SPX!MEMX!SELB!SP17>
        SP      MEMX,SELB,SP12          ;RESTORE        SP12
        MICPC=MICPC+1
        <MOVE!SPX!MEMX!SELB!SP12>
        SP      BR,DECA,SP10            ;TURN OFF UNNUM MESSAGE PENDING AND
        MICPC=MICPC+1
        <MOVE!SPX!BR!DECA!SP10>
                                        ;ZERO THE BRG
        SP      BR,DECA,SP1             ;CLEAR INIT MODE
        MICPC=MICPC+1
        <MOVE!SPX!BR!DECA!SP1>
        BRWRTE  IMM,206
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<206>>
        ALWAYS  NAK1
        MICPC=MICPC+1
        <JUMP!ALCOND!<NAK1-INIT&3000*4>!<NAK1-INIT&777/2>>
ININT:  PSTATE  INWAT2
        MEM     IMM,<<INWAT2-INIT&777/2>>
        MICPC=MICPC+1
        <MOVE!WRMEM!IMM!<<INWAT2-INIT&777/2>>>
        BRWRTE  IMM,15                  ;MASK FOR TURN OFF ALL BUT EXT MEM BITS + NXM
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<15>>
        SP      IBUS,UBBR,SP0           ;READ BR CONTROL REG
        MICPC=MICPC+1
```

```
(1)  016734  123220
707  016736
(1)          000206
(1)  016736  063260
708  016740
(1)          000207
(1)  016740  000600
709  016742
(1)          000210
(1)  016742  104457
```

```
        <MOVE!SPX!IBUS!UBBR!SPO>
SP      BR,AANDB,SPO            ;MASK OFF VECTOR TO X04
MICPC=MICPC+1
        <MOVE!SPX!BR!AANDB!SPO>
BRWRTE  IMM,200                 ;MASK FOR INTERRUPT
MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<200>>
ALWAYS RB4
MICPC=MICPC+1
        <JUMP!ALCOND!<RB4-INIT&3000*4>!<RB4-INIT&777/2>>
```

```
OUTINT---SET UP OUTPUT INTERRUPT [RDYO]

                .SBTTL  OUTINT---SET UP OUTPUT INTERRUPT [RDYO]
OUTINT:
                PSTATE  PINT2
                MEM     IMM,<<PINT2-INIT&777/2>>
                MICPC=MICPC+1
                <MOVE!WRMEM!IMM!<<PINT2-INIT&777/2>>>
                                        ;COMPLETION
                LDMA    IMM,NXTINT              ;ADDRESS OF NEXT INTERRUPT POINTER
                MICPC=MICPC+1
                <MOVE!LDMAR!IMM!<NXTINT&377>>
                LDMA    MEMX,SELB               ;NEXT INTERRUPT
                MICPC=MICPC+1
                <MOVE!LDMAR!MEMX!<SELB>>
                SP      IBUS,OCON,SPO           ;READ THE OUTPUT CONTROL CSR
                MICPC=MICPC+1
                <MOVE!SPX!IBUS!OCON!SPO>
                OUT     <MEMX!INCMAR>,<AORB!OOCON>      ;WRITE THE OUT CONTROL CSR
                MICPC=MICPC+1
                <MOVE!WROUTX!MEMX!INCMAR!<AORB!OOCON>>
                LDMA    MEMX,SELB                       ;ADDRESS LINK
                MICPC=MICPC+1
                <MOVE!LDMAR!MEMX!<SELB>>
                BRWRTE  <BR!INCMAR>,<AA!SPO>            ;KICK PAST LINK STATUS BYTE
                MICPC=MICPC+1
                <MOVE!WRTEBR!BR!INCMAR!<AA!SPO>>
                                                        ;SHIFT CSRO IMAGE LEFT
                                                ;***DO NOT CHANGE BR UNTIL BR7***
                OUT     <MEMX!INCMAR>,<SELB!OPORT1>     ;WRITE LOW BYTE OF BA TO CSR
                MICPC=MICPC+1
                <MOVE!WROUTX!MEMX!INCMAR!<SELB!OPORT1>>
                OUT     <MEMX!INCMAR>,<SELB!OPORT2>     ;WRITE HIGH BYTE OF BA TO CSR
                MICPC=MICPC+1
                <MOVE!WROUTX!MEMX!INCMAR!<SELB!OPORT2>>
                OUT     <MEMX!INCMAR>,<SELB!OPORT4>     ;WRITE HIGH BYTE OF COUNT TO CSR
                MICPC=MICPC+1
                <MOVE!WROUTX!MEMX!INCMAR!<SELB!OPORT4>>
                OUT     <MEMX!INCMAR>,<SELB!OPORT3>     ;WRITE THE LOW BYTE OF COUNT
                MICPC=MICPC+1
                <MOVE!WROUTX!MEMX!INCMAR!<SELB!OPORT3>>
                                                ;***HERE IS BR7***
                BR7     PE1                     ;INTERRUPT ENABLE IS SET
                MICPC=MICPC+1
                <JUMP!BR7CON!<PE1-INIT&3000*4>!<PE1-INIT&777/2>>
                                                ;GENERATE AN INTERRUPT
                ALWAYS  IDLE
                MICPC=MICPC+1
                <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
PINT2:  PSTATE  OUTWAIT
                MEM     IMM,<<OUTWAIT-INIT&777/2>>
                MICPC=MICPC+1
                <MOVE!WRMEM!IMM!<<OUTWAIT-INIT&777/2>>>
                LDMA    IMM,NXTINT              ;ADDRESS NEXT INTERRUPT QUEUE
                MICPC=MICPC+1
                <MOVE!LDMAR!IMM!<NXTINT&377>>
                SP      MEMX,SELB,SPO           ;COPY ADDRESS FOR NEXT INT TO SPO
                MICPC=MICPC+1
```

```
(1)  017002  043220
744  017004
(1)            000231
(1)  017004  002642
745  017006
(1)            000232
(1)  017006  000776
746  017010
(1)            000233
(1)  017010  060360
747  017012
(1)            000234
(1)  017012  101637
748  017014
(1)            000235
(1)  017014  000402
749  017016
(1)            000236
(1)  017016  062400
750  017020
(1)            000237
(1)  017020  043220
751  017022
(1)            000240
(1)  017022  010241
752  017024
(1)            000241
(1)  017024  040360
753  017026
(1)            000242
(1)  017026  101644
754  017030
(1)            000243
(1)  017030  100447
755  017032
(1)            000244
(1)  017032  000757
756  017034
(1)            000245
(1)  017034  063261
757  017036
(1)            000246
(1)  017036  100447
```

OUTINT---SET UP OUTPUT INTERRUPT [RDY0]

```
                <MOVE!SPX!MEMX!SELB!SPO>
                MEM     IMM,INTSTK              ;ASSUME WRAP AROUND CASE
                MICPC=MICPC+1
                <MOVE!WRMEM!IMM!<INTSTK>>
                BRWRTE  IMM,<<MMEND-2>>         ;ADDRESS OF LAST INT IN STACK
                MICPC=MICPC+1
                <MOVE!WRTEBR!IMM!<<MMEND-2>>>
                CMP     BR,SPO                  ;SHOULD WE WRAP
                MICPC=MICPC+1
                <SUBTC!BR!SPO>
                Z       5$                      ;YES--BRANCH
                MICPC=MICPC+1
                <JUMP!ZCOND!<5$-INIT&3000*4>!<5$-INIT&777/2>>
                BRWRTE  IMM,2                   ;OFFSET FOR NEXT POINTER
                MICPC=MICPC+1
                <MOVE!WRTEBR!IMM!<2>>
                MEM     BR,ADD!SPO              ;UPDATE POINTER
                MICPC=MICPC+1
                <MOVE!WRMEM!BR!<ADD!SPO>>
5$:             SP      MEMX,SELB,SPO           ;COPY POINTER TO SPO
                MICPC=MICPC+1
                <MOVE!SPX!MEMX!SELB!SPO>
                LDMA    IMM,NXTSP               ;PICK UP START OF IN QUEUE
                MICPC=MICPC+1
                <MOVE!LDMAR!IMM!<NXTSP&377>>
                CMP     MEMX,SPO                ;COMPARE TO END
                MICPC=MICPC+1
                <SUBTC!MEMX!SPO>
                Z       10$                     ;IF EQUAL--CLEAR INT PENDING
                MICPC=MICPC+1
                <JUMP!ZCOND!<10$-INIT&3000*4>!<10$-INIT&777/2>>
                ALWAYS  IDLE
                MICPC=MICPC+1
                <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
10$:            BRWRTE  IMM,357                 ;MASK TO CLEAR INT PENDING
                MICPC=MICPC+1
                <MOVE!WRTEBR!IMM!<357>>
CLRIDL:         SP      BR,AANDB,SP1
                MICPC=MICPC+1
                <MOVE!SPX!BR!AANDB!SP1>
                ALWAYS  IDLE
                MICPC=MICPC+1
                <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
```

```
                                OUTWAI--WAIT FOR RDYO TO GO AWAY

                                        .SBTTL  OUTWAI--WAIT FOR RDYO TO GO AWAY
759
760  017040                     OUTWAI: SPBR    IBUS,OCON,SPO           ;READ OUTPUT CONTROL CSR
(1)          000247                     MICPC=MICPC+1
(1)  017040  123440                     <MOVE!SPBRX!IBUS!OCON!SPO>
765  017042                             BR7     IDLE
(1)          000250                     MICPC=MICPC+1
(1)  017042  103447                     <JUMP!BR7CON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
767  017044                             BRWRTE  IMM,100         ;CLEAR CONTROL BITS
(1)          000251                     MICPC=MICPC+1
(1)  017044  000500                     <MOVE!WRTEBR!IMM!<100>>
768  017046                             OUT     BR,OOCON!AANDB
(1)          000252                     MICPC=MICPC+1
(1)  017046  061262                     <MOVE!WROUTX!BR!<OOCON!AANDB>>
769  017050                             ALWAYS  INS13
(1)          000253                     MICPC=MICPC+1
(1)  017050  100666                     <JUMP!ALCOND!<INS13-INIT&3000*4>!<INS13-INIT&777/2>>
770                                     .SBTTL  CTLSRV--CNTL I SERVICE
771  017052                     CTLSRV: SPBR    IBUS,PORT4,SPO                  ;TO SPO
(1)          000254                     MICPC=MICPC+1
(1)  017052  123560                     <MOVE!SPBRX!IBUS!PORT4!SPO>
772  017054                             BRSHFT
(1)          000255                     MICPC=MICPC+1
(1)  017054  001620                     <MOVE!SHFTBR!WRTEBR!SELB>
773  017056                             BR1     HDSEL                           ;IF SET IS HALF DUPLEX
(1)          000256                     MICPC=MICPC+1
(1)  017056  102676                     <JUMP!BR1CON!<HDSEL-INIT&3000*4>!<HDSEL-INIT&777/2>>
774  017060                             OUTPUT  IMM,<100!OMODEM>                ;MASK DTR, TURN OFF HDX
(1)          000257                     MICPC=MICPC+1
(1)  017060  002113                     <MOVE!WROUT!IMM!<100!OMODEM>>
775  017062                     INS11:  BRWRTE  DP,<SELA!SPO>                   ;RESTORE THE CNTL WORD
(1)          000260                     MICPC=MICPC+1
(1)  017062  060600                     <MOVE!WRTEBR!DP!<SELA!SPO>>
776  017064                             BR0     CBOOT                           ;IF SET IS BOOT
(1)          000261                     MICPC=MICPC+1
(1)  017064  116365                     <JUMP!BR0CON!<CBOOT-INIT&3000*4>!<CBOOT-INIT&777/2>>
777  017066                     INS12:  SP      IBUS,INCON,SPO          ;READ THE INPUT CONTROL CSR
(1)          000262                     MICPC=MICPC+1
(1)  017066  123000                     <MOVE!SPX!IBUS!INCON!SPO>
778  017070                             BRWRTE  IMM,100                 ;ZERO THE BR REGISTER EXCEPT INT ENABLE
(1)          000263                     MICPC=MICPC+1
(1)  017070  000500                     <MOVE!WRTEBR!IMM!<100>>
779  017072                             OUT     BR,<AANDB!OINCON>               ;CLEAR IN CONTROL CSR
(1)          000264                     MICPC=MICPC+1
(1)  017072  061260                     <MOVE!WROUTX!BR!<AANDB!OINCON>>
780  017074                             LDMA    IMM,PRTST                       ;ADDRESS PORT STATE
(1)          000265                     MICPC=MICPC+1
(1)  017074  010211                     <MOVE!LDMAR!IMM!<PRTST&377>>
781  017076                     INS13:  PSTATE  NIDLE2
(1)  017076                             MEM     IMM,<<NIDLE2-INIT&777/2>>
(2)          000266                     MICPC=MICPC+1
(2)  017076  002472                     <MOVE!WRMEM!IMM!<<NIDLE2-INIT&777/2>>>
782  017100                             ALWAYS  IDLE
(1)          000267                     MICPC=MICPC+1
(1)  017100  100447                     <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
783                                     ;
784  017102                     TABMXT: SP      IBUS,UBBR,SPO           ;READ BR CONTROL
```

```
(1)             000270
(1)  017102     123220
785  017104
(1)             000271
(1)  017104     000515
786  017106
(1)             000272
(1)  017106     063260
787  017110
(1)             000273
(1)  017110     000404
788  017112
(1)             000274
(1)  017112     061011
789  017114
(1)             000275
(1)  017114     114743
790  017116
(1)             000276
(1)  017116     000500
791  017120
(1)             000277
(1)  017120     063310
792  017122
(1)             000300
(1)  017122     100660
```

```
        MICPC=MICPC+1
        <MOVE!SPX!IBUS!UBBR!SPO>
        BRWRTE  IMM,115                 ;MASK TO CLEAR BR REQ
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<115>>
        SP      BR,AANDB,SPO
        MICPC=MICPC+1
        <MOVE!SPX!BR!AANDB!SPO>
        BRWRTE  IMM,4                   ;INCREMENT MXT
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<4>>
        OUT     BR,ADD!OBR
        MICPC=MICPC+1
        <MOVE!WROUTX!BR!<ADD!OBR>>
        ALWAYS  ECX
        MICPC=MICPC+1
        <JUMP!ALCOND!<ECX-INIT&3000*4>!<ECX-INIT&777/2>>
HDSEL:  BRWRTE  IMM,100                 ;HD MASK TO BR
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<100>>
        SP      BR,AORB,SP10            ;UPDATE PORT STATUS WORD
        MICPC=MICPC+1
        <MOVE!SPX!BR!AORB!SP10>
        ALWAYS  INS11
        MICPC=MICPC+1
        <JUMP!ALCOND!<INS11-INIT&3000*4>!<INS11-INIT&777/2>>
```

```
                                .SBTTL  TBASRV--TRANSMITTER BUFFER ADDRESS SERVICE
794
795  017124                     TBASRV: LDMA    IMM,ETC                         ;GET POINTER TO END OF TMT CHAIN
(1)                                     MICPC=MICPC+1
(1)  017124  000301                     <MOVE!LDMAR!IMM!<ETC&377>>
(1)  017124  010070
796  017126                             LDMA    MEMX,<SELB!SPX!SPO>             ;FIND THE LINK
(1)                                     MICPC=MICPC+1
(1)  017126  000302                     <MOVE!LDMAR!MEMX!<SELB!SPX!SPO>>
(1)  017126  053220
797  017130                             MEMINC  IMM,1                           ;BUFFER ASSIGNED IN IN LINK FLAGS
(1)                                     MICPC=MICPC+1
(1)  017130  000303                     <MOVE!WRMEM!INCMAR!IMM!<1>>
(1)  017130  016401
798  017132                             BRWRTE  <IMM!INCMAR>,TML8               ;POINT PAST NUMBER FIELD
(1)                                     MICPC=MICPC+1
(1)  017132  000304                     <MOVE!WRTEBR!IMM!INCMAR!<TML8>>
(1)  017132  014543
799                                                                             ;SET BR FOR ADDITION TO SPO
800  017134                             MEMINC  IBUS,PORT1
(1)                                     MICPC=MICPC+1
(1)  017134  000305                     <MOVE!WRMEM!INCMAR!IBUS!<PORT1>>
(1)  017134  136500
801  017136                             MEMINC  IBUS,PORT2
(1)                                     MICPC=MICPC+1
(1)  017136  000306                     <MOVE!WRMEM!INCMAR!IBUS!<PORT2>>
(1)  017136  136520
802  017140                             MEMINC  IBUS,PORT4
(1)                                     MICPC=MICPC+1
(1)  017140  000307                     <MOVE!WRMEM!INCMAR!IBUS!<PORT4>>
(1)  017140  136560
803  017142                             MEMINC  IBUS,PORT3
(1)                                     MICPC=MICPC+1
(1)  017142  000310                     <MOVE!WRMEM!INCMAR!IBUS!<PORT3>>
(1)  017142  136540
804  017144                             LDMA    IMM,ETC
(1)                                     MICPC=MICPC+1
(1)  017144  000311                     <MOVE!LDMAR!IMM!<ETC&377>>
(1)  017144  010070
805  017146                             MEM     IMM,TML1                        ;ASSUME QUEUE WRAP AROUND
(1)                                     MICPC=MICPC+1
(1)  017146  000312                     <MOVE!WRMEM!IMM!<TML1>>
(1)  017146  002471
806  017150                             CMP     BR,SPO                          ;END OF CHAIN?
(1)                                     MICPC=MICPC+1
(1)  017150  000313                     <SUBTC!BR!SPO>
(1)  017150  060360
807  017152                             Z       10$                             ;IF YES--BRANCH
(1)                                     MICPC=MICPC+1
(1)  017152  000314                     <JUMP!ZCOND!<10$-INIT&3000*4>!<10$-INIT&777/2>>
(1)  017152  101717
808  017154                             BRWRTE  IMM,6                           ;QUEUE ENTRY LENGTH
(1)                                     MICPC=MICPC+1
(1)  017154  000315                     <MOVE!WRTEBR!IMM!<6>>
(1)  017154  000406
809  017156                             MEM     BR,ADD!SPO                      ;UPDATE THE END POINTER IN MEMORY
(1)                                     MICPC=MICPC+1
(1)  017156  000316                     <MOVE!WRMEM!BR!<ADD!SPO>>
(1)  017156  062400
810  017160                     10$:    BRWRTE  IMM,2                           ;NUMBERED MSG PENDING MASK
(1)                                     MICPC=MICPC+1
(1)  017160  000317                     <MOVE!WRTEBR!IMM!<2>>
(1)  017160  000402
811  017162                             SP      BR,AORB,SP10                    ;UPDATE LINE STATUS
(1)                                     MICPC=MICPC+1
(1)  017162  000320                     <MOVE!SPX!BR!AORB!SP10>
(1)  017162  063310
812  017164                             ALWAYS  INS12
(1)                                     MICPC=MICPC+1
(1)  017164  000321                     <JUMP!ALCOND!<INS12-INIT&3000*4>!<INS12-INIT&777/2>>
(1)  017164  100662
813                                 .SBTTL  RBASRV--RECEIVE BUFFER ADDRESS SERVICE
814  017166                     RBASRV: LDMA    IMM,ERC                         ;ADDRES END OF RECEIVE CHAIN
(1)                                     MICPC=MICPC+1
(1)  017166  000322
```

```
    (1)  017166  010023                <MOVE!LDMAR!IMM!<ERC&377>>
815  017170                            LDMA    MEMX,<SELB!SPX!SPO>          ;GET THE POINTER TO LINK
    (1)          000323                MICPC=MICPC+1
    (1)  017170  053220                <MOVE!LDMAR!MEMX!<SELB!SPX!SPO>>
816  017172                            MEMINC  IMM,1
    (1)          000324                MICPC=MICPC+1
    (1)  017172  016401                <MOVE!WRMEM!INCMAR!IMM!<1>>
817  017174                            MEMINC  IBUS,PORT1
    (1)          000325                MICPC=MICPC+1
    (1)  017174  136500                <MOVE!WRMEM!INCMAR!IBUS!<PORT1>>
818  017176                            MEMINC  IBUS,PORT2
    (1)          000326                MICPC=MICPC+1
    (1)  017176  136520                <MOVE!WRMEM!INCMAR!IBUS!<PORT2>>
819  017200                            MEMINC  IBUS,PORT4
    (1)          000327                MICPC=MICPC+1
    (1)  017200  136560                <MOVE!WRMEM!INCMAR!IBUS!<PORT4>>
820  017202                            MEMINC  IBUS,PORT3
    (1)          000330                MICPC=MICPC+1
    (1)  017202  136540                <MOVE!WRMEM!INCMAR!IBUS!<PORT3>>
821                                    ;;;;NOTE INVERTED ORDER OF PORT 3 AND PORT4
822  017204                            LDMA    IMM,ERC
    (1)          000331                MICPC=MICPC+1
    (1)  017204  010023                <MOVE!LDMAR!IMM!<ERC&377>>
823  017206                            MEM     IMM,RCL1                     ;ASSUME WRAP AROUND CASE
    (1)          000332                MICPC=MICPC+1
    (1)  017206  002424                <MOVE!WRMEM!IMM!<RCL1>>
824  017210                            BRWRTE  IMM,RCL7                     ;GET ADDRESS OF END OF CAHIN AREA
    (1)          000333                MICPC=MICPC+1
    (1)  017210  000462                <MOVE!WRTEBR!IMM!<RCL7>>
825  017212                            CMP     BR,SPO                               ;CHECK FOR END
    (1)          000334                MICPC=MICPC+1
    (1)  017212  060360                <SUBTC!BR!SPO>
826  017214                            Z       INS12                                ;IF EQUAL BRANCH
    (1)          000335                MICPC=MICPC+1
    (1)  017214  101662                <JUMP!ZCOND!<INS12-INIT&3000*4>!<INS12-INIT&777/2>>
827  017216                            BRWRTE  IMM,5                        ;CALCULATE ADDRESS OF NEXT LINK
    (1)          000336                MICPC=MICPC+1
    (1)  017216  000405                <MOVE!WRTEBR!IMM!<5>>
828  017220                            MEM     BR,ADD!SPO                   ;..
    (1)          000337                MICPC=MICPC+1
    (1)  017220  062400                <MOVE!WRMEM!BR!<ADD!SPO>>
829  017222                            ALWAYS  INS12                                ;EXIT
    (1)          000340                MICPC=MICPC+1
    (1)  017222  100662                <JUMP!ALCOND!<INS12-INIT&3000*4>!<INS12-INIT&777/2>>
830  017224                 RA1:       BRWRTE  IMM,357                      ;MASK TO CLEAR CLR ACTIVE
    (1)          000341                MICPC=MICPC+1
    (1)  017224  000757                <MOVE!WRTEBR!IMM!<357>>
831  017226                            SPBR    BR,AANDB,SP10                ;CLEAR BIT IN LINE STATUS WORD
    (1)          000342                MICPC=MICPC+1
    (1)  017226  063670                <MOVE!SPBRX!BR!AANDB!SP10>
832  017230                            BRSHFT
    (1)          000343                MICPC=MICPC+1
    (1)  017230  001620                <MOVE!SHFTBR!WRTEBR!SELB>
833  017232                            BR4     FLUSH
    (1)          000344                MICPC=MICPC+1
    (1)  017232  107015                <JUMP!BR4CON!<FLUSH-INIT&3000*4>!<FLUSH-INIT&777/2>>
```

```
834  017234
(1)            000345
(1)  017234    000400
835  017236
(1)            000346
(1)  017236    063233
836  017240
(1)            000347
(1)  017240    000427
837  017242
(1)            000350
(1)  017242    104425
838
```

```
RA3:    BRWRTE  IMM,0                          ;CLEAR BR
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<0>>
        SP      BR,SELB,SP13                   ;SET NUMB MESSAGE TYPE IN SP13
        MICPC=MICPC+1
        <MOVE!SPX!BR!SELB!SP13>
        STATE   RCVB                           ;CHANGE RECEIVE STATE POINTER TO STATE B
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<RCVB-INIT&777/2>>
        ALWAYS  REXIT
        MICPC=MICPC+1
        <JUMP!ALCOND!<REXIT-INIT&3000*4>!<REXIT-INIT&777/2>>
        ;
```

RBASRV--RECEIVE BUFFER ADDRESS SERVICE

```
840  017244
(1)            000351
(1)  017244    000700
841  017246
(1)            000352
(1)  017246    123220
842  017250
(1)            000353
(1)  017250    061311
844  017252
(1)            000354
(1)  017252    100447
849
850  017254
(1)            000355
(1)  017254    000676
851
852  017256
(1)            000356
(1)  017256    063223
853  017260
(1)            000357
(1)  017260    000402
854  017262
(1)            000360
(1)  017262    061231
855  017264
(1)            000361
(1)  017264    120620
856  017266
(1)            000362
(1)  017266    102761
857  017270
(1)            000363
(1)  017270    174603
858
859  017272
(1)            000364
(1)  017272    120600
860  017274
(1)            000365
(1)  017274    102047
861  017276
(1)            000366
(1)  017276    114734
862
863
864  017300
(1)            000367
(1)  017300    000404
865  017302
(1)            000370
(1)  017302    114637
866  017304
(1)            000371
(1)  017304    000704
```

```
PE1:    BRWRTE  IMM,300                 ;MASK FOR INTERRUPT AND VECTOR
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<300>>
        SP      IBUS,UBBR,SPO           ;READ BR CONTROL REG
        MICPC=MICPC+1
        <MOVE!SPX!IBUS!UBBR!SPO>
        OUT     BR,<AORB!OBR>           ;INTERRUPT
        MICPC=MICPC+1
        <MOVE!WROUTX!BR!<AORB!OBR>>
        ALWAYS  IDLE
        MICPC=MICPC+1
        <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
;
HALTED: BRADDR  EM6
        MICPC=MICPC+1
        <MOVE!WRTEBR!<EM6-INIT&777/2>>
                                        ;FALL INTO ACLOW
ACLOW:  SP      BR,SELB,SP3             ;CHANGE RECEIVE STATE POINTER
        MICPC=MICPC+1
        <MOVE!SPX!BR!SELB!SP3>
        BRWRTE  IMM,2                   ;CAUSE AN AC LOW
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<2>>
        OUT     BR,<SELB!OBR>
        MICPC=MICPC+1
        <MOVE!WROUTX!BR!<SELB!OBR>>
5$:     BRWRTE  IBUS,UBBR               ;WAIT FOR IT TO COMPLETE
        MICPC=MICPC+1
        <MOVE!WRTEBR!IBUS!<UBBR>>
        BR1     5$
        MICPC=MICPC+1
        <JUMP!BR1CON!<5$-INIT&3000*4>!<5$-INIT&777/2>>
        .ALWAY  BR,SELA,SP3!PAGE3
        MICPC=MICPC+1
        <JUMP!ALCOND!BR!SELA!SP3!PAGE3>
;
;BU1:    BRWRTE  IBUS,NPR
        MICPC=MICPC+1
        <MOVE!WRTEBR!IBUS!<NPR>>
        BRO     IDLE
        MICPC=MICPC+1
        <JUMP!BROCON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
        ALWAYS  EC2
        MICPC=MICPC+1
        <JUMP!ALCOND!<EC2-INIT&3000*4>!<EC2-INIT&777/2>>
;
;OVRRUN: BRWRTE  IMM,4
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<4>>
        ALWAYS  NTRSO
        MICPC=MICPC+1
        <JUMP!ALCOND!<NTRSO-INIT&3000*4>!<NTRSO-INIT&777/2>>
HEH1:   STATE   TEOM
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<TEOM-INIT&777/2>>
```

```
867  017306                          ALWAYS  XEXIT
(1)            000372                MICPC=MICPC+1
(1)  017306    110556                <JUMP!ALCOND!<XEXIT-INIT&3000*4>!<XEXIT-INIT&777/2>>
868  017310                ACK1:     BRWRTE  IMM,327
(1)            000373                MICPC=MICPC+1
(1)  017310    000727                <MOVE!WRTEBR!IMM!<327>>
869  017312                          SP      BR,AANDB,SP10
(1)            000374                MICPC=MICPC+1
(1)  017312    063270                <MOVE!SPX!BR!AANDB!SP10>
870  017314                          ALWAYS  RD5
(1)            000375                MICPC=MICPC+1
(1)  017314    104504                <JUMP!ALCOND!<RD5-INIT&3000*4>!<RD5-INIT&777/2>>
872  017316                RK4:      BRO     IDLE
(1)            000376                MICPC=MICPC+1
(1)  017316    102047                <JUMP!BROCON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
873  017320                          ALWAYS  RK5
(1)            000377                MICPC=MICPC+1
(1)  017320    104647                <JUMP!ALCOND!<RK5-INIT&3000*4>!<RK5-INIT&777/2>>
875
```

```
                                .=INIT+1000
                                MICPC=377
                                .SBTTL  RCVA--ROUTINE TO HANDLE FIRST DDCMP CHARACTER
                                ;ENTERED FROM IDLE LOOP
                                ;DETERMINES IF MESSAGE TYPE IS NUMBERED,UNNUMBERED OR BOOT
                                ;SETS UP APROPRIATE STATES FOR REST OF MESSAGE.
017322                  RCVA:   SP      IBUS,RCVDAT,SPO         ;READ RECEIVE CHARACTER TO SPO
(1)     000400                  MICPC=MICPC+1
(1)     017322  023200          <MOVE!SPX!IBUS!RCVDAT!SPO>
017324                          BRWRTE  BR,SELA!SP1             ;READ PORT STATUS WORD
(1)     000401                  MICPC=MICPC+1
(1)     017324  060601          <MOVE!WRTEBR!BR!<SELA!SP1>>
017326                          BRO     5$                      ;IF INIT MODE---ONLY BOOT OK
(1)     000402                  MICPC=MICPC+1
(1)     017326  106012          <JUMP!BROCON!<5$-INIT&3000*4>!<5$-INIT&777/2>>
017330                          BR7     5$                      ;IF BOOT MODE---ONLY BOOT OK
(1)     000403                  MICPC=MICPC+1
(1)     017330  107412          <JUMP!BR7CON!<5$-INIT&3000*4>!<5$-INIT&777/2>>
017332                          BRWRTE  IMM,201                 ;SOH TO BR
(1)     000404                  MICPC=MICPC+1
(1)     017332  000601          <MOVE!WRTEBR!IMM!<201>>
017334                          CMP     BR,SPO                  ;COMPARE BR TO SPO
(1)     000405                  MICPC=MICPC+1
(1)     017334  060360          <SUBTC!BR!SPO>
017336                          Z       RA1                     ;IF EQUAL-IS NUMBERED MESSAGE
(1)     000406                  MICPC=MICPC+1
(1)     017336  101741          <JUMP!ZCOND!<RA1-INIT&3000*4>!<RA1-INIT&777/2>>
017340                          BRWRTE  IMM,5                   ;ENQ TO BR
(1)     000407                  MICPC=MICPC+1
(1)     017340  000405          <MOVE!WRTEBR!IMM!<5>>
017342                          CMP     BR,SPO                  ;COMPARE ENQ TO SPO
(1)     000410                  MICPC=MICPC+1
(1)     017342  060360          <SUBTC!BR!SPO>
017344                          Z       RA2                     ;IF EQUAL-IS UNNUMBERED MESSAGE
(1)     000411                  MICPC=MICPC+1
(1)     017344  105424          <JUMP!ZCOND!<RA2-INIT&3000*4>!<RA2-INIT&777/2>>
017346                  5$:     BRWRTE  IMM,220                 ;DLE TO BR
(1)     000412                  MICPC=MICPC+1
(1)     017346  000620          <MOVE!WRTEBR!IMM!<220>>
017350                          CMP     BR,SPO                  ;COMPARE DLE TO SPO
(1)     000413                  MICPC=MICPC+1
(1)     017350  060360          <SUBTC!BR!SPO>
017352                          Z       BOOT                    ;IF EQUAL IS BOOT
(1)     000414                  MICPC=MICPC+1
(1)     017352  105771          <JUMP!ZCOND!<BOOT-INIT&3000*4>!<BOOT-INIT&777/2>>
017354                  FLUSH:  OUTPUT  IMM,<200!ORCVCO>        ;FLUSH INPUT SILO
(1)     000415                  MICPC=MICPC+1
(1)     017354  002212          <MOVE!WROUT!IMM!<200!ORCVCO>>
                                                                ;(LOW ORDER BITS READ ONLY)
017356                          RSTATE  RCVA                    ;SET STATE TO RCVA
(1)     000416                  MICPC=MICPC+1
(1)     017356  000400          <MOVE!WRTEBR!IMM!<RCVA-INIT&777/2>>
(1)     000417                  MICPC=MICPC+1
(1)     017360  063223          <MOVE!SPX!BR!SELB!SP3>
017362                          BRWRTE  IMM,357                 ;MASK TO CLEAR--CLEAR ACTIVE
(1)     000420                  MICPC=MICPC+1
```

```
(1)  017362  000757
900  017364
(1)          000421
(1)  017364  063270
901  017366
(1)          000422
(1)  017366  060070
902  017370
(1)          000423
(1)  017370  110651
903  017372
(1)          000424
(1)  017372  000703
904  017374
(1)          000425
(1)  017374  063223
905  017376
(1)          000426
(1)  017376  100447
```

```
        <MOVE!WRTEBR!IMM!<357>>
        SP      BR,AANDB,SP10           ;IN LINE STATUS WORD
        MICPC=MICPC+1
        <MOVE!SPX!BR!AANDB!SP10>
FLUSHA: NOP     BR,INCA,SP10            ;CLEAR   "C" BIT FOR TIMEO
        MICPC=MICPC+1
        <BR!INCA!SP10>
        ALWAYS  TIMEO                   ;SEE IF WE CAN SEND AN ACK
        MICPC=MICPC+1
        <JUMP!ALCOND!<TIMEO-INIT&3000*4>!<TIMEO-INIT&777/2>>
RA2:    STATE   RCVI                    ;CHANGE RECEIVE STATE TO I
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<RCVI-INIT&777/2>>
REXIT:  SP      BR,SELB,SP3
        MICPC=MICPC+1
        <MOVE!SPX!BR!SELB!SP3>
        ALWAYS  IDLE
        MICPC=MICPC+1
        <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
```

```
                                .SBTTL  RCVB--ROUTINE TO HANDLE FIRST CHARACTER OF COUNT FIELD
                                ;ENTERED FROM IDLE LOOP
                                ;STORES COUNT FIELD AND SETS UP RCVC AS NEXT STATE
                                ;
017400                  RCVB:
017400                          SP      IBUS,RCVDAT,SP4         ;READ CHARACTER TO SP4
                                MICPC=MICPC+1
                                <MOVE!SPX!IBUS!RCVDAT!SP4>
017402                          LDMA    BR,<SELA!SP15>                  ;LOAD MAR WITH ADDRESS OF CURRENT BA
                                MICPC=MICPC+1
                                <MOVE!LDMAR!BR!<SELA!SP15>>
017404                          BRWRTE  MEMX!INCMAR,SELB        ;READ THE FLAGS BYTE
                                MICPC=MICPC+1
                                <MOVE!WRTEBR!MEMX!INCMAR!<SELB>>
017406                          BRO     RB1                     ;BUFFER IS ASSIGNED
                                MICPC=MICPC+1
                                <JUMP!BROCON!<RB1-INIT&3000*4>!<RB1-INIT&777/2>>
017410                          BRWRTE  BR,SELA!SP1             ;NO BUFFER ASSIGNED - MAINT MODE?
                                MICPC=MICPC+1
                                <MOVE!WRTEBR!BR!<SELA!SP1>>
017412                          BR7     RB3                     ;YES
                                MICPC=MICPC+1
                                <JUMP!BR7CON!<RB3-INIT&3000*4>!<RB3-INIT&777/2>>
017414                          BRWRTE  IMM,310                 ;SET NAK REASON CODE
                                MICPC=MICPC+1
                                <MOVE!WRTEBR!IMM!<310>>
017416                          LDMA    IMM,NTLS                ;ADDRESS CUMULATIVE COUNTER
                                MICPC=MICPC+1
                                <MOVE!LDMAR!IMM!<NTLS&377>>
017420                          ALWAYS  RH5                     ;SEND A NAK
                                MICPC=MICPC+1
                                <JUMP!ALCOND!<RH5-INIT&3000*4>!<RH5-INIT&777/2>>
017422                  RB3:    BRWRTE  IMM,2                   ;FLAG FOR NO BUF ASSIGNED IN MAINT MODE
                                MICPC=MICPC+1
                                <MOVE!WRTEBR!IMM!<2>>
017424                          SP      BR,AORB,SP1             ;SET THE FLAG
                                MICPC=MICPC+1
                                <MOVE!SPX!BR!AORB!SP1>
017426                  RB1:    STATE   RCVC
                                MICPC=MICPC+1
                                <MOVE!WRTEBR!IMM!<RCVC-INIT&777/2>>
017430                  RB0:    SP      BR,SELB,SP3
                                MICPC=MICPC+1
                                <MOVE!SPX!BR!SELB!SP3>
017432                          OUTPUT  <MEMX!INCMAR>,<SELB!OBA1>       ;OUTPUT LOW ORDER BYTE OF ADDRESS
                                MICPC=MICPC+1
                                <MOVE!WROUT!MEMX!INCMAR!<SELB!OBA1>>
017434                          OUTPUT  MEMX!INCMAR,<SELB!OBA2> ;OUTPUT HIGH BYTE OF ADDRESS
                                MICPC=MICPC+1
                                <MOVE!WROUT!MEMX!INCMAR!<SELB!OBA2>>
017436                          SP      IBUS,UBBR,SP0           ;READ THE BUS REQ REGISTER
                                MICPC=MICPC+1
                                <MOVE!SPX!IBUS!UBBR!SP0>
017440                          BRWRTE  IMM,101                 ;MASK OFF ALL BUT NXM AND VEC4 BITS
                                MICPC=MICPC+1
                                <MOVE!WRTEBR!IMM!<101>>
```

```
  929  017442                                  SP      BR,AANDB,SPO            ;AND SAVE IN SPO
  (1)            000450                        MICPC=MICPC+1
  (1)  017442    063260                        <MOVE!SPX!BR!AANDB!SPO>
  930  017444                                  SP      IMM,300,SP5             ;MASK TO ISOLATE EX. MEM BITS
  (1)            000451                        MICPC=MICPC+1
  (1)  017444    003305                        <MOVE!SPX!IMM!300!SP5>
  931                                                                          ;NOTE THIS REALLY WRITES A 305 BUT THE
  932                                                                          ;5 GETS SHIFTED OUT
  933  017446                                  BRWRTE  MEMX,AANDB!SP5          ;MASK ALL BUT EX. MEM BITS
  (1)            000452                        MICPC=MICPC+1
  (1)  017446    040665                        <MOVE!WRTEBR!MEMX!<AANDB!SP5>>
  934  017450                                  BRSHFT                          ;SHIFT THEM INTO THE CORRECT POSITION
  (1)            000453                        MICPC=MICPC+1
  (1)  017450    001620                        <MOVE!SHFTBR!WRTEBR!SELB>
  935  017452                                  BRSHFT
  (1)            000454                        MICPC=MICPC+1
  (1)  017452    001620                        <MOVE!SHFTBR!WRTEBR!SELB>
  936  017454                                  BRSHFT
  (1)            000455                        MICPC=MICPC+1
  (1)  017454    001620                        <MOVE!SHFTBR!WRTEBR!SELB>
  937  017456                                  BRSHFT
  (1)            000456                        MICPC=MICPC+1
  (1)  017456    001620                        <MOVE!SHFTBR!WRTEBR!SELB>
  938  017460                        RB4:      OUT     BR,AORB!OBR                     ;WRITE EX MEM BITS OUT
  (1)            000457                        MICPC=MICPC+1
  (1)  017460    061311                        <MOVE!WROUTX!BR!<AORB!OBR>>
  939  017462                                  ALWAYS  IDLE
  (1)            000460                        MICPC=MICPC+1
  (1)  017462    100447                        <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
  940  017464                        RB2:      ALWAYS  I2
  (1)            000461                        MICPC=MICPC+1
  (1)  017464    100453                        <JUMP!ALCOND!<I2-INIT&3000*4>!<I2-INIT&777/2>>
  941                                          .SBTTL  RCVC--ROUTINE TO HANDLE SECOND CHARACTER OF COUNT FIELD, SELECT AND FINA
  942                                          ;ENTERED FROM IDLE LOOP
  943                                          ;INTERPRETS SELECT AND FINAL
  944                                          ;CHECKS FOR COUNT TOO LARGE
  945                                          ;
  946  017466                        RCVC:
  951  017466                                  SP      IBUS,RCVDAT,SP5         ;GET CHARACTER
  (1)            000462                        MICPC=MICPC+1
  (1)  017466    023205                        <MOVE!SPX!IBUS!RCVDAT!SP5>
  953  017470                                  LDMA    IMM,BC                  ;LOAD MAR TO BYTE COUNT
  (1)            000463                        MICPC=MICPC+1
  (1)  017470    010167                        <MOVE!LDMAR!IMM!<BC&377>>
  954  017472                                  MEMINC  BR,SELA!SP4             ;SAVE LOW BYTE
  (1)            000464                        MICPC=MICPC+1
  (1)  017472    076604                        <MOVE!WRMEM!INCMAR!BR!<SELA!SP4>>
  955  017474                                  MEMINC  BR,SELA!SP5             ;AND NOW HIGH BYTE
  (1)            000465                        MICPC=MICPC+1
  (1)  017474    076605                        <MOVE!WRMEM!INCMAR!BR!<SELA!SP5>>
  956  017476                                  STATE   RCVD                    ;SET NEXT STATE TO D
  (1)            000466                        MICPC=MICPC+1
  (1)  017476    000470                        <MOVE!WRTEBR!IMM!<RCVD-INIT&777/2>>
  957  017500                                  ALWAYS  REXIT
  (1)            000467                        MICPC=MICPC+1
  (1)  017500    104425                        <JUMP!ALCOND!<REXIT-INIT&3000*4>!<REXIT-INIT&777/2>>
```

```
                                .SBTTL  RCVD--ROUTINE TO HANDLE RESPONSE FIELD FOR NUMBERED MESSAGES
                                ;
959                             ;
960                             ;
961  017502                     RCVD:   STATE   RCVE
(1)             000470                  MICPC=MICPC+1
(1)  017502     000522                  <MOVE!WRTEBR!IMM!<RCVE-INIT&777/2>>
962  017504                     RD2:    SP      BR,SELB,SP3             ;SAVE THE STATE
(1)             000471                  MICPC=MICPC+1
(1)  017504     063223                  <MOVE!SPX!BR!SELB!SP3>
963  017506                             SPBR    IBUS,RCVDAT,SPO         ;INPUT THE CHARACTER
(1)             000472                  MICPC=MICPC+1
(1)  017506     023600                  <MOVE!SPBRX!IBUS!RCVDAT!SPO>
964  017510                             BRWRTE  BR,SUB!SP17             ;COMPARE NEW R TO LAST R
(1)             000473                  MICPC=MICPC+1
(1)  017510     060757                  <MOVE!WRTEBR!BR!<SUB!SP17>>
965  017512                             BRWRTE  BR,AORNB!SP1            ;IF NEW IS THE SAME OR LESS OR
(1)             000474                  MICPC=MICPC+1
(1)  017512     060641                  <MOVE!WRTEBR!BR!<AORNB!SP1>>
966  017514                             BR7     RD7                     ;MAINT MODE - GET OUT
(1)             000475                  MICPC=MICPC+1
(1)  017514     107514                  <JUMP!BR7CON!<RD7-INIT&3000*4>!<RD7-INIT&777/2>>
967  017516                             BRWRTE  BR,SELA!SP10
(1)             000476                  MICPC=MICPC+1
(1)  017516     060610                  <MOVE!WRTEBR!BR!<SELA!SP10>>
968  017520                             BRSHFT
(1)             000477                  MICPC=MICPC+1
(1)  017520     001620                  <MOVE!SHFTBR!WRTEBR!SELB>
969  017522                             BR4     IDLE
(1)             000500                  MICPC=MICPC+1
(1)  017522     103047                  <JUMP!BR4CON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
970  017524                             BRO     RD4                     ;MESSAGE IN PROGRESS
(1)             000501                  MICPC=MICPC+1
(1)  017524     106110                  <JUMP!BROCON!<RD4-INIT&3000*4>!<RD4-INIT&777/2>>
971  017526                     RD3:    LDMA    IMM,ISP17               ;ADDRESS LAST ACKED IMAGE
(1)             000502                  MICPC=MICPC+1
(1)  017526     010153                  <MOVE!LDMAR!IMM!<ISP17&377>>
972  017530                             MEM     BR,SELA!SPO             ;COPY THE CHAR
(1)             000503                  MICPC=MICPC+1
(1)  017530     062600                  <MOVE!WRMEM!BR!<SELA!SPO>>
973  017532                     RD5:    LDMA    IMM,REPST               ;POINT TO REP THRESHOLD COUNTER
(1)             000504                  MICPC=MICPC+1
(1)  017532     010003                  <MOVE!LDMAR!IMM!<REPST&377>>
974  017534                             MEM     IMM,1                   ;RESET REP THRESHOLD
(1)             000505                  MICPC=MICPC+1
(1)  017534     002401                  <MOVE!WRMEM!IMM!<1>>
975  017536                     RD6:    SP      IMM,374,SP14            ;RESET THE COUNT
(1)             000506                  MICPC=MICPC+1
(1)  017536     003374                  <MOVE!SPX!IMM!374!SP14>
976  017540                             ALWAYS  IDLE
(1)             000507                  MICPC=MICPC+1
(1)  017540     100447                  <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
977  017542                     RD4:    BRWRTE  BR,SELA!SPO
(1)             000510                  MICPC=MICPC+1
(1)  017542     060600                  <MOVE!WRTEBR!BR!<SELA!SPO>>
978  017544                             BRWRTE  BR,SUBTC!SP12
(1)             000511                  MICPC=MICPC+1
(1)  017544     060772                  <MOVE!WRTEBR!BR!<SUBTC!SP12>>
```

```
 979  017546
(1)             000512
(1)   017546    103447          BR7     IDLE
                                MICPC=MICPC+1
                                <JUMP!BR7CON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
 980  017550                    ALWAYS  RD3
(1)             000513          MICPC=MICPC+1
(1)   017550    104502          <JUMP!ALCOND!<RD3-INIT&3000*4>!<RD3-INIT&777/2>>
 981  017552    RD7:    BRWRTE  BR,SELA!SPO
(1)             000514          MICPC=MICPC+1
(1)   017552    060600          <MOVE!WRTEBR!BR!<SELA!SPO>>
 982  017554                    CMP     BR,SP12
(1)             000515          MICPC=MICPC+1
(1)   017554    060372          <SUBTC!BR!SP12>
 983  017556                    Z       RD6
(1)             000516          MICPC=MICPC+1
(1)   017556    105506          <JUMP!ZCOND!<RD6-INIT&3000*4>!<RD6-INIT&777/2>>
 984  017560                    BRWRTE  BR,AA!SP10                      ;IF HDX
(1)             000517          MICPC=MICPC+1
(1)   017560    060530          <MOVE!WRTEBR!BR!<AA!SP10>>
 985  017562                    BR7     TIME2                           ;QUEUE UP REP
(1)             000520          MICPC=MICPC+1
(1)   017562    113671          <JUMP!BR7CON!<TIME2-INIT&3000*4>!<TIME2-INIT&777/2>>
 986  017564                    ALWAYS  IDLE
(1)             000521          MICPC=MICPC+1
(1)   017564    100447          <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
 987                            .SBTTL  RCVE--ROUTINE TO HANDLE N FIELD OF NUMBERED MESSAGE
 988                            ;
 989  017566    RCVE:   BRWRTE  BR,SELA!SP1             ;READ THE STATUS BYTE
(1)             000522          MICPC=MICPC+1
(1)   017566    060601          <MOVE!WRTEBR!BR!<SELA!SP1>>
 990  017570                    BR7     RCVQ
(1)             000523          MICPC=MICPC+1
(1)   017570    107721          <JUMP!BR7CON!<RCVQ-INIT&3000*4>!<RCVQ-INIT&777/2>>
 991  017572                    BRWRTE  IBUS,RCVDAT             ;INPUT THE CHARACTER
(1)             000524          MICPC=MICPC+1
(1)   017572    020600          <MOVE!WRTEBR!IBUS!<RCVDAT>>
 992  017574                    CMP     BR,SP11
(1)             000525          MICPC=MICPC+1
(1)   017574    060371          <SUBTC!BR!SP11>
 993  017576                    Z       5$
(1)             000526          MICPC=MICPC+1
(1)   017576    105531          <JUMP!ZCOND!<5$-INIT&3000*4>!<5$-INIT&777/2>>
 994  017600                    SP      BR,DECA,SP13            ;FORCE MSG TYPE TO -1
(1)             000527          MICPC=MICPC+1
(1)   017600    063173          <MOVE!SPX!BR!DECA!SP13>
 995  017602                    ALWAYS  RE2
(1)             000530          MICPC=MICPC+1
(1)   017602    104532          <JUMP!ALCOND!<RE2-INIT&3000*4>!<RE2-INIT&777/2>>
 996  017604    5$:     SP      BR,INCA,SP11            ;UPDATE R FIELD
(1)             000531          MICPC=MICPC+1
(1)   017604    063071          <MOVE!SPX!BR!INCA!SP11>
 997  017606    RE2:    STATE   RCVF                    ;NEXT RECEIVE STATE IS F
(1)             000532          MICPC=MICPC+1
(1)   017606    000534          <MOVE!WRTEBR!IMM!<RCVF-INIT&777/2>>
 998  017610                    ALWAYS  REXIT
(1)             000533          MICPC=MICPC+1
(1)   017610    104425          <JUMP!ALCOND!<REXIT-INIT&3000*4>!<REXIT-INIT&777/2>>
```

```
 999
1000  017612
 (1)            000534
 (1)  017612    063164
1001  017614
 (1)            000535
 (1)  017614    105137
1002  017616
 (1)            000536
 (1)  017616    063165
1003  017620
 (1)            000537
 (1)  017620    000542
1004  017622
 (1)            000540
 (1)  017622    020200
1005  017624
 (1)            000541
 (1)  017624    104425
1006
1007
1008
1009  017626
 (1)            000542
 (1)  017626    000544
1010  017630
 (1)            000543
 (1)  017630    104540
```

RCVF--ROUTINE TO IGNORE ADDRESS

```
                .SBTTL  RCVF--ROUTINE TO IGNORE ADDRESS
RCVF:   SP      BR,DECA,SP4             ;DECREMENTLOW BYTE OF COUNT
        MICPC=MICPC+1
        <MOVE!SPX!BR!DECA!SP4>
        C       RCVF0                   ;NO OVERFLOW
        MICPC=MICPC+1
        <JUMP!CCOND!<RCVF0-INIT&3000*4>!<RCVF0-INIT&777/2>>
        SP      BR,DECA,SP5             ;OVERFLOW - DECREMENT HIGH BYTE
        MICPC=MICPC+1
        <MOVE!SPX!BR!DECA!SP5>
RCVF0:  STATE   RCVG
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<RCVG-INIT&777/2>>
RCVF1:  NOP     IBUS,RCVDAT,0           ;INPUT CHARACTER - AND DISCARD
        MICPC=MICPC+1
        <IBUS!RCVDAT!0>
        ALWAYS  REXIT
        MICPC=MICPC+1
        <JUMP!ALCOND!<REXIT-INIT&3000*4>!<REXIT-INIT&777/2>>
        ;
        .SBTTL  RCVG--ROUTINE TO IGNORE CRC1
        ;
RCVG:   STATE   RCVH                    ;NEXT STATE IS RCVH
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<RCVH-INIT&777/2>>
        ALWAYS  RCVF1
        MICPC=MICPC+1
        <JUMP!ALCOND!<RCVF1-INIT&3000*4>!<RCVF1-INIT&777/2>>
```

```
                        .SBTTL  RCVH--ROUTINE TO HANDLE CRC2 AND TO DISPATCH NUMBERED AND UNNUMBERED TYP
                        ;
1012
1013
1014  017632    RCVH:
1015  017632              SP      IBUS,RCVDAT,SPO                 ;GET CHAR IN SPO
 (1)          000544      MICPC=MICPC+1
 (1)  017632  023200      <MOVE!SPX!IBUS!RCVDAT!SPO>
1016  017634              BRWRTE  IBUS,RCVCON                     ;READ RECVR CONTROL REGISTER
 (1)          000545      MICPC=MICPC+1
 (1)  017634  020640      <MOVE!WRTEBR!IBUS!<RCVCON>>
1017  017636              BRO     TDON1                           ;IF BCC MATCH SET CRC IS GOOD
 (1)          000546      MICPC=MICPC+1
 (1)  017636  116142      <JUMP!BROCON!<TDON1-INIT&3000*4>!<TDON1-INIT&777/2>>
1018  017640              BRWRTE  BR,SELA!SP1                     ;READ STATUS BYTE
 (1)          000547      MICPC=MICPC+1
 (1)  017640  060601      <MOVE!WRTEBR!BR!<SELA!SP1>>
1019  017642              BR7     RHX                             ;MAINT MODE
 (1)          000550      MICPC=MICPC+1
 (1)  017642  107753      <JUMP!BR7CON!<RHX-INIT&3000*4>!<RHX-INIT&777/2>>
1020  017644              BRWRTE  DP,<SELA!SP10>                  ;READ PORT STATUS WORD TO BR
 (1)          000551      MICPC=MICPC+1
 (1)  017644  060610      <MOVE!WRTEBR!DP!<SELA!SP10>>
1021  017646              BRSHFT
 (1)          000552      MICPC=MICPC+1
 (1)  017646  001620      <MOVE!SHFTBR!WRTEBR!SELB>
1022  017650              BR4     SNAK1                           ;IF START MODE--PROCEED TO RESEND START
 (1)          000553      MICPC=MICPC+1
 (1)  017650  117263      <JUMP!BR4CON!<SNAK1-INIT&3000*4>!<SNAK1-INIT&777/2>>
1023  017652              LDMA    IMM,ISP17                       ;ELSE BCC ERROR--POINT TO IMAGE OF SP17 IN RAM
 (1)          000554      MICPC=MICPC+1
 (1)  017652  010153      <MOVE!LDMAR!IMM!<ISP17&377>>
1024  017654              MEM     BR,SELA!SP17                    ;RESTORE LAST ACKED IMAGE
 (1)          000555      MICPC=MICPC+1
 (1)  017654  062617      <MOVE!WRMEM!BR!<SELA!SP17>>
1025  017656              BRWRTE  IMM,301                         ;WRITE HEADER BCC ERROR SUBTYPE
 (1)          000556      MICPC=MICPC+1
 (1)  017656  000701      <MOVE!WRTEBR!IMM!<301>>
1026  017660              LDMA    IMM,NHDS                        ;ADDRESS CUM ERROR COUNTER
 (1)          000557      MICPC=MICPC+1
 (1)  017660  010013      <MOVE!LDMAR!IMM!<NHDS&377>>
1027  017662    RH5:      SP      MEMX,SELB,SPO                   ;WRITE IT TO SPO
 (1)          000560      MICPC=MICPC+1
 (1)  017662  043220      <MOVE!SPX!MEMX!SELB!SPO>
1028  017664              MEM     BR,INCA!SPO                     ;INCREMENT IT
 (1)          000561      MICPC=MICPC+1
 (1)  017664  062460      <MOVE!WRMEM!BR!<INCA!SPO>>
1029  017666              LDMA    IMM,T                                   ;LOAD ADDRESS OF TYPE FIELD
 (1)          000562      MICPC=MICPC+1
 (1)  017666  010151      <MOVE!LDMAR!IMM!<T&377>>
1030  017670              MEMINC  IMM,2                                   ;WRITE NAK TYPE
 (1)          000563      MICPC=MICPC+1
 (1)  017670  016402      <MOVE!WRMEM!INCMAR!IMM!<2>>
1031  017672              MEM     BR,SELB                                 ;WRITE NAK SUBYTPE FIELD SUBTYPE
 (1)          000564      MICPC=MICPC+1
 (1)  017672  062620      <MOVE!WRMEM!BR!<SELB>>
1032  017674              LDMA    IMM,NAKST                       ;ADDRESS NAKS TMTED DYNAMIC
 (1)          000565      MICPC=MICPC+1
```

```
  (1)  017674  010001          <MOVE!LDMAR!IMM!<NAKST&377>>
1033   017676                 BRWRTE  MEMX,SELB              ;WRITE IT TO BR
  (1)          000566          MICPC=MICPC+1
  (1)  017676  040620          <MOVE!WRTEBR!MEMX!<SELB>>
1034   017700                 BSHFTB                         ;SHIFT IT RIGHT
  (1)          000567          MICPC=MICPC+1
  (1)  017700  061620          <MOVE!SHFTBR!SELB!BR>
1035   017702                 MEM     BR,SELB                ;UPDATE IT
  (1)          000570          MICPC=MICPC+1
  (1)  017702  062620          <MOVE!WRMEM!BR!<SELB>>
1036   017704                 BRO     NTHRES                 ;BRANCH IF THRESHOLD EXCEEDED
  (1)          000571          MICPC=MICPC+1
  (1)  017704  116232          <JUMP!BROCON!<NTHRES-INIT&3000*4>!<NTHRES-INIT&777/2>>
1037   017706                 ALWAYS  SNAK
  (1)          000572          MICPC=MICPC+1
  (1)  017706  114660          <JUMP!ALCOND!<SNAK-INIT&3000*4>!<SNAK-INIT&777/2>>
1038   017710         RH3:    BRWRTE  DP,<DECA!SP13>         ;LOAD TYPE RECEIVED--DECREMENTING
  (1)          000573          MICPC=MICPC+1
  (1)  017710  060573          <MOVE!WRTEBR!DP!<DECA!SP13>>
1039   017712                 Z       RH1                    ;IF ALUOUT IS ALL ONES IS NUMBERED MSG
  (1)          000574          MICPC=MICPC+1
  (1)  017712  115441          <JUMP!ZCOND!<RH1-INIT&3000*4>!<RH1-INIT&777/2>>
1040   017714                 RSTATE  RCVA
  (1)          000575          MICPC=MICPC+1
  (1)  017714  000400          <MOVE!WRTEBR!IMM!<RCVA-INIT&777/2>>
  (1)          000576          MICPC=MICPC+1
  (1)  017716  063223          <MOVE!SPX!BR!SELB!SP3>
1041   017720                 BRWRTE  DP,<SELA!SP10>         ;LOAD LINE STATUS WORD IN BR
  (1)          000577          MICPC=MICPC+1
  (1)  017720  060610          <MOVE!WRTEBR!DP!<SELA!SP10>>
1047   017722                 OUTPUT  IMM,<200!ORCVCO>
  (1)          000600          MICPC=MICPC+1
  (1)  017722  002212          <MOVE!WROUT!IMM!<200!ORCVCO>>
1049   017724                 BRSHFT                         ;SHIFT RIGHT
  (1)          000601          MICPC=MICPC+1
  (1)  017724  001620          <MOVE!SHFTBR!WRTEBR!SELB>
1050   017726                 BR4     10$
  (1)          000602          MICPC=MICPC+1
  (1)  017726  107210          <JUMP!BR4CON!<10$-INIT&3000*4>!<10$-INIT&777/2>>
1051   017730                 LDMA    IMM,TYPTAB             ;ADDRESS TYPE TABLE
  (1)          000603          MICPC=MICPC+1
  (1)  017730  010162          <MOVE!LDMAR!IMM!<TYPTAB&377>>
1052   017732                 CMP     <MEMX!INCMAR>,SP13
  (1)          000604          MICPC=MICPC+1
  (1)  017732  054373          <SUBTC!MEMX!INCMAR!SP13>
1053   017734                 Z       REP
  (1)          000605          MICPC=MICPC+1
  (1)  017734  111770          <JUMP!ZCOND!<REP-INIT&3000*4>!<REP-INIT&777/2>>
1054   017736                 CMP     <MEMX!INCMAR>,SP13
  (1)          000606          MICPC=MICPC+1
  (1)  017736  054373          <SUBTC!MEMX!INCMAR!SP13>
1055   017740                 Z       NAK
  (1)          000607          MICPC=MICPC+1
  (1)  017740  115416          <JUMP!ZCOND!<NAK-INIT&3000*4>!<NAK-INIT&777/2>>
1056   017742         10$:    LDMA    IMM,TYPSTT             ;SET POINTER TO START TYPE
  (1)          000610          MICPC=MICPC+1
```

```
  (1)  017742  010164          <MOVE!LDMAR!IMM!<TYPSTT&377>>
1057   017744                  CMP     <MEMX!INCMAR>,SP13
  (1)          000611          MICPC=MICPC+1
  (1)  017744  054373          <SUBTC!MEMX!INCMAR!SP13>
1058   017746                  Z       START
  (1)          000612          MICPC=MICPC+1
  (1)  017746  115501          <JUMP!ZCOND!<START-INIT&3000*4>!<START-INIT&777/2>>
1059                                                        ;STACK TYPE
1060   017750                  CMP     <MEMX!INCMAR>,SP13
  (1)          000613          MICPC=MICPC+1
  (1)  017750  054373          <SUBTC!MEMX!INCMAR!SP13>
1061   017752                  Z       STACK
  (1)          000614          MICPC=MICPC+1
  (1)  017752  111777          <JUMP!ZCOND!<STACK-INIT&3000*4>!<STACK-INIT&777/2>>
1062   017754                  CMP     <MEMX!INCMAR>,SP13      ;ACK TYPE
  (1)          000615          MICPC=MICPC+1
  (1)  017754  054373          <SUBTC!MEMX!INCMAR!SP13>
1063   017756                  Z       ACK
  (1)          000616          MICPC=MICPC+1
  (1)  017756  115761          <JUMP!ZCOND!<ACK-INIT&3000*4>!<ACK-INIT&777/2>>
1064   017760                  ALWAYS  IDLE                    ;OTHERWISE IGNORE--MUST BE OBS MSG
  (1)          000617          MICPC=MICPC+1
  (1)  017760  100447          <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
```

```
1072
1073
1074  017762
 (1)              000620
 (1)  017762      123600
1075  017764
 (1)              000621
 (1)  017764      102047
1076  017766
 (1)              000622
 (1)  017766      022203
1077  017770
 (1)              000623
 (1)  017770      000671
1078  017772
 (1)              000624
 (1)  017772      063223
1079  017774
 (1)              000625
 (1)  017774      000621
1080  017776
 (1)              000626
 (1)  017776      104635
1081
1082  020000
 (1)              000627
 (1)  020000      123600
1084  020002
 (1)              000630
 (1)  020002      106227
1089  020004
 (1)              000631
 (1)  020004      000645
1090  020006
 (1)              000632
 (1)  020006      063223
1091  020010
 (1)              000633
 (1)  020010      022203
1092  020012
 (1)              000634
 (1)  020012      000421
1093  020014
 (1)              000635
 (1)  020014      061310
1094  020016
 (1)              000636
 (1)  020016      100447
1095  020020
1097  020020
 (1)              000637
 (1)  020020      000637
 (1)              000640
 (1)  020022      063223
1099  020024
 (1)              000641
```

```
;*****************TIME CRITICAL CODE-- CHANGE WITH GREAT CARE*********************
        .SBTTL  RCVKO1--ROUTINE TO HANDLE FIRST BYTE ODD RECEIVE
RCVKO1: SPBR    IBUS,NPR,SPO                    ;READ NPR REGISTER
        MICPC=MICPC+1
        <MOVE!SPBRX!IBUS!NPR!SPO>
        BRO     IDLE
        MICPC=MICPC+1
        <JUMP!BROCON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
        OUTPUT  IBUS,RCVDAT!OUTDA2              ;OUTPUT A CHAR
        MICPC=MICPC+1
        <MOVE!WROUT!IBUS!<RCVDAT!OUTDA2>>
        STATE   RKE1
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<RKE1-INIT&777/2>>
RCVKO2: SP      BR,SELB,SP3                     ;SET STATE
        MICPC=MICPC+1
        <MOVE!SPX!BR!SELB!SP3>
        BRWRTE  IMM,221
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<221>>
        ALWAYS  RK7
        MICPC=MICPC+1
        <JUMP!ALCOND!<RK7-INIT&3000*4>!<RK7-INIT&777/2>>
        .SBTTL  RCVKO--PROCESS ODD CHARACTER
RCVKO:  SPBR    IBUS,NPR,SPO                    ;IS AN NPR GOING
        MICPC=MICPC+1
        <MOVE!SPBRX!IBUS!NPR!SPO>
        BRO     RCVKO                           ;IF SO, REITERATE ODD AND EXIT
        MICPC=MICPC+1
        <JUMP!BROCON!<RCVKO-INIT&3000*4>!<RCVKO-INIT&777/2>>
        STATE   RCVKE
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<RCVKE-INIT&777/2>>
        SP      BR,SELB,SP3                     ;SET STATE
        MICPC=MICPC+1
        <MOVE!SPX!BR!SELB!SP3>
        OUTPUT  IBUS,RCVDAT!OUTDA2              ;OUTPUT A CHAR
        MICPC=MICPC+1
        <MOVE!WROUT!IBUS!<RCVDAT!OUTDA2>>
RK8:    BRWRTE  IMM,21                          ;SET OUT NPR (C1) AND NPR REQ
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<21>>
RK7:    OUT     BR,<AORB!ONPR>                  ;WRITE NPR REGISTER
        MICPC=MICPC+1
        <MOVE!WROUTX!BR!<AORB!ONPR>>
        ALWAYS  IDLE
        MICPC=MICPC+1
        <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
RL4:
        RSTATE  RL4
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<RL4-INIT&777/2>>
        MICPC=MICPC+1
        <MOVE!SPX!BR!SELB!SP3>
        SPBR    IBUS,NPR,SPO
        MICPC=MICPC+1
```

RCVKO--PROCESS ODD CHARACTER

```
  (1)  020024  123600                  <MOVE!SPBRX!IBUS!NPR!SPO>
1101   020026                          BRO     IDLE
  (1)                 000642           MICPC=MICPC+1
  (1)  020026  102047                  <JUMP!BROCON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
1103   020030                          STATE   RCVL
  (1)                 000643           MICPC=MICPC+1
  (1)  020030  000723                  <MOVE!WRTEBR!IMM!<RCVL-INIT&777/2>>
1104   020032                          ALWAYS  RCVK02
  (1)                 000644           MICPC=MICPC+1
  (1)  020032  104624                  <JUMP!ALCOND!<RCVK02-INIT&3000*4>!<RCVK02-INIT&777/2>>
1105                                    .SBTTL  RCVKE--HANDLE EVEN BYTES
1106   020034                  RCVKE:   BRWRTE  IBUS,NPR                ;READ NPR CONTROL REGISTER
  (1)                 000645           MICPC=MICPC+1
  (1)  020034  120600                  <MOVE!WRTEBR!IBUS!<NPR>>
1108   020036                          BR4     RK4                      ;IF  RECV NPR--BRANCH
  (1)                 000646           MICPC=MICPC+1
  (1)  020036  103376                  <JUMP!BR4CON!<RK4-INIT&3000*4>!<RK4-INIT&777/2>>
1113   020040                  RK5:     SP      IBUS,IOBA1,SPO           ;READ LOW BYTE OF BA TO SP
  (1)                 000647           MICPC=MICPC+1
  (1)  020040  023140                  <MOVE!SPX!IBUS!IOBA1!SPO>
1114   020042                          OUTPUT  DP,<INCA!OBA1>           ;WRITE INCREMENTED BA
  (1)                 000650           MICPC=MICPC+1
  (1)  020042  062066                  <MOVE!WROUT!DP!<INCA!OBA1>>
1115   020044                  RK50:    SP      BR,DECA,SP4              ;DECREMENT CHARACTER COUNT
  (1)                 000651           MICPC=MICPC+1
  (1)  020044  063164                  <MOVE!SPX!BR!DECA!SP4>
1116   020046                          C       10$                      ;NO OVERFLOW
  (1)                 000652           MICPC=MICPC+1
  (1)  020046  105255                  <JUMP!CCOND!<10$-INIT&3000*4>!<10$-INIT&777/2>>
1117   020050                          SP      BR,DECA,SP5              ;OVERFLOW - DECREMENT HIGH BYTE
  (1)                 000653           MICPC=MICPC+1
  (1)  020050  063165                  <MOVE!SPX!BR!DECA!SP5>
1118   020052                          Z       RL3                      ;BYTE COUNT ZERO
  (1)                 000654           MICPC=MICPC+1
  (1)  020052  105727                  <JUMP!ZCOND!<RL3-INIT&3000*4>!<RL3-INIT&777/2>>
1119   020054                  10$:     OUTPUT  IBUS,<RCVDAT!OUTDA1>     ;READ CHARACTER AND WRITE IT
  (1)                 000655           MICPC=MICPC+1
  (1)  020054  022202                  <MOVE!WROUT!IBUS!<RCVDAT!OUTDA1>>
1120   020056                          SP      IBUS,IOBA1,SPO           ;READ INCREMENTED BA
  (1)                 000656           MICPC=MICPC+1
  (1)  020056  023140                  <MOVE!SPX!IBUS!IOBA1!SPO>
1121   020060                          OUTPUT  DP,<INCA!OBA1>           ;WRITE INCREMENTED BA
  (1)                 000657           MICPC=MICPC+1
  (1)  020060  062066                  <MOVE!WROUT!DP!<INCA!OBA1>>
1126   020062                          C       ICBA23                   ;OVER FLOW
  (1)                 000660           MICPC=MICPC+1
  (1)  020062  115004                  <JUMP!CCOND!<ICBA23-INIT&3000*4>!<ICBA23-INIT&777/2>>
1127   020064                  RK3:     SP      BR,DECA,SP4              ;DECREMENT THE COUNT OF BYTES
  (1)                 000661           MICPC=MICPC+1
  (1)  020064  063164                  <MOVE!SPX!BR!DECA!SP4>
1128   020066                          C       RK6                      ;NO OVERFLOW
  (1)                 000662           MICPC=MICPC+1
  (1)  020066  105265                  <JUMP!CCOND!<RK6-INIT&3000*4>!<RK6-INIT&777/2>>
1129   020070                          SP      BR,DECA,SP5              ;DECREMENT HIGH BYTE OF COUNT
  (1)                 000663           MICPC=MICPC+1
  (1)  020070  063165                  <MOVE!SPX!BR!DECA!SP5>
```

```
1130  020072
 (1)          000664
 (1)  020072  105637
1132  020074
 (1)          000665
 (1)  020074  020640
1133  020076
 (1)          000666
 (1)  020076  107227
1134  020100
 (1)          000667
 (1)  020100  000627
1135  020102
 (1)          000670
 (1)  020102  104425
1141
1142  020104
 (1)          000671
 (1)  020104  123200
1144  020106
 (1)          000672
 (1)  020106  102047
1146  020110
 (1)          000673
 (1)  020110  000577
1147  020112
 (1)          000674
 (1)  020112  061270
1148  020114
 (1)          000675
 (1)  020114  104651
1149
1150
1151  020116
 (1)          000676
 (1)  020116  023200
1152  020120
 (1)          000677
 (1)  020120  062202
1153  020122
 (1)          000700
 (1)  020122  060601
1154  020124
 (1)          000701
 (1)  020124  117554
1155  020126
 (1)          000702
 (1)  020126  104661
```

```
        Z       RL4                             ;BYTE COUNT ZERO
        MICPC=MICPC+1
        <JUMP!ZCOND!<RL4-INIT&3000*4>!<RL4-INIT&777/2>>
RK6:    BRWRTE  IBUS,RCVCON                     ;READ RECEIVER CONTROL REGISTER
        MICPC=MICPC+1
        <MOVE!WRTEBR!IBUS!<RCVCON>>
        BR4     RCVKO                           ;IF ANOTHER CHARACTER--PROCESS
        MICPC=MICPC+1
        <JUMP!BR4CON!<RCVKO-INIT&3000*4>!<RCVKO-INIT&777/2>>
RK66:   STATE   RCVKO
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<RCVKO-INIT&777/2>>
        ALWAYS  REXIT
        MICPC=MICPC+1
        <JUMP!ALCOND!<REXIT-INIT&3000*4>!<REXIT-INIT&777/2>>
        .
RKE1:   SP      IBUS,NPR,SPO                    ;READ NPR REGISTER
        MICPC=MICPC+1
        <MOVE!SPX!IBUS!NPR!SPO>
        BR0     IDLE                            ;NPR STILL IN PROGRESS
        MICPC=MICPC+1
        <JUMP!BR0CON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
        BRWRTE  IMM,177                         ;MASK FOR ALL BUT C0
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<177>>
        OUT     BR,<AANDB!ONPR>                 ;TURN OFF ALL BUT C0
        MICPC=MICPC+1
        <MOVE!WROUTX!BR!<AANDB!ONPR>>
        ALWAYS  RK50
        MICPC=MICPC+1
        <JUMP!ALCOND!<RK50-INIT&3000*4>!<RK50-INIT&777/2>>
;*****************END OF TIME CRITICAL PATH*****************************************

RCVKE0: SP      IBUS,RCVDAT,SPO                 ;READ CHARACTER AND SAVE IN SPO
        MICPC=MICPC+1
        <MOVE!SPX!IBUS!RCVDAT!SPO>
        OUTPUT  BR,<SELA!OUTDA1>                ;SEND NONSENSE CHARACTER
        MICPC=MICPC+1
        <MOVE!WROUT!BR!<SELA!OUTDA1>>
        BRWRTE  BR,SELA!SP1                     ;READ STATUS BYTE
        MICPC=MICPC+1
        <MOVE!WRTEBR!BR!<SELA!SP1>>
        BR7     PASWRD                          ;MAINT MODE - SEE IF RLD MESSAGE
        MICPC=MICPC+1
        <JUMP!BR7CON!<PASWRD-INIT&3000*4>!<PASWRD-INIT&777/2>>
        ALWAYS  RK3                             ;OTHERWISE PROCESS NORMALLY
        MICPC=MICPC+1
        <JUMP!ALCOND!<RK3-INIT&3000*4>!<RK3-INIT&777/2>>
```

```
1157
1158  020130                        .SBTTL  RCVI--STORE UNNUMBERED MESSAGE TYPE
 (1)          000703
 (1)  020130  023213  RCVI:  SP      IBUS,RCVDAT,SP13        ;STORE UNNUMBERED TYPE
                             MICPC=MICPC+1
                             <MOVE!SPX!IBUS!RCVDAT!SP13>
1159  020132                STATE   RCVJ                    ;NEXT STATE IS J
 (1)          000704        MICPC=MICPC+1
 (1)  020132  000706        <MOVE!WRTEBR!IMM!<RCVJ-INIT&777/2>>
1160  020134                ALWAYS  REXIT
 (1)          000705        MICPC=MICPC+1
 (1)  020134  104425        <JUMP!ALCOND!<REXIT-INIT&3000*4>!<REXIT-INIT&777/2>>
1161                        ;
1162                        .SBTTL  RCVJ--ROUTINE TO HANDLE SUBTYPE FIELD,SELECT AND FINAL
1163  020136         RCVJ:
1168  020136                SP      IBUS,RCVDAT,SP5         ;GET CHARACTER
 (1)          000706        MICPC=MICPC+1
 (1)  020136  023205        <MOVE!SPX!IBUS!RCVDAT!SP5>
1169  020140                BRWRTE  IMM,200                 ;CONDITIONALLY SET BIT
 (1)          000707        MICPC=MICPC+1
 (1)  020140  000600        <MOVE!WRTEBR!IMM!<200>>
1170  020142                BRWRTE  BR,AANDB!SP5
 (1)          000710        MICPC=MICPC+1
 (1)  020142  060665        <MOVE!WRTEBR!BR!<AANDB!SP5>>
1171  020144                SP      BR,AORB,SP10
 (1)          000711        MICPC=MICPC+1
 (1)  020144  063310        <MOVE!SPX!BR!AORB!SP10>
1173  020146                STATE   RCVR                    ;NEXT STATE IS N
 (1)          000712        MICPC=MICPC+1
 (1)  020146  000714        <MOVE!WRTEBR!IMM!<RCVR-INIT&777/2>>
1174  020150                ALWAYS  REXIT
 (1)          000713        MICPC=MICPC+1
 (1)  020150  104425        <JUMP!ALCOND!<REXIT-INIT&3000*4>!<REXIT-INIT&777/2>>
1175                        .SBTTL  RCVR--UNNUMBERED MESSAGE RESPONSE FIELD
1176                        ;ENTERED FROM IDLE LOOP
1177                        ;
1178  020152         RCVR:  BRWRTE  IMM,3                   ;REP MESSAGE TYPE TO BR
 (1)          000714        MICPC=MICPC+1
 (1)  020152  000403        <MOVE!WRTEBR!IMM!<3>>
1179  020154                NOP     BR,SUB,SP13             ;IS TYPE ACK OR NAK
 (1)          000715        MICPC=MICPC+1
 (1)  020154  060353        <BR!SUB!SP13>
1180  020156                STATE   RCVQ                    ;NEXT STATE IS RCVQ
 (1)          000716        MICPC=MICPC+1
 (1)  020156  000721        <MOVE!WRTEBR!IMM!<RCVQ-INIT&777/2>>
1181                                                        ;***NOTE THIS INSTR DOES NOT CLOCK "C"
1182  020160                C       RCVF1                   ;IF NOT IGNORE
 (1)          000717        MICPC=MICPC+1
 (1)  020160  105140        <JUMP!CCOND!<RCVF1-INIT&3000*4>!<RCVF1-INIT&777/2>>
1183  020162                ALWAYS  RD2                     ;DO RANGE CHECKS
 (1)          000720        MICPC=MICPC+1
 (1)  020162  104471        <JUMP!ALCOND!<RD2-INIT&3000*4>!<RD2-INIT&777/2>>
1184                        .SBTTL  RCVQ--UNNUMBERED MESSAGE--NUMBER FIELD
1185                        ;ENTER  FROM IDLE
1186                        ;
1187  020164         RCVQ:  STATE   RCVF                    ;NEXT STATE IS ADDRESS
 (1)          000721        MICPC=MICPC+1
 (1)  020164  000534        <MOVE!WRTEBR!IMM!<RCVF-INIT&777/2>>
```

```
1188  020166                    ALWAYS  RCVF1
 (1)            000722          MICPC=MICPC+1
 (1)  020166    104540          <JUMP!ALCOND!<RCVF1-INIT&3000*4>!<RCVF1-INIT&777/2>>
```

```
1190
1191
1192  020170                    .SBTTL  RCVL--PROCESS CRC3
                                ;ENTERED FROM  IDLE LOOP
RCVL:   SPBR    IBUS,NPP,SPO                    ;READ NPR CONTROL
        (1)     000723          MICPC=MICPC+1
        (1)     020170  123600  <MOVE!SPBRX!IBUS!NPR!SPO>
1194  020172                    BR4     RL1                     ;RCV NPR BRANCH
        (1)     000724          MICPC=MICPC+1
        (1)     020172  107331  <JUMP!BR4CON!<RL1-INIT&3000*4>!<RL1-INIT&777/2>>
1199  020174    
RL2:    BRWRTE  IMM,176                         ;MASK TO TURN OFF CO
        (1)     000725          MICPC=MICPC+1
        (1)     020174  000576  <MOVE!WRTEBR!IMM!<176>>
1200  020176                    OUT     BR,AANDB!ONPR
        (1)     000726          MICPC=MICPC+1
        (1)     020176  061270  <MOVE!WROUTX!BR!<AANDB!ONPR>>
1201                            .
1202  020200    
RL3:    STATE   RCVM
        (1)     000727          MICPC=MICPC+1
        (1)     020200  000733  <MOVE!WRTEBR!IMM!<RCVM-INIT&777/2>>
1203  020202                    ALWAYS  RCVF1
        (1)     000730          MICPC=MICPC+1
        (1)     020202  104540  <JUMP!ALCOND!<RCVF1-INIT&3000*4>!<RCVF1-INIT&777/2>>
1204                            .
1206  020204    
RL1:    BRO     IDLE
        (1)     000731          MICPC=MICPC+1
        (1)     020204  102047  <JUMP!BROCON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
1207  020206                    ALWAYS  RL2
        (1)     000732          MICPC=MICPC+1
        (1)     020206  104725  <JUMP!ALCOND!<RL2-INIT&3000*4>!<RL2-INIT&777/2>>
1209                            .SBTTL  RCVM--PROCESS CRC4--END OF DATA MESSAGE
1210                            ;ENTERED FROM IDLE LOOP
1211                            ;IF CRC CORRECT -- QUEUE INTERUPT AND UPDATE RESPONSE
1212                            ;
1213                            ;IF CRC WRONG SEND NAK
1214  020210    
RCVM:   BRWRTE  IBUS,UBBR                       ;READ UNIBUS BR REGISTER
        (1)     000733          MICPC=MICPC+1
        (1)     020210  120620  <MOVE!WRTEBR!IBUS!<UBBR>>
1215  020212                    BRO     NXMERR                  ;NON-EXISTANT MEMORY
        (1)     000734          MICPC=MICPC+1
        (1)     020212  106364  <JUMP!BROCON!<NXMERR-INIT&3000*4>!<NXMERR-INIT&777/2>>
1216  020214                    SP      IBUS,RCVDAT,SPO                 ;READ CRC CHARACTER
        (1)     000735          MICPC=MICPC+1
        (1)     020214  023200  <MOVE!SPX!IBUS!RCVDAT!SPO>
1217  020216    
        BRWRTE  IBUS,RCVCON                     ;READ RECEIVER CONTROL REGISTER
        (1)     000736          MICPC=MICPC+1
        (1)     020216  020640  <MOVE!WRTEBR!IBUS!<RCVCON>>
1218  020220                    BRO     RCVM1                   ;IF CRC GOOD -- PROCESS
        (1)     000737          MICPC=MICPC+1
        (1)     020220  116171  <JUMP!BROCON!<RCVM1-INIT&3000*4>!<RCVM1-INIT&777/2>>
1219  020222    
        BRWRTE  BR,SELA!SP1                     ;READ STATUS BYTE
        (1)     000740          MICPC=MICPC+1
        (1)     020222  060601  <MOVE!WRTEBR!BR!<SELA!SP1>>
1220  020224                    BR7     RHX                     ;CRC ERROR IN BOOT MODE - FLUSH
        (1)     000741          MICPC=MICPC+1
        (1)     020224  107753  <JUMP!BR7CON!<RHX-INIT&3000*4>!<RHX-INIT&777/2>>
1221  020226    
        BRWRTE  IMM,302                         ;DATA ERROR SUBTYPE
        (1)     000742          MICPC=MICPC+1
```

```
  (1)  020226  000702          <MOVE!WRTEBR!IMM!<302>>
1222   020230                 LDMA    IMM,NDATS                       ;ADDRESS CUM ERROR COUNTER
  (1)          000743         MICPC=MICPC+1
  (1)  020230  010014         <MOVE!LDMAR!IMM!<NDATS&377>>
1223   020232                 ALWAYS  RH5                             ;SEND NAK
  (1)          000744         MICPC=MICPC+1
  (1)  020232  104560         <JUMP!ALCOND!<RH5-INIT&3000*4>!<RH5-INIT&777/2>>
1224
1225   020234          RCVM0: BRWRTE  IMM,10                          ;MAINT MESS ERROR
  (1)          000745         MICPC=MICPC+1
  (1)  020234  000410         <MOVE!WRTEBR!IMM!<10>>
1226   020236          RCVM2: LDMA    IMM,<<RTHRS+3>>                 ;POINT TO ERROR WORD
  (1)          000746         MICPC=MICPC+1
  (1)  020236  010177         <MOVE!LDMAR!IMM!<<RTHRS+3>&377>>
1227   020240                 ALWAYS RCEXY                           ;GIVE FATAL ERROR
  (1)          000747         MICPC=MICPC+1
  (1)  020240  114472         <JUMP!ALCOND!<RCEXY-INIT&3000*4>!<RCEXY-INIT&777/2>>
1228                           .SBTTL  EM2--PROCESS RLD MESSAGE
1229                           ;ENTERED FROM IDLE LOOP
1230                           ;IF RLD PASSWORD CHECKS TRIGGER THE BOOT ROM
1231
1232   020242          EM2:   BRWRTE  IBUS,RCVDAT                     ;READ THE CHAR
  (1)          000750         MICPC=MICPC+1
  (1)  020242  020600         <MOVE!WRTEBR!IBUS!<RCVDAT>>
1233   020244                 CMP     BR,SP13                        ;IS IT A MATCH
  (1)          000751         MICPC=MICPC+1
  (1)  020244  060373         <SUBTC!BR!SP13>
1234   020246                 Z       EM3
  (1)          000752         MICPC=MICPC+1
  (1)  020246  105761         <JUMP!ZCOND!<EM3-INIT&3000*4>!<EM3-INIT&777/2>>
1235                                                                  ;FALL INTO RHX
1236   020250          RHX:   BRWRTE  BR,AA!SP1                       ;READ STATUS BYTE SHIFTED LEFT
  (1)          000753         MICPC=MICPC+1
  (1)  020250  060521         <MOVE!WRTEBR!BR!<AA!SP1>>
1237   020252                 BR4     10$                            ;DLE RECEIVED IN NORMAL MODE
  (1)          000754         MICPC=MICPC+1
  (1)  020252  107356         <JUMP!BR4CON!<10$-INIT&3000*4>!<10$-INIT&777/2>>
1238   020254                 ALWAYS  FLUSH                          ;ALREADY IN MAINT MODE
  (1)          000755         MICPC=MICPC+1
  (1)  020254  104415         <JUMP!ALCOND!<FLUSH-INIT&3000*4>!<FLUSH-INIT&777/2>>
1239   020256          10$:   BRWRTE  IMM,165                         ;MASK TO CLEAR ALL MAINT RELATED BITS
  (1)          000756         MICPC=MICPC+1
  (1)  020256  000565         <MOVE!WRTEBR!IMM!<165>>
1240   020260                 SP      BR,AANDB,SP1                   ;CLEAR THEM
  (1)          000757         MICPC=MICPC+1
  (1)  020260  063261         <MOVE!SPX!BR!AANDB!SP1>
1241   020262                 ALWAYS  FLUSH
  (1)          000760         MICPC=MICPC+1
  (1)  020262  104415         <JUMP!ALCOND!<FLUSH-INIT&3000*4>!<FLUSH-INIT&777/2>>
1242   020264          EM3:   SP      BR,DECA,SP4
  (1)          000761         MICPC=MICPC+1
  (1)  020264  063164         <MOVE!SPX!BR!DECA!SP4>
1243   020266                 Z       EMTRIG
  (1)          000762         MICPC=MICPC+1
  (1)  020266  115666         <JUMP!ZCOND!<EMTRIG-INIT&3000*4>!<EMTRIG-INIT&777/2>>
1244   020270                 ALWAYS  IDLE
```

```
(1)          000763          MICPC=MICPC+1
(1)  020270  100447          <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
```

```
 1247
 1248  020272
  (1)                000764
  (1)  020272  010177
 1249  020274
  (1)                000765
  (1)  020274  016401
 1250  020276
  (1)                000766
  (1)  020276  002400
 1251  020300
  (1)                000767
  (1)  020300  043230
 1252  020302
  (1)                000770
  (1)  020302  114474
```

```
                .SBTTL NXMERR ---NON EXISTANT MEMORY HANDLER
NXMERR: LDMA    IMM,<<RTHRS+3>>          ;ADDRESS ERROR LINK
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<<RTHRS+3>&377>>
        MEMINC  IMM,1
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<1>>
        MEM     IMM,0                    ;NXM ERROR BIT
        MICPC=MICPC+1
        <MOVE!WRMEM!IMM!<0>>
        SP      MEMX,SELB,SP10           ;CLEAR STATUS
        MICPC=MICPC+1
        <MOVE!SPX!MEMX!SELB!SP10>
        ALWAYS  RCEXX
        MICPC=MICPC+1
        <JUMP!ALCOND!<RCEXX-INIT&3000*4>!<RCEXX-INIT&777/2>>
```

```
 1271
 1273  020304                 .
   (1)                 000771
   (1)  020304  060601
 1274  020306
   (1)                 000772
   (1)  020306  103745
 1275  020310
   (1)                 000773
   (1)  020310  000610
 1276  020312
   (1)                 000774
   (1)  020312  063301
 1277  020314
   (1)                 000775
   (1)  020314  100745
 1279  020316
   (1)                 000776
   (1)  020316  000402
 1280  020320
   (1)                 000777
   (1)  020320  114637
 1281
 1288
```

```
BOOT:   BRWRTE  BR,SELA!SP1            ;SEE IF IN MAINT. MODE
        MICPC=MICPC+1
        <MOVE!WRTEBR!BR!<SELA!SP1>>
        BR7     RA3                     ;BRANCH IF SO AND TREAT DLE LIKE NUM. MSG.
        MICPC=MICPC+1
        <JUMP!BR7CON!<RA3-INIT&3000*4>!<RA3-INIT&777/2>>
        BRWRTE  IMM,210                 ;MASK TO SET MAINT MODE AND DLE RECV'D
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<210>>
        SP      BR,AORB,SP1             ;SET THE BITS
        MICPC=MICPC+1
        <MOVE!SPX!BR!AORB!SP1>
        ALWAYS  RA3                     ;TREAT LIKE NUMBERED MESSAGE
        MICPC=MICPC+1
        <JUMP!ALCOND!<RA3-INIT&3000*4>!<RA3-INIT&777/2>>
RTHRES: BRWRTE  IMM,2
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<2>>
        ALWAYS  ERRXX
        MICPC=MICPC+1
        <JUMP!ALCOND!<ERRXX-INIT&3000*4>!<ERRXX-INIT&777/2>>
;
;
```

```
 1290           020322
 1291           000777
 1292
 1293
 1294  020322
 1295  020322
  (1)           001000
  (1)  020322   060530
 1296  020324
  (1)           001001
  (1)  020324   117774
 1297  020326
 1298  020326
  (1)           001002
  (1)  020326   060610
 1299  020330
  (1)           001003
  (1)  020330   113405
 1300  020332
  (1)           001004
  (1)  020332   100451
 1301  020334
  (1)           001005
  (1)  020334   112012
 1302  020336
  (1)           001006
  (1)  020336   112410
 1303  020340
  (1)           001007
  (1)  020340   100451
 1304  020342
 1305  020342
  (1)           001010
  (1)  020342   001620
 1306  020344
  (1)           001011
  (1)  020344   103051
 1312  020346
  (1)           001012
  (1)  020346   020660
 1313  020350
  (1)           001013
  (1)  020350   001620
 1314  020352
  (1)           001014
  (1)  020352   103051
 1315  020354
  (1)           001015
  (1)  020354   000441
  (1)           001016
  (1)  020356   063222
 1316  020360
  (1)           001017
  (1)  020360   000412
 1317  020362
  (1)           001020
```

```
        .=INIT+2000
        MICPC=777
        .SBTTL  TMTA--FIRST CHARACTER OF HEADER
        ;
TMTA:
        BRWRTE  BR,AA!SP10                      ;SHIFT LEFT
        MICPC=MICPC+1
        <MOVE!WRTEBR!BR!<AA!SP10>>
        BR7     RCVCK
        MICPC=MICPC+1
        <JUMP!BR7CON!<RCVCK-INIT&3000*4>!<RCVCK-INIT&777/2>>
TA1:
        BRWRTE  BR,SELA!SP10                    ;REREAD STATUS
        MICPC=MICPC+1
        <MOVE!WRTEBR!BR!<SELA!SP10>>
        BR7     TA2                             ;OK TO SEND?
        MICPC=MICPC+1
        <JUMP!BR7CON!<TA2-INIT&3000*4>!<TA2-INIT&777/2>>
TMTA6:  ALWAYS  I1                              ;NO
        MICPC=MICPC+1
        <JUMP!ALCOND!<I1-INIT&3000*4>!<I1-INIT&777/2>>
TA2:    BRO     NUMSYN                          ;IF UNNUMBPENDING  -- SEND IT
        MICPC=MICPC+1
        <JUMP!BROCON!<NUMSYN-INIT&3000*4>!<NUMSYN-INIT&777/2>>
        BR1     10$
        MICPC=MICPC+1
        <JUMP!BR1CON!<10$-INIT&3000*4>!<10$-INIT&777/2>>
        ALWAYS  I1
        MICPC=MICPC+1
        <JUMP!ALCOND!<I1-INIT&3000*4>!<I1-INIT&777/2>>
10$:
        BRSHFT
        MICPC=MICPC+1
        <MOVE!SHFTBR!WRTEBR!SELB>
        BR4     I1                              ;IF START MODE EXIT
        MICPC=MICPC+1
        <JUMP!BR4CON!<I1-INIT&3000*4>!<I1-INIT&777/2>>
NUMSYN: BRWRTE  IBUS,MODEM                      ;ARE WE STILL SENDING?
        MICPC=MICPC+1
        <MOVE!WRTEBR!IBUS!<MODEM>>
        BRSHFT
        MICPC=MICPC+1
        <MOVE!SHFTBR!WRTEBR!SELB>
        BR4     I1                              ;RTS SET? IF SO WE ARE--STALL
        MICPC=MICPC+1
        <JUMP!BR4CON!<I1-INIT&3000*4>!<I1-INIT&777/2>>
        TSTATE  TMTA1
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<TMTA1-INIT&777/2>>
        MICPC=MICPC+1
        <MOVE!SPX!BR!SELB!SP2>
        BRWRTE  IMM,12
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<12>>
        SP      BR,SELB,SP6                     ;STORE IN SP6
        MICPC=MICPC+1
```

TMTA--FIRST CHARACTER OF HEADER

```
  (1)  020362  063226          <MOVE!SPX!BR!SELB!SP6>
1318   020364                  ALWAYS  I1                              ;BACK TO IDLE LOOP
  (1)          001021          MICPC=MICPC+1
  (1)  020364  100451          <JUMP!ALCOND!<I1-INIT&3000*4>!<I1-INIT&777/2>>
1320   020366          TMTEXT: TSTATE  TMTB                            ;STATE TO SEND COUNT
  (1)          001022          MICPC=MICPC+1
  (1)  020366  000450          <MOVE!WRTEBR!IMM!<TMTB-INIT&777/2>>
  (1)          001023          MICPC=MICPC+1
  (1)  020370  063222          <MOVE!SPX!BR!SELB!SP2>
1321   020372                  BRWRTE  BR,AA!SP1                       ;IN PORT STATUS
  (1)          001024          MICPC=MICPC+1
  (1)  020372  060521          <MOVE!WRTEBR!BR!<AA!SP1>>
1322   020374                  BR4     10$                             ;DLE RCV'D SET-IGNORE MAINT MODE BIT
  (1)          001025          MICPC=MICPC+1
  (1)  020374  113030          <JUMP!BR4CON!<10$-INIT&3000*4>!<10$-INIT&777/2>>
1323   020376                  BRWRTE  IMM,220                         ;WRITE DLE TO BR
  (1)          001026          MICPC=MICPC+1
  (1)  020376  000620          <MOVE!WRTEBR!IMM!<220>>
1324   020400                  C       TMTA5                           ;MAINT MODE - SEND DLE
  (1)          001027          MICPC=MICPC+1
  (1)  020400  111033          <JUMP!CCOND!<TMTA5-INIT&3000*4>!<TMTA5-INIT&777/2>>
1325   020402          10$:    BRWRTE  BR,SELA!SP10                    ;NUMBERED MESSAGE?
  (1)          001030          MICPC=MICPC+1
  (1)  020402  060610          <MOVE!WRTEBR!BR!<SELA!SP10>>
1326   020404                  BRO     TMTUN                           ;NO - BRANCH
  (1)          001031          MICPC=MICPC+1
  (1)  020404  112035          <JUMP!BROCON!<TMTUN-INIT&3000*4>!<TMTUN-INIT&777/2>>
1327   020406                  BRWRTE  IMM,201                         ;OTHERWISE SEND AN SOH
  (1)          001032          MICPC=MICPC+1
  (1)  020406  000601          <MOVE!WRTEBR!IMM!<201>>
1328   020410          TMTA5:  OUTPUT  BR,<SELB!TMTDAT>                ;IN TMT SILO
  (1)          001033          MICPC=MICPC+1
  (1)  020410  062230          <MOVE!WROUT!BR!<SELB!TMTDAT>>
1329   020412                  ALWAYS  I1
  (1)          001034          MICPC=MICPC+1
  (1)  020412  100451          <JUMP!ALCOND!<I1-INIT&3000*4>!<I1-INIT&777/2>>
1330   020414          TMTUN:  TSTATE  TMTI
  (1)          001035          MICPC=MICPC+1
  (1)  020414  000602          <MOVE!WRTEBR!IMM!<TMTI-INIT&777/2>>
  (1)          001036          MICPC=MICPC+1
  (1)  020416  063222          <MOVE!SPX!BR!SELB!SP2>
1331   020420                  BRWRTE  IMM,5                           ;ENQ TO BR
  (1)          001037          MICPC=MICPC+1
  (1)  020420  000405          <MOVE!WRTEBR!IMM!<5>>
1332   020422                  ALWAYS  TMTA5
  (1)          001040          MICPC=MICPC+1
  (1)  020422  110433          <JUMP!ALCOND!<TMTA5-INIT&3000*4>!<TMTA5-INIT&777/2>>
```

TMTA--FIRST CHARACTER OF HEADER

```
1350  020424
 (1)          001041
 (1)  020424  002011
1351  020426
 (1)          001042
 (1)  020426  000626
1352  020430
 (1)          001043
 (1)  020430  062230
1353  020432
 (1)          001044
 (1)  020432  063566
1354  020434
 (1)          001045
 (1)  020434  111422
1355  020436
 (1)          001046
 (1)  020436  112041
1356  020440
 (1)          001047
 (1)  020440  100451
```

```
TMTA1:  OUTPUT  IMM,<1!OTMTCO>                  ;WRITE SOM TO TMTR CONTROL
        MICPC=MICPC+1
        <MOVE!WROUT!IMM!<1!OTMTCO>>
        BRWRTE  IMM,226                         ;SYNC CHAR
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<226>>
        OUTPUT  BR,<SELB!TMTDAT>                ;IN TMTR SILO
        MICPC=MICPC+1
        <MOVE!WROUT!BR!<SELB!TMTDAT>>
        SPBR    BR,DECA,SP6                     ;DECREMENT SYNC COUNT
        MICPC=MICPC+1
        <MOVE!SPBRX!BR!DECA!SP6>
        Z       TMTEXT                          ;DONE?
        MICPC=MICPC+1
        <JUMP!ZCOND!<TMTEXT-INIT&3000*4>!<TMTEXT-INIT&777/2>>
        BRO     TMTA1                           ;NO,OUTPUT SECOND SYNC?
        MICPC=MICPC+1
        <JUMP!BROCON!<TMTA1-INIT&3000*4>!<TMTA1-INIT&777/2>>
        ALWAYS  I1                              ;NO
        MICPC=MICPC+1
        <JUMP!ALCOND!<I1-INIT&3000*4>!<I1-INIT&777/2>>
```

```
1359                                    .SBTTL  TMTB--OUTPUT FIRST CHAR OF COUNT
1360                                    ;
```

```
1362  020442
 (1)            001050
 (1)  020442    063472  TMTB:   SPBR    BR,INCA,SP12                    ;INCREMENT MSG NO
                                MICPC=MICPC+1
                                <MOVE!SPBRX!BR!INCA!SP12>
1363  020444
 (1)            001051          OUTPUT  BR,SELB!OINDAT1
 (1)  020444    062220          MICPC=MICPC+1
                                <MOVE!WROUT!BR!<SELB!OINDAT1>>
1364  020446
 (1)            001052          LDMA    BR,SELA!SP16                    ;GETPOINTER TO NEXT TMT LINK
 (1)  020446    070216          MICPC=MICPC+1
                                <MOVE!LDMAR!BR!<SELA!SP16>>
1365  020450
 (1)            001053          MEMINC  IMM,3                           ;WRITE MSG TMTED TO FLAGS
 (1)  020450    016403          MICPC=MICPC+1
                                <MOVE!WRMEM!INCMAR!IMM!<3>>
1366  020452
 (1)            001054          MEMINC  BR,SELA!SP12                    ;PICK UP MSGNO
 (1)  020452    076612          MICPC=MICPC+1
                                <MOVE!WRMEM!INCMAR!BR!<SELA!SP12>>
1367  020454
 (1)            001055          BRWRTE  IMM,TML8                        ;GET WRAPAROUND ADDRESS
 (1)  020454    000543          MICPC=MICPC+1
                                <MOVE!WRTEBR!IMM!<TML8>>
1368  020456
 (1)            001056          CMP     BR,SP16                         ;WRAPAROUND
 (1)  020456    060376          MICPC=MICPC+1
                                <SUBTC!BR!SP16>
1369  020460
 (1)            001057          Z       TB2
 (1)  020460    111504          MICPC=MICPC+1
                                <JUMP!ZCOND!<TB2-INIT&3000*4>!<TB2-INIT&777/2>>
1370  020462                    BRWRTE  IMM,6                           ;OFFSET TO NEXT LINK
 (1)            001060          MICPC=MICPC+1
 (1)  020462    000406          <MOVE!WRTEBR!IMM!<6>>
1371  020464    TB1:    SP      BR,ADD,SP16                             ;UPDATE THE POINTER
 (1)            001061          MICPC=MICPC+1
 (1)  020464    063016          <MOVE!SPX!BR!ADD!SP16>
1372  020466                    STATE   TMTC                            ;ADDRESS TMTR STATE
 (1)            001062          MICPC=MICPC+1
 (1)  020466    000506          <MOVE!WRTEBR!IMM!<TMTC-INIT&777/2>>
1373  020470    TB0:    SP      BR,SELB,SP2                             ;UPDATE IT
 (1)            001063          MICPC=MICPC+1
 (1)  020470    063222          <MOVE!SPX!BR!SELB!SP2>
1374  020472                    OUTPUT  <MEMX!INCMAR>,SELB!IBA1         ;WRITELOWBYTEOFADDRESS
 (1)            001064          MICPC=MICPC+1
 (1)  020472    056224          <MOVE!WROUT!MEMX!INCMAR!<SELB!IBA1>>
1375  020474                    OUTPUT  <MEMX!INCMAR>,SELB!IBA2         ;WRITE HIGH BYTE OF ADDRESS
 (1)            001065          MICPC=MICPC+1
 (1)  020474    056225          <MOVE!WROUT!MEMX!INCMAR!<SELB!IBA2>>
1376  020476                    SP      MEMX,SELB,SP7   ;HIGH BYTE OF COUNT TO SP7
 (1)            001066          MICPC=MICPC+1
 (1)  020476    043227          <MOVE!SPX!MEMX!SELB!SP7>
1377                                                    ;WAIT TO MASK OFF MEM EXT. BITS
1378  020500                    SP      IBUS,NPR,SPO
 (1)            001067          MICPC=MICPC+1
 (1)  020500    123200          <MOVE!SPX!IBUS!NPR!SPO>
1379  020502                    BRWRTE  IMM,220
 (1)            001070          MICPC=MICPC+1
 (1)  020502    000620          <MOVE!WRTEBR!IMM!<220>>
1380  020504                    SP      BR,AANDB,SPO
 (1)            001071          MICPC=MICPC+1
 (1)  020504    063260          <MOVE!SPX!BR!AANDB!SPO>
1381  020506                    SP      IMM,300,SP6                     ;MASK FOR MXT
```

```
  (1)           001072
  (1)  020506   003306
1382   020510
  (1)           001073
  (1)  020510   054666
1383   020512
  (1)           001074
  (1)  020512   042230
1384   020514
  (1)           001075
  (1)  020514   001620
1385   020516
  (1)           001076
  (1)  020516   001620
1386   020520
  (1)           001077
  (1)  020520   001620
1387   020522
  (1)           001100
  (1)  020522   001620
1388   020524
  (1)           001101
  (1)  020524   061310
1389   020526
  (1)           001102
  (1)  020526   043626
1394   020530
  (1)           001103
  (1)  020530   100451
1396   020532
  (1)           001104
  (1)  020532   000726
1397   020534
  (1)           001105
  (1)  020534   110461
1398

        MICPC=MICPC+1
        <MOVE!SPX!IMM!300!SP6>
        BRWRTE  MEMX!INCMAR,AANDB!SP6           ;TURN OFF CC2
        MICPC=MICPC+1
        <MOVE!WRTEBR!MEMX!INCMAR!<AANDB!SP6>>
        OUTPUT  MEMX,SELB!TMTDAT                ;ALSO WRITE COUNT TO TMTR SILO
        MICPC=MICPC+1
        <MOVE!WROUT!MEMX!<SELB!TMTDAT>>
        BRSHFT                                  ;SHIFT BITS INTO CORRECT POSITION
        MICPC=MICPC+1
        <MOVE!SHFTBR!WRTEBR!SELB>
        BRSHFT
        MICPC=MICPC+1
        <MOVE!SHFTBR!WRTEBR!SELB>
        BRSHFT
        MICPC=MICPC+1
        <MOVE!SHFTBR!WRTEBR!SELB>
        BRSHFT
        MICPC=MICPC+1
        <MOVE!SHFTBR!WRTEBR!SELB>
        OUT     BR,AORB!ONPR
        MICPC=MICPC+1
        <MOVE!WROUTX!BR!<AORB!ONPR>>
        SPBR    MEMX,SELB,SP6           ;LOWBYTE OF COUNT TO SP6
        MICPC=MICPC+1
        <MOVE!SPBRX!MEMX!SELB!SP6>
        ALWAYS  I1
        MICPC=MICPC+1
        <JUMP!ALCOND!<I1-INIT&3000*4>!<I1-INIT&777/2>>
TB2:    BRWRTE  IMM,<<TML1-TML8>&377>           ;GO BACK TOFIRST LINK
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<<TML1-TML8>&377>>
        ALWAYS  TB1
        MICPC=MICPC+1
        <JUMP!ALCOND!<TB1-INIT&3000*4>!<TB1-INIT&777/2>>
        ;
```

```
1400
1401
1402  020536                        .SBTTL  TMTC--OUTPUT SECOND CHAR OF COUNT
                                    ;
TMTC:   BRWRTE  IMM,77                          ;MASK TO CLEAR MXT BITS
  (1)           001106              MICPC=MICPC+1
  (1)   020536  000477              <MOVE!WRTEBR!IMM!<77>>
1403  020540                        SPBR    BR,AANDB,SP7                      ;CLEAR THEM
  (1)           001107              MICPC=MICPC+1
  (1)   020540  063667              <MOVE!SPBRX!BR!AANDB!SP7>
1404  020542                        OUTPUT  DP,<SELB!TMTDAT>                  ;WRITE TO TMT SILO
  (1)           001110              MICPC=MICPC+1
  (1)   020542  062230              <MOVE!WROUT!DP!<SELB!TMTDAT>>
1409                                ;
1410                                .SBTTL  TMTD--RESPONSE FIELD-NUMBERED MESSAGE
1411  020544                        TMTD:   STATE   TMTE
  (1)           001111              MICPC=MICPC+1
  (1)   020544  000520              <MOVE!WRTEBR!IMM!<TMTE-INIT&777/2>>
1412  020546                        SP      BR,DECA,SP6                       ;ADJUSRT COUNT FOR  TWO'S  COMPLEMENT
  (1)           001112              MICPC=MICPC+1
  (1)   020546  063166              <MOVE!SPX!BR!DECA!SP6>
1413  020550                        C       TD2                               ;NO OVERFLOW
  (1)           001113              MICPC=MICPC+1
  (1)   020550  111115              <JUMP!CCOND!<TD2-INIT&3000*4>!<TD2-INIT&777/2>>
1414  020552                        SP      BR,DECA,SP7                       ;DECREMENT HIGH BYTE OF COUNT
  (1)           001114              MICPC=MICPC+1
  (1)   020552  063167              <MOVE!SPX!BR!DECA!SP7>
1415  020554                        TD2:    LDMA    IMM,ISP11                 ;RESP FIELD ADDR TO MAR
  (1)           001115              MICPC=MICPC+1
  (1)   020554  010171              <MOVE!LDMAR!IMM!<ISP11&377>>
1416  020556                        TD3:    OUTPUT  MEMX,SELB!TMTDAT          ;WRITE IT TO SILO
  (1)           001116              MICPC=MICPC+1
  (1)   020556  042230              <MOVE!WROUT!MEMX!<SELB!TMTDAT>>
1417  020560                        ALWAYS  XEXIT
  (1)           001117              MICPC=MICPC+1
  (1)   020560  110556              <JUMP!ALCOND!<XEXIT-INIT&3000*4>!<XEXIT-INIT&777/2>>
1418                                .SBTTL  TMTE--NUMBER FIELD--NUMBERED MESSAGE
1419  020562                        TMTE:
1420  020562                        SPBR    IBUS,NPR,SPO              ;READ NPR CONTROL REGISTER
  (1)           001120              MICPC=MICPC+1
  (1)   020562  123600              <MOVE!SPBRX!IBUS!NPR!SPO>
1421  020564                        BRO     I1                                ;BUSY - GET OUT
  (1)           001121              MICPC=MICPC+1
  (1)   020564  102051              <JUMP!BROCON!<I1-INIT&3000*4>!<I1-INIT&777/2>>
1422  020566                        OUTPUT  IBUS,<INDAT1!TMTDAT>              ;WRITE IT TO THE SILO
  (1)           001122              MICPC=MICPC+1
  (1)   020566  022010              <MOVE!WROUT!IBUS!<INDAT1!TMTDAT>>
1423  020570                        STATE   TMTF
  (1)           001123              MICPC=MICPC+1
  (1)   020570  000525              <MOVE!WRTEBR!IMM!<TMTF-INIT&777/2>>
1424  020572                        ALWAYS  TH3
  (1)           001124              MICPC=MICPC+1
  (1)   020572  110573              <JUMP!ALCOND!<TH3-INIT&3000*4>!<TH3-INIT&777/2>>
1425                                .SBTTL  TMTF--NUMBERED MSG ADDRESS FIELD
1426                                ;
1427  020574                        TMTF:   STATE   TF1
  (1)           001125              MICPC=MICPC+1
  (1)   020574  000532              <MOVE!WRTEBR!IMM!<TF1-INIT&777/2>>
```

```
1428  020576                TF2:    SP      BR,SELB,SP2
 (1)          001126                MICPC=MICPC+1
 (1)  020576  063222                <MOVE!SPX!BR!SELB!SP2>
1429  020600                        BRWRTE  IMM,1                   ;LOAD ADDRESS
 (1)          001127                MICPC=MICPC+1
 (1)  020600  000401                <MOVE!WRTEBR!IMM!<1>>
1431  020602                TF3:    OUTPUT  BR,<SELB!TMTDAT>
 (1)          001130                MICPC=MICPC+1
 (1)  020602  062230                <MOVE!WROUT!BR!<SELB!TMTDAT>>
1432  020604                        ALWAYS  I1
 (1)          001131                MICPC=MICPC+1
 (1)  020604  100451                <JUMP!ALCOND!<I1-INIT&3000*4>!<I1-INIT&777/2>>
1437                                .SBTTL  TF1-NUMBERED MSG HEADER EOM
1438  020606                TF1:    BRWRTE  IMM,2                           ;EOM  MASK TO BR
 (1)          001132                MICPC=MICPC+1
 (1)  020606  000402                <MOVE!WRTEBR!IMM!<2>>
1439  020610                        OUTPUT  BR,<SELB!OTMTCO>                ;UPDATE TMTR CONTROL REGISTER
 (1)          001133                MICPC=MICPC+1
 (1)  020610  062231                <MOVE!WROUT!BR!<SELB!OTMTCO>>
1440  020612                        OUTPUT  BR,<SELB!TMTDAT>                ;OUTPUT A GARBAGE CHAR
 (1)          001134                MICPC=MICPC+1
 (1)  020612  062230                <MOVE!WROUT!BR!<SELB!TMTDAT>>
1441  020614                        BRWRTE  IBUS,IIBA1                      ;READ LOW ORDER FROM INBA
 (1)          001135                MICPC=MICPC+1
 (1)  020614  020500                <MOVE!WRTEBR!IBUS!<IIBA1>>
1442  020616                        BRO     TMTF1                           ;IF ODD BYTE--BRANCH
 (1)          001136                MICPC=MICPC+1
 (1)  020616  112155                <JUMP!BROCON!<TMTF1-INIT&3000*4>!<TMTF1-INIT&777/2>>
1443  020620                        STATE   TMTH
 (1)          001137                MICPC=MICPC+1
 (1)  020620  000541                <MOVE!WRTEBR!IMM!<TMTH-INIT&777/2>>
1444  020622                        ALWAYS  XEXIT
 (1)          001140                MICPC=MICPC+1
 (1)  020622  110556                <JUMP!ALCOND!<XEXIT-INIT&3000*4>!<XEXIT-INIT&777/2>>
```

```
1446
1447
1448  020624
 (1)                001141
 (1)   020624       123600
1450  020626
 (1)                001142
 (1)   020626       113144
1452  020630
 (1)                001143
 (1)   020630       102051
1453  020632
 (1)                001144
 (1)   020632       022010
1454  020634
 (1)                001145
 (1)   020634       023100
1455  020636
 (1)                001146
 (1)   020636       062064
1456  020640
 (1)                001147
 (1)   020640       063166
1457  020642
 (1)                001150
 (1)   020642       111153
1458  020644
 (1)                001151
 (1)   020644       063167
1459  020646
 (1)                001152
 (1)   020646       101771
1460  020650
1462  020650
 (1)                001153
 (1)   020650       020620
1463  020652
 (1)                001154
 (1)   020652       113160
1465  020654
 (1)                001155
 (1)   020654       000560
1466  020656
 (1)                001156
 (1)   020656       063222
1467  020660
 (1)                001157
 (1)   020660       100451
1468  020662
1473  020662
 (1)                001160
 (1)   020662       022030
1474  020664
 (1)                001161
 (1)   020664       023100
1475  020666
```

```
;************TIME CRITICAL PATH--MODIFY WITH GREAT CARE
        .SBTTL  TMTH--ROUTINE TO OUTPUT DATA CHARACTERS
TMTH:   SPBR    IBUS,NPR,SPO                    ;READ NPR CONTROL
        MICPC=MICPC+1
        <MOVE!SPBRX!IBUS!NPR!SPO>
        BR4     5$                              ;IF RECV NPR --PROCESS
        MICPC=MICPC+1
        <JUMP!BR4CON!<5$-INIT&3000*4>!<5$-INIT&777/2>>
        BRO     I1                              ;IF NPR IN PROGRESS --BRANCH
        MICPC=MICPC+1
        <JUMP!BROCON!<I1-INIT&3000*4>!<I1-INIT&777/2>>
5$:     OUTPUT  IBUS,<INDAT1!TMTDAT>            ;WRITE THE EVEN CHAR TO TMT SILO
        MICPC=MICPC+1
        <MOVE!WROUT!IBUS!<INDAT1!TMTDAT>>
        SP      IBUS,IIBA1,SPO                  ;READ LOW BYTE OF BA TO SP
        MICPC=MICPC+1
        <MOVE!SPX!IBUS!IIBA1!SPO>
        OUTPUT  BR,<INCA!IBA1>                  ;OUTPUT INCREMENTED BA
        MICPC=MICPC+1
        <MOVE!WROUT!BR!<INCA!IBA1>>
        SP      BR,DECA,SP6                     ;DECREMENT CHARACTER  COUNT
        MICPC=MICPC+1
        <MOVE!SPX!BR!DECA!SP6>
        C       TH6                             ;NO OVERFLOW
        MICPC=MICPC+1
        <JUMP!CCOND!<TH6-INIT&3000*4>!<TH6-INIT&777/2>>
        SP      BR,DECA,SP7                     ;DECREMENT HIGH BYTE OF COUNT
        MICPC=MICPC+1
        <MOVE!SPX!BR!DECA!SP7>
        Z       HEH1                            ;BYTE  COUNT ZERO
        MICPC=MICPC+1
        <JUMP!ZCOND!<HEH1-INIT&3000*4>!<HEH1-INIT&777/2>>
TH6:
        BRWRTE  IBUS,TMTCON             ;READ TMTR CONTROL CSR
        MICPC=MICPC+1
        <MOVE!WRTEBR!IBUS!<TMTCON>>
        BR4     TH9                             ;IF MORE ROOM IN SILO--BRANCH
        MICPC=MICPC+1
        <JUMP!BR4CON!<TH9-INIT&3000*4>!<TH9-INIT&777/2>>
TMTF1:  STATE   TMTHO
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<TMTHO-INIT&777/2>>
XEXIT:  SP      BR,SELB,SP2                     ;STORE NEW TRANSMIT STATE
        MICPC=MICPC+1
        <MOVE!SPX!BR!SELB!SP2>
        ALWAYS  I1
        MICPC=MICPC+1
        <JUMP!ALCOND!<I1-INIT&3000*4>!<I1-INIT&777/2>>
TMTHO:
TH9:    OUTPUT  IBUS,<INDAT2!TMTDAT>            ;ODD CHAR TO SILO
        MICPC=MICPC+1
        <MOVE!WROUT!IBUS!<INDAT2!TMTDAT>>
        SP      IBUS,IIBA1,SPO                  ;READ LOW BYTE TO BA
        MICPC=MICPC+1
        <MOVE!SPX!IBUS!IIBA1!SPO>
        OUTPUT  BR,<INCA!IBA1>                  ;OUTPUT THE INCREMENTED BA
```

```
  (1)           001162
  (1)  020666   062064
1480   020670
  (1)           001163
  (1)  020670   111360
1481   020672
  (1)           001164
  (1)  020672   063166
1482   020674
  (1)           001165
  (1)  020674   111170
1483   020676
  (1)           001166
  (1)  020676   063167
1484   020700
  (1)           001167
  (1)  020700   101771
1485   020702
  (1)           001170
  (1)  020702   123600
1487   020704
  (1)           001171
  (1)  020704   112200
1489   020706
  (1)           001172
  (1)  020706   000541
1490   020710
  (1)           001173
  (1)  020710   063222
1491   020712
  (1)           001174
  (1)  020712   000556
1492   020714
  (1)           001175
  (1)  020714   063260
1493   020716
  (1)           001176
  (1)  020716   061070
1498   020720
  (1)           001177
  (1)  020720   100451
1499   020722
  (1)           001200
  (1)  020722   000570
1500   020724
  (1)           001201
  (1)  020724   110556
1502
```

```
        MICPC=MICPC+1
        <MOVE!WROUT!BR!<INCA!IBA1>>
        C       HOINCH
        MICPC=MICPC+1
        <JUMP!CCOND!<HOINCH-INIT&3000*4>!<HOINCH-INIT&777/2>>
TH8:    SP      BR,DECA,SP6                     ;DECREMENT CHARACTERCOUNT
        MICPC=MICPC+1
        <MOVE!SPX!BR!DECA!SP6>
        C       TH7                             ;NO OVERFLOW
        MICPC=MICPC+1
        <JUMP!CCOND!<TH7-INIT&3000*4>!<TH7-INIT&777/2>>
        SP      BR,DECA,SP7                     ;DECREMENT HIGH BYTE OF  COUNT
        MICPC=MICPC+1
        <MOVE!SPX!BR!DECA!SP7>
        Z       HEH1                            ;BYTE COUNT ZERO
        MICPC=MICPC+1
        <JUMP!ZCOND!<HEH1-INIT&3000*4>!<HEH1-INIT&777/2>>
TH7:    SPBR    IBUS,NPR,SPO                    ;READ NPR REGISTER
        MICPC=MICPC+1
        <MOVE!SPBRX!IBUS!NPR!SPO>
        BRO     TH2                             ;IF NPR BUSY WAIT TO GO
        MICPC=MICPC+1
        <JUMP!BROCON!<TH2-INIT&3000*4>!<TH2-INIT&777/2>>
        STATE   TMTH
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<TMTH-INIT&777/2>>
TH3:    SP      BR,SELB,SP2                     ;SAVE TSTATE
        MICPC=MICPC+1
        <MOVE!SPX!BR!SELB!SP2>
TH3X:   BRWRTE  IMM,156                         ,CLEAR C0 AND C1
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<156>>
        SP      BR,AANDB,SPO                    ;CLEAR THE BITS
        MICPC=MICPC+1
        <MOVE!SPX!BR!AANDB!SPO>
        OUT     BR,<INCA!ONPR>
        MICPC=MICPC+1
        <MOVE!WROUTX!BR!<INCA!ONPR>>
        ALWAYS  I1
        MICPC=MICPC+1
        <JUMP!ALCOND!<I1-INIT&3000*4>!<I1-INIT&777/2>>
TH2:    STATE   TH7
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<TH7-INIT&777/2>>
        ALWAYS  XEXIT
        MICPC=MICPC+1
        <JUMP!ALCOND!<XEXIT-INIT&3000*4>!<XEXIT-INIT&777/2>>
;***************END TIME CRITICAL PATH*******************************
```

```
1504
1505  020726                    .SBTTL  TMTI--SEND UNNUMBERED TYPE FIELD
TMTI:   LDMA    IMM,T                                   ;ADDRESS OF TYPE FIELD TO MAR
 (1)            001202          MICPC=MICPC+1
 (1)    020726  010151          <MOVE!LDMAR!IMM!<T&377>>
1506  020730                    SP      MEMX,SELB,SP6                   ;COPY IT TO SP6
 (1)            001203          MICPC=MICPC+1
 (1)    020730  043226          <MOVE!SPX!MEMX!SELB!SP6>
1507  020732                    STATE   TMTJ
 (1)            001204          MICPC=MICPC+1
 (1)    020732  000606          <MOVE!WRTEBR!IMM!<TMTJ-INIT&777/2>>
1508  020734                    ALWAYS  TD3
 (1)            001205          MICPC=MICPC+1
 (1)    020734  110516          <JUMP!ALCOND!<TD3-INIT&3000*4>!<TD3-INIT&777/2>>
1509                            ;
1510                            .SBTTL  TMTJ--SEND SUB-TYPE FIELD
1511  020736                    TMTJ:   LDMA    IMM,ST                  ;ADDRESS OF SUB-TYPE FIELD TO MAR
 (1)            001206          MICPC=MICPC+1
 (1)    020736  010152          <MOVE!LDMAR!IMM!<ST&377>>
1512  020740                    STATE   TMTK
 (1)            001207          MICPC=MICPC+1
 (1)    020740  000611          <MOVE!WRTEBR!IMM!<TMTK-INIT&777/2>>
1513  020742                    ALWAYS  TD3
 (1)            001210          MICPC=MICPC+1
 (1)    020742  110516          <JUMP!ALCOND!<TD3-INIT&3000*4>!<TD3-INIT&777/2>>
1514                            .SBTTL  TMTK--OUTPUT RESPONSE FIELD (UNNUMB MSG)
1515                            ;
1516  020744                    TMTK:   BRWRTE  IMM,3                   ;WRITE A 3 TO BR
 (1)            001211          MICPC=MICPC+1
 (1)    020744  000403          <MOVE!WRTEBR!IMM!<3>>
1517  020746                    NOP     BR,SUB,SP6                      ;IF TYPE LESS THAN 3
 (1)            001212          MICPC=MICPC+1
 (1)    020746  060346          <BR!SUB!SP6>
1518  020750                    TSTATE  TMTL
 (1)            001213          MICPC=MICPC+1
 (1)    020750  000617          <MOVE!WRTEBR!IMM!<TMTL-INIT&777/2>>
 (1)            001214          MICPC=MICPC+1
 (1)    020752  063222          <MOVE!SPX!BR!SELB!SP2>
1519  020754                    C       TMTLO
 (1)            001215          MICPC=MICPC+1
 (1)    020754  111224          <JUMP!CCOND!<TMTLO-INIT&3000*4>!<TMTLO-INIT&777/2>>
1520  020756                    ALWAYS  TD2
 (1)            001216          MICPC=MICPC+1
 (1)    020756  110515          <JUMP!ALCOND!<TD2-INIT&3000*4>!<TD2-INIT&777/2>>
1521                            ;
1522                            .SBTTL  TMTL--UNNUMB MSG NUMBER FIELD
1523  020760                    TMTL:   TSTATE  TMTM
 (1)            001217          MICPC=MICPC+1
 (1)    020760  000631          <MOVE!WRTEBR!IMM!<TMTM-INIT&777/2>>
 (1)            001220          MICPC=MICPC+1
 (1)    020762  063222          <MOVE!SPX!BR!SELB!SP2>
1524  020764                    BRWRTE  IMM,3
 (1)            001221          MICPC=MICPC+1
 (1)    020764  000403          <MOVE!WRTEBR!IMM!<3>>
1525  020766                    CMP     BR,SP6                          ;IS MESSAGE REP
 (1)            001222          MICPC=MICPC+1
 (1)    020766  060366          <SUBTC!BR!SP6>
```

```
1526  020770
 (1)            001223
 (1)  020770    111627
1527  020772
 (1)            001224
 (1)  020772    000400
1532  020774
 (1)            001225
 (1)  020774    062230
1533  020776
 (1)            001226
 (1)  020776    100451
1535
1536  021000
 (1)            001227
 (1)  021000    060612
1537  021002
 (1)            001230
 (1)  021002    110433
1538
1539
1540  021004
 (1)            001231
 (1)  021004    000633
1541  021006
 (1)            001232
 (1)  021006    110526
1542  021010
 (1)            001233
 (1)  021010    000402
1543  021012
 (1)            001234
 (1)  021012    062231
1544  021014
 (1)            001235
 (1)  021014    062230
1549  021016
 (1)            001236
 (1)  021016    060530
1550  021020
 (1)            001237
 (1)  021020    113644
1551  021022
 (1)            001240
 (1)  021022    000776
1552  021024
 (1)            001241
 (1)  021024    063270
1553  021026
 (1)            001242
 (1)  021026    000400
1554  021030
 (1)            001243
 (1)  021030    110556
1555  021032
 (1)            001244
```

```
                Z       TMTL1                           ;YES
                MICPC=MICPC+1
                <JUMP!ZCOND!<TMTL1-INIT&3000*4>!<TMTL1-INIT&777/2>>
TMTLO:  BRWRTE  IMM,0                                   ;ADDRESS CONTNAT OF ZERO
                MICPC=MICPC+1
                <MOVE!WRTEBR!IMM!<0>>
                OUTPUT  BR,<SELB!TMTDAT>                ;SEND IT OUT
                MICPC=MICPC+1
                <MOVE!WROUT!BR!<SELB!TMTDAT>>
                ALWAYS  I1                              ;BACK TO IDLE LOOP
                MICPC=MICPC+1
                <JUMP!ALCOND!<I1-INIT&3000*4>!<I1-INIT&777/2>>
;
TMTL1:  BRWRTE  BR,SELA!SP12                            ;WRITE A RESPONSE
                MICPC=MICPC+1
                <MOVE!WRTEBR!BR!<SELA!SP12>>
                ALWAYS  TMTA5
                MICPC=MICPC+1
                <JUMP!ALCOND!<TMTA5-INIT&3000*4>!<TMTA5-INIT&777/2>>
                ;
                .SBTTL  TMTM--UNNUMB MSG--STATION ADDRESS
TMTM:   STATE   TNEOM
                MICPC=MICPC+1
                <MOVE!WRTEBR!IMM!<TNEOM-INIT&777/2>>
                ALWAYS  TF2
                MICPC=MICPC+1
                <JUMP!ALCOND!<TF2-INIT&3000*4>!<TF2-INIT&777/2>>
TNEOM:  BRWRTE  IMM,2                                   ;END OF MESSAGE TO BR
                MICPC=MICPC+1
                <MOVE!WRTEBR!IMM!<2>>
                OUTPUT  BR,<SELB!OTMTCO>
                MICPC=MICPC+1
                <MOVE!WROUT!BR!<SELB!OTMTCO>>
                OUTPUT  BR,<SELB!TMTDAT>                ;OUTPUT A GARBAGE CHARACTER
                MICPC=MICPC+1
                <MOVE!WROUT!BR!<SELB!TMTDAT>>
                BRWRTE  BR,AA!SP10                      ;SHIFT STATUS LEFT
                MICPC=MICPC+1
                <MOVE!WRTEBR!BR!<AA!SP10>>
                BR7     TNEOM2                          ;IF HDX SET---BRANCH TO CLEAR OK TO SEND
                MICPC=MICPC+1
                <JUMP!BR7CON!<TNEOM2-INIT&3000*4>!<TNEOM2-INIT&777/2>>
                BRWRTE  IMM,376                         ;MASK TO TURN OFF UNNUMB PENDDING
                MICPC=MICPC+1
                <MOVE!WRTEBR!IMM!<376>>
TNEOM1: SP      BR,AANDB,SP10                           ;MASK TO LINE STATUS WORD
                MICPC=MICPC+1
                <MOVE!SPX!BR!AANDB!SP10>
TEOM2:  STATE   TMTA
                MICPC=MICPC+1
                <MOVE!WRTEBR!IMM!<TMTA-INIT&777/2>>
                ALWAYS  XEXIT
                MICPC=MICPC+1
                <JUMP!ALCOND!<XEXIT-INIT&3000*4>!<XEXIT-INIT&777/2>>
TNEOM2: BRWRTE  IMM,176                                 ;CLEAR OK TO SEND AND UNNUMB PENPENDING
                MICPC=MICPC+1
```

```
   (1)  021032  000576          <MOVE!WRTEBR!IMM!<176>>
1556    021034                ALWAYS  TNEOM1
   (1)          001245        MICPC=MICPC+1
   (1)  021034  110641        <JUMP!ALCOND!<TNEOM1-INIT&3000*4>!<TNEOM1-INIT&777/2>>
```

```
                        .SBTTL  TIMSRV--TIMEOUT ROUTINE--SENDS  REP
                        ;
                        .ENABLE LSB
TIMSRV: BRWRTE  IMM,177                                 ;MASK OFF  BR REQ
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<177>>
                                                                ;RESET TIMER---SLICK MOVE
                                                                ;SINCE TIMER IS RESET BY WRITING
                                                                ;A 1 AND THE EXPIRATION  LOOKS
                                                                ;LIKE 1--VOILA
        OUT     BR,<AANDB!OBR>                          ;AND THE BIT ON
        MICPC=MICPC+1
        <MOVE!WROUTX!BR!<AANDB!OBR>>
        SP      BR,INCA,SP14                            ;INCREMENT THE COUNT
        MICPC=MICPC+1
        <MOVE!SPX!BR!INCA!SP14>
                                                        ;NOTE: STATE OF "C" BIT
                                                        ;MUST BE PRESERVED FROM THIS
                                                        ;POINT UNTIL IT IS TESTED
TIMEO:
        BRWRTE  BR,SELA!SP1                             ;READ STATUS BYTE
        MICPC=MICPC+1
        <MOVE!WRTEBR!BR!<SELA!SP1>>
        BRO     IDLE
        MICPC=MICPC+1
        <JUMP!BROCON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
        BR7     IDLE                                    ;IF IN MAINT. MODE DISABLE TIMER
        MICPC=MICPC+1
        <JUMP!BR7CON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
        BRWRTE  BR,SELA!SP10                            ;READ LINE STATUS REGISTER
        MICPC=MICPC+1
        <MOVE!WRTEBR!BR!<SELA!SP10>>
        BSHFTB                                          ;ROTATE IT RIGHT ONE BIT
        MICPC=MICPC+1
        <MOVE!SHFTBR!SELB!BR>
        BR4     BS1                                     ;START MODE - RESEND A START
        MICPC=MICPC+1
        <JUMP!BR4CON!<BS1-INIT&3000*4>!<BS1-INIT&777/2>>
                                                        ;TEST STATE OF C BIT FROM INC SP14
        C       20$                                     ;IF CARRY SET COUNT HAS EXPIRED
        MICPC=MICPC+1
        <JUMP!CCOND!<20$-INIT&3000*4>!<20$-INIT&777/2>>
        BRO     TABUPD                                  ;NUMBERED MESSAGE IN PROGRESS
        MICPC=MICPC+1
        <JUMP!BROCON!<TABUPD-INIT&3000*4>!<TABUPD-INIT&777/2>>
        BR7     TABUPD                                  ;UNNUMBMSGIN PROGRESS
        MICPC=MICPC+1
        <JUMP!BR7CON!<TABUPD-INIT&3000*4>!<TABUPD-INIT&777/2>>
        ALWAYS  SNDACK                                  ;ELSE TRY TO SEND AN ACK
        MICPC=MICPC+1
        <JUMP!ALCOND!<SNDACK-INIT&3000*4>!<SNDACK-INIT&777/2>>
TIME1:
20$:
        BRWRTE  IMM,205                                 ;SET OK TO SEND,UNNUMB PEND,AND
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<205>>
```

```
1586
1587  021072
 (1)              001264
 (1)  021072      063310
1588  021074
 (1)              001265
 (1)  021074      003374
1589  021076
 (1)              001266
 (1)  021076      060612
1590  021100
 (1)              001267
 (1)  021100      060377
1591  021102
 (1)              001270
 (1)  021102      111723
1592  021104
 (1)              001271
 (1)  021104      010015
1593  021106
 (1)              001272
 (1)  021106      043220
1594  021110
 (1)              001273
 (1)  021110      062460
1595  021112
 (1)              001274
 (1)  021112      010003
1596  021114
 (1)              001275
 (1)  021114      040620
1597  021116
 (1)              001276
 (1)  021116      061620
1598  021120
 (1)              001277
 (1)  021120      062620
1599  021122
 (1)              001300
 (1)  021122      106376
1600  021124
 (1)              001301
 (1)  021124      010151
1601  021126
 (1)              001302
 (1)  021126      016403
1602  021130
 (1)              001303
 (1)  021130      110725
1603
1604
```

```
                                                    ;LINE GONE IDLE
        SP      BR,AORB,SP10
        MICPC=MICPC+1
        <MOVE!SPX!BR!AORB!SP10>
        SP      IMM,374,SP14                        ;RESET THE TIMER TICK COUNT
        MICPC=MICPC+1
        <MOVE!SPX!IMM!374!SP14>
        BRWRTE  BR,SELA!SP12                        ;GET LAST NUMBER SENT
        MICPC=MICPC+1
        <MOVE!WRTEBR!BR!<SELA!SP12>>
        CMP     BR,SP17                             ;COMPARE TO LAST ACKED
        MICPC=MICPC+1
        <SUBTC!BR!SP17>
        Z       SNDACK                      ;IF EQ --SEND ACK
        MICPC=MICPC+1
        <JUMP!ZCOND!<SNDACK-INIT&3000*4>!<SNDACK-INIT&777/2>>
TIME2:  LDMA    IMM,REPCS                           ;CUMULATIVE REPS SENT
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<REPCS&377>>
        SP      MEMX,SELB,SP0                       ;COPY IT TO SP0
        MICPC=MICPC+1
        <MOVE!SPX!MEMX!SELB!SP0>
        MEM     BR,INCA!SP0                         ;INCREMENT IT
        MICPC=MICPC+1
        <MOVE!WRMEM!BR!<INCA!SP0>>
        LDMA    IMM,REPST                           ;ADDRESS DYNAMIC REP COUNTER
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<REPST&377>>
        BRWRTE  MEMX,SELB                           ;COPY IT TO THE BR
        MICPC=MICPC+1
        <MOVE!WRTEBR!MEMX!<SELB>>
        BSHFTB
        MICPC=MICPC+1
        <MOVE!SHFTBR!SELB!BR>
        MEM     BR,SELB
        MICPC=MICPC+1
        <MOVE!WRMEM!BR!<SELB>>
        BRO     RTHRES
        MICPC=MICPC+1
        <JUMP!BROCON!<RTHRES-INIT&3000*4>!<RTHRES-INIT&777/2>>
        LDMA    IMM,T                               ;LOAD ADDRESS OF TYPE FIELD IN UNNUMB SK
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<T&377>>
        MEMINC  IMM,3                               ;LOAD REP TYPE
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<3>>
        ALWAYS  SA1
        MICPC=MICPC+1
        <JUMP!ALCOND!<SA1-INIT&3000*4>!<SA1-INIT&777/2>>
        .DSABLE LSB
        ;
```

```
1606  021132
 (1)            001304
 (1)  021132    120620  TEOM:   BRWRTE  IBUS,UBBR
                                MICPC=MICPC+1
                                <MOVE!WRTEBR!IBUS!<UBBR>>
1607  021134
 (1)            001305         BRO     NXMERR                          ;NON-EXISTANT MEMORY
 (1)  021134    106364         MICPC=MICPC+1
                                <JUMP!BROCON!<NXMERR-INIT&3000*4>!<NXMERR-INIT&777/2>>
1608  021136                   BRWRTE  IMM,2                           ;EOM TO BR
 (1)            001306         MICPC=MICPC+1
 (1)  021136    000402         <MOVE!WRTEBR!IMM!<2>>
1609  021140                   OUTPUT  BR,<SELB!OTMTCO>                ;WRITE TMTR CONTROL
 (1)            001307         MICPC=MICPC+1
 (1)  021140    062231         <MOVE!WROUT!BR!<SELB!OTMTCO>>
1610  021142                   OUTPUT  BR,<SELB!TMTDAT>                ;WRITE GARBAGE DATA
 (1)            001310         MICPC=MICPC+1
 (1)  021142    062230         <MOVE!WROUT!BR!<SELB!TMTDAT>>
1611  021144                   BRWRTE  BR,SELA!SP1                     ;CKECK FOR BOOT MODE
 (1)            001311         MICPC=MICPC+1
 (1)  021144    060601         <MOVE!WRTEBR!BR!<SELA!SP1>>
1612  021146                   BR7     BTEOM                           ;---IF SET IS MAINT MSG
 (1)            001312         MICPC=MICPC+1
 (1)  021146    113750         <JUMP!BR7CON!<BTEOM-INIT&3000*4>!<BTEOM-INIT&777/2>>
1613  021150    TEOM1:  LDMA    BR,SELA!SP16                            ;ADDRESS LAST TMT LINK
 (1)            001313         MICPC=MICPC+1
 (1)  021150    070216         <MOVE!LDMAR!BR!<SELA!SP16>>
1614  021152                   BRWRTE  MEMX,SELB
 (1)            001314         MICPC=MICPC+1
 (1)  021152    040620         <MOVE!WRTEBR!MEMX!<SELB>>
1615  021154                   BRO     TEOM2
 (1)            001315         MICPC=MICPC+1
 (1)  021154    112242         <JUMP!BROCON!<TEOM2-INIT&3000*4>!<TEOM2-INIT&777/2>>
1616  021156    TEOM3:  BRWRTE  IMM,375                         ;TURN OFF MESSAGE PENDING
 (1)            001316         MICPC=MICPC+1
 (1)  021156    000775         <MOVE!WRTEBR!IMM!<375>>
1617  021160                   SPBR    BR,AANDB,SP10           ;
 (1)            001317         MICPC=MICPC+1
 (1)  021160    063670         <MOVE!SPBRX!BR!AANDB!SP10>
1618  021162                   BRO     TEOM2                           ;IF UNNUMB PENDING--GO AWAY
 (1)            001320         MICPC=MICPC+1
 (1)  021162    112242         <JUMP!BROCON!<TEOM2-INIT&3000*4>!<TEOM2-INIT&777/2>>
1619  021164                   TSTATE  TMTA
 (1)            001321         MICPC=MICPC+1
 (1)  021164    000400         <MOVE!WRTEBR!IMM!<TMTA-INIT&777/2>>
 (1)            001322         MICPC=MICPC+1
 (1)  021166    063222         <MOVE!SPX!BR!SELB!SP2>
1620                     .SBTTL  SNDACK--ROUTINE TO SEND AN ACK
1621  021170    SNDACK: LDMA    IMM,T
 (1)            001323         MICPC=MICPC+1
 (1)  021170    010151         <MOVE!LDMAR!IMM!<T&377>>
1622  021172                   MEMINC  IMM,1
 (1)            001324         MICPC=MICPC+1
 (1)  021172    016401         <MOVE!WRMEM!INCMAR!IMM!<1>>
1623  021174    SA1:    BRWRTE  IMM,5
 (1)            001325         MICPC=MICPC+1
 (1)  021174    000405         <MOVE!WRTEBR!IMM!<5>>
1624  021176    SA2:    MEMINC  IMM,300
 (1)            001326         MICPC=MICPC+1
```

```
                                SNDACK--ROUTINE TO SEND AN ACK

                                        <MOVE!WRMEM!INCMAR!IMM!<300>>
  (1)  021176  016700           SA3:    SP      BR,AORB,SP10
 1625  021200                          MICPC=MICPC+1
  (1)          001327                   <MOVE!SPX!BR!AORB!SP10>
  (1)  021200  063310                   ALWAYS  IDLE
 1626  021202                          MICPC=MICPC+1
  (1)          001330                   <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
  (1)  021202  100447           
 1627                                   .
 1628  021204                   FUDGE:  BRWRTE  IBUS,NPR                ;READ NPR CONTROL
  (1)          001331                  MICPC=MICPC+1
  (1)  021204  120600                   <MOVE!WRTEBR!IBUS!<NPR>>
 1629  021206                          BRO     IDLE                    ;IF NPR GOING---LEAVE
  (1)          001332                  MICPC=MICPC+1
  (1)  021206  102047                   <JUMP!BROCON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
 1630  021210                          BRWRTE  BR!LDMAR,SELA!SP4       ;LOAD THE MAR
  (1)          001333                  MICPC=MICPC+1
  (1)  021210  070604                   <MOVE!WRTEBR!BR!LDMAR!<SELA!SP4>>
 1631  021212                          BR7     BS2                     ;IF SET - READ BACK ALL 200
  (1)          001334                  MICPC=MICPC+1
  (1)  021212  103571                   <JUMP!BR7CON!<BS2-INIT&3000*4>!<BS2-INIT&777/2>>
 1632  021214                          MEMINC  IBUS,INDAT1             ;OTHERWISE RESTORE TWO BYTES
  (1)          001335                  MICPC=MICPC+1
  (1)  021214  036400                   <MOVE!WRMEM!INCMAR!IBUS!<INDAT1>>
 1633  021216                          MEMINC  IBUS,INDAT2             ;..
  (1)          001336                  MICPC=MICPC+1
  (1)  021216  036420                   <MOVE!WRMEM!INCMAR!IBUS!<INDAT2>>
 1634  021220                          BRWRTE  IMM,2                   ;UPDATE---UNIBUS ADDRESS
  (1)          001337                  MICPC=MICPC+1
  (1)  021220  000402                   <MOVE!WRTEBR!IMM!<2>>
 1635  021222                          SP      BR,ADD,SP4              ;UPDATE NPR COUNTER
  (1)          001340                  MICPC=MICPC+1
  (1)  021222  063004                   <MOVE!SPX!BR!ADD!SP4>
 1636  021224                          SP      IBUS,IIBA1,SPO          ;UPDATE ADDRESS LOW
  (1)          001341                  MICPC=MICPC+1
  (1)  021224  023100                   <MOVE!SPX!IBUS!IIBA1!SPO>
 1637  021226                          OUTPUT  BR,ADD!IBA1
  (1)          001342                  MICPC=MICPC+1
  (1)  021226  062004                   <MOVE!WROUT!BR!<ADD!IBA1>>
 1638  021230                          SP      IBUS,IIBA2,SPO          ;READ HIGH ADDRESS
  (1)          001343                  MICPC=MICPC+1
  (1)  021230  023120                   <MOVE!SPX!IBUS!IIBA2!SPO>
 1639  021232                          OUTPUT  BR,AC!IBA2              ;UPDATE HIGH
  (1)          001344                  MICPC=MICPC+1
  (1)  021232  062105                   <MOVE!WROUT!BR!<AC!IBA2>>
 1640  021234                          SP      IBUS,NPR,SPO            ;READ NPR REGISTER
  (1)          001345                  MICPC=MICPC+1
  (1)  021234  123200                   <MOVE!SPX!IBUS!NPR!SPO>
 1641  021236                          C       RESEXT                  ;IF CARRY---UPDATE MXT
  (1)          001346                  MICPC=MICPC+1
  (1)  021236  115151                   <JUMP!CCOND!<RESEXT-INIT&3000*4>!<RESEXT-INIT&777/2>>
 1642  021240                          ALWAYS  TH3X                    ;GO DO ANOTHER NPR
  (1)          001347                  MICPC=MICPC+1
  (1)  021240  110574                   <JUMP!ALCOND!<TH3X-INIT&3000*4>!<TH3X-INIT&777/2>>
```

```
                        SNDACK--ROUTINE TO SEND AN ACK

1644  021242
 (1)            001350
 (1)  021242    000774  BTEOM:  BRWRTE  IMM,374                 ;MASK FOR CLEAR MSG PENDING
                                MICPC=MICPC+1
                                <MOVE!WRTEBR!IMM!<374>>
1645  021244                    SP      BR,AANDB,SP10           ;TURN THEM OFF IN LINE STATUS WORD
 (1)            001351         MICPC=MICPC+1
 (1)  021244    063270         <MOVE!SPX!BR!AANDB!SP10>
1646  021246                    SP      BR,SELB,SP13            ;STORE UNRECOGNIZABLE VALUE INTO SP13
 (1)            001352         MICPC=MICPC+1
 (1)  021246    063233         <MOVE!SPX!BR!SELB!SP13>
1647                                                            ;SO "RH3" WILL EXIT BACK TO IDLE LOOP
1648  021250                    LDMA    IMM,STC                 ;ADDRESS START OF TMT  CHAIN
 (1)            001353         MICPC=MICPC+1
 (1)  021250    010067         <MOVE!LDMAR!IMM!<STC&377>>
1649  021252                    SP      MEMX,SELB,SPO           ;COPY LINK ADDRESS
 (1)            001354         MICPC=MICPC+1
 (1)  021252    043220         <MOVE!SPX!MEMX!SELB!SPO>
1654  021254                    TSTATE  TMTA                    ;CHANGE XMIT STATE TO LINE IS IDLE
 (1)            001355         MICPC=MICPC+1
 (1)  021254    000400         <MOVE!WRTEBR!IMM!<TMTA-INIT&777/2>>
 (1)            001356         MICPC=MICPC+1
 (1)  021256    063222         <MOVE!SPX!BR!SELB!SP2>
1655  021260                    ALWAYS  TDON2                   ;POST A DONE
 (1)            001357         MICPC=MICPC+1
 (1)  021260    114514         <JUMP!ALCOND!<TDON2-INIT&3000*4>!<TDON2-INIT&777/2>>
1656                            ;INCREMENT MXT BITS
1657  021262
1659  021262            HOINCH:
 (1)            001360         SP      IBUS,IIBA2,SPO          ;GET HIGH BYTE
 (1)  021262    023120         MICPC=MICPC+1
1660  021264                    <MOVE!SPX!IBUS!IIBA2!SPO>
 (1)            001361         OUTPUT  BR,<INCA!IBA2>          ;INCREMENT AND OUTPUT
 (1)  021264    062065         MICPC=MICPC+1
1661  021266                    <MOVE!WROUT!BR!<INCA!IBA2>>
 (1)            001362         C       IO$                     ;OVERFLOW-GO INCREMENT MXT BITS
 (1)  021266    111364         MICPC=MICPC+1
1662  021270                    <JUMP!CCOND!<IO$-INIT&3000*4>!<IO$-INIT&777/2>>
 (1)            001363         ALWAYS  TH8                     ;OTHERWISE RETURN
 (1)  021270    110564         MICPC=MICPC+1
1663  021272                    <JUMP!ALCOND!<TH8-INIT&3000*4>!<TH8-INIT&777/2>>
1665  021272            IO$:
 (1)            001364         SP      IBUS,NPR,SPO            ;READ NPR REG IWTH CURRENT MXT BITS
 (1)  021272    123200         MICPC=MICPC+1
1666  021274                    <MOVE!SPX!IBUS!NPR!SPO>
 (1)            001365         BRWRTE  IMM,4                   ;WRITE BIT TO ADD
 (1)  021274    000404         MICPC=MICPC+1
1667  021276                    <MOVE!WRTEBR!IMM!<4>>
 (1)            001366         OUT     BR,<ADD!ONPR>           ;TURN ON PROPER MXT BITS
 (1)  021276    061010         MICPC=MICPC+1
1668  021300                    <MOVE!WROUTX!BR!<ADD!ONPR>>
 (1)            001367         ALWAYS  TH8
 (1)  021300    110564         MICPC=MICPC+1
1669                            <JUMP!ALCOND!<TH8-INIT&3000*4>!<TH8-INIT&777/2>>
1670  021302                    .SBTTL  REP HANDLER
                        REP:    LDMA    IMM,REPCR               ;LOAD MAR ADDRESS WITH POINTER TO REPS RECC
 (1)            001370         MICPC=MICPC+1
 (1)  021302    010016         <MOVE!LDMAR!IMM!<REPCR&377>>
1671  021304                    SP      MEMX,SELB,SPO           ;READ NUMBER OF REPS RECD
```

```
(1)             001371
(1)  021304     043220
1672 021306
(1)             001372
(1)  021306     062460
1673 021310
(1)             001373
(1)  021310     010151
1674 021312
(1)             001374
(1)  021312     016402
1675 021314
(1)             001375
(1)  021314     016703
1676 021316
(1)             001376
(1)  021316     114660
1677
1678 021320
(1)             001377
(1)  021320     117001
1679 021322
(1)             001400
(1)  021322     100447
1680 021324
(1)             001401
(1)  021324     000727
1681 021326
(1)             001402
(1)  021326     063270
1682 021330
(1)             001403
(1)  021330     110723
```

```
        MICPC=MICPC+1
        <MOVE!SPX!MEMX!SELB!SPO>
        MEM     DP,<INCA!SPO>           ;INCREMNT REPS RECD
        MICPC=MICPC+1
        <MOVE!WRMEM!DP!<INCA!SPO>>
        LDMA    IMM,T                   ;LOAD ADDRESS OF TYPE FIELD
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<T&377>>
        MEMINC  IMM,2                   ; LOAD NAK TYPE
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<2>>
        MEMINC  IMM,303                 ;LOAD REP RESPONSE SUB-TYPE
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<303>>
        ALWAYS  SNAK                    ;SEND AN UNNUMB MSG
        MICPC=MICPC+1
        <JUMP!ALCOND!<SNAK-INIT&3000*4>!<SNAK-INIT&777/2>>
        .
STACK:  BR4     5$                      ;IF START MODE ACCEPT STACK
        MICPC=MICPC+1
        <JUMP!BR4CON!<5$-INIT&3000*4>!<5$-INIT&777/2>>
        ALWAYS  IDLE                    ;IGNORE STACK
        MICPC=MICPC+1
        <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
5$:     BRWRTE  IMM,327                 ;MASK TO CLEAR START
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<327>>
        SP      BR,AANDB,SP10           ;CLEAR START MODE
        MICPC=MICPC+1
        <MOVE!SPX!BR!AANDB!SP10>
        ALWAYS  SNDACK          ;RESET TIMER AND IDLE
        MICPC=MICPC+1
        <JUMP!ALCOND!<SNDACK-INIT&3000*4>!<SNDACK-INIT&777/2>>
```

```
1684  021332
1686  021332
 (1)            001404
 (1)  021332    023160
1687  021334
 (1)            001405
 (1)  021334    062067
1688  021336
 (1)            001406
 (1)  021336    115010
1689  021340
 (1)            001407
 (1)  021340    104661
1690  021342
1692  021342
 (1)            001410
 (1)  021342    123220
1693  021344
 (1)            001411
 (1)  021344    000515
1694  021346
 (1)            001412
 (1)  021346    063260
1695  021350
 (1)            001413
 (1)  021350    000404
1696  021352
 (1)            001414
 (1)  021352    061011
1697  021354
 (1)            001415
 (1)  021354    104661
1702  021356
 (1)            001416
 (1)  021356    010011
1703  021360
 (1)            001417
 (1)  021360    043220
1704  021362
 (1)            001420
 (1)  021362    042460
1705  021364
 (1)            001421
 (1)  021364    060617
1706  021366
 (1)            001422
 (1)  021366    063232
1707  021370
 (1)            001423
 (1)  021370    000406
1708
1709  021372
 (1)            001424
 (1)  021372    010067
1710  021374
 (1)            001425
```

```
ICBA23:
        SP      IBUS,IOBA2,SPO                  ;GET HIGH BYTE
        MICPC=MICPC+1
        <MOVE!SPX!IBUS!IOBA2!SPO>
        OUTPUT  DP,<INCA!OBA2>                  ;INCREMENT IT
        MICPC=MICPC+1
        <MOVE!WROUT!DP!<INCA!OBA2>>
        C       10$                             ;OVERFLOW - INCREMENT MXT BITS
        MICPC=MICPC+1
        <JUMP!CCOND!<10$-INIT&3000*4>!<10$-INIT&777/2>>
        ALWAYS  RK3
        MICPC=MICPC+1
        <JUMP!ALCOND!<RK3-INIT&3000*4>!<RK3-INIT&777/2>>
10$:
        SP      IBUS,UBBR,SPO
        MICPC=MICPC+1
        <MOVE!SPX!IBUS!UBBR!SPO>
        BRWRTE  IMM,115
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<115>>
        SP      BR,AANDB,SPO
        MICPC=MICPC+1
        <MOVE!SPX!BR!AANDB!SPO>
        BRWRTE  IMM,4
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<4>>
        OUT     BR,<ADD!OBR>
        MICPC=MICPC+1
        <MOVE!WROUTX!BR!<ADD!OBR>>
        ALWAYS  RK3
        MICPC=MICPC+1
        <JUMP!ALCOND!<RK3-INIT&3000*4>!<RK3-INIT&777/2>>
NAK:    LDMA    IMM,NDATR                       ;CUMMULATIVE NAK COUNTER
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<NDATR&377>>
        SP      MEMX,SELB,SPO                   ;READ IT
        MICPC=MICPC+1
        <MOVE!SPX!MEMX!SELB!SPO>
        MEM     MEMX,INCA!SPO                   ;INCREMENT THE COUNTER
        MICPC=MICPC+1
        <MOVE!WRMEM!MEMX!<INCA!SPO>>
        BRWRTE  BR,SELA!SP17                            ;GETLASTMESSAGE ACKED
        MICPC=MICPC+1
        <MOVE!WRTEBR!BR!<SELA!SP17>>
        SP      BR,SELB,SP12                            ;COPY TO CURRENT NUMBER
        MICPC=MICPC+1
        <MOVE!SPX!BR!SELB!SP12>
        BRWRTE  IMM,6                           ;WRITE NUMBERED MSG PENDING
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<6>>
                                                ;AND LINE HAS GONE IDLE
NAK1:   LDMA    IMM,STC                         ;ADDRESS START OF XMIT CHAIN
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<STC&377>>
        SP      MEMX!LDMAR,SELB,SP16                    ;COPY START OF CHAIN TO LAST XMIT
        MICPC=MICPC+1
```

```
  (1)  021374  053236
1711   021376
  (1)          001426
  (1)  021376  063310
1712   021400
  (1)          001427
  (1)  021400  003374
1713   021402
  (1)          001430
  (1)  021402  113713
1714   021404
  (1)          001431
  (1)  021404  040620
1715   021406
  (1)          001432
  (1)  021406  102051
1716   021410
  (1)          001433
  (1)  021410  110716
1717   021412
  (1)          001434
  (1)  021412  000600
1718   021414
  (1)          001435
  (1)  021414  123000
1719   021416
  (1)          001436
  (1)  021416  061300
1720   021420
  (1)  021420
  (2)          001437
  (2)  021420  002502
1721   021422
  (1)          001440
  (1)  021422  100447
1722
```

```
        <MOVE!SPX!MEMX!LDMAR!SELB!SP16>
        SP      BR,AORB,SP10            ;SET IT IN LINE STATUS WORD
        MICPC=MICPC+1
        <MOVE!SPX!BR!AORB!SP10>
        SP      IMM,374,SP14            ;RESET TIMER COUNT
        MICPC=MICPC+1
        <MOVE!SPX!IMM!374!SP14>
        BR7     TEOM1
        MICPC=MICPC+1
        <JUMP!BR7CON!<TEOM1-INIT&3000*4>!<TEOM1-INIT&777/2>>
        BRWRTE  MEMX,SELB
        MICPC=MICPC+1
        <MOVE!WRTEBR!MEMX!<SELB>>
        BRO     I1
        MICPC=MICPC+1
        <JUMP!BROCON!<I1-INIT&3000*4>!<I1-INIT&777/2>>
        ALWAYS  TEOM3
        MICPC=MICPC+1
        <JUMP!ALCOND!<TEOM3-INIT&3000*4>!<TEOM3-INIT&777/2>>
INWAT1: BRWRTE  IMM,200                 ;MASK FOR RDYI
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<200>>
        SP      IBUS,INCON,SPO          ;READ INPUT CONTROL CSR
        MICPC=MICPC+1
        <MOVE!SPX!IBUS!INCON!SPO>
        OUT     BR,AORB!OINCON          ;SET RDYI
        MICPC=MICPC+1
        <MOVE!WROUTX!BR!<AORB!OINCON>>
        PSTATE  IEIWAT                  ;CHANGE STATE TO WAIT FOR IEI OR RQI
        MEM     IMM,<<IEIWAT-INIT&777/2>>
        MICPC=MICPC+1
        <MOVE!WRMEM!IMM!<<IEIWAT-INIT&777/2>>>
        ALWAYS  IDLE
        MICPC=MICPC+1
        <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
        ;
```

```
1732
1733  021424
 (1)                001441
 (1)  021424  070075
1734  021426
 (1)                001442
 (1)  021426  060601
1735  021430
 (1)                001443
 (1)  021430  116455
1736  021432
 (1)                001444
 (1)  021432  014477
1737  021434
 (1)                001445
 (1)  021434  063265
1738  021436
 (1)                001446
 (1)  021436  077220
1739  021440
 (1)                001447
 (1)  021440  054660
1740  021442
 (1)                001450
 (1)  021442  060365
1741  021444
 (1)                001451
 (1)  021444  115461
1742  021446
 (1)                001452
 (1)  021446  115063
1743  021450
 (1)                001453
 (1)  021450  020540
1744  021452
 (1)                001454
 (1)  021452  116057
1745  021454
 (1)                001455
 (1)  021454  000676
1746  021456
 (1)                001456
 (1)  021456  104425
1747  021460
 (1)                001457
 (1)  021460  000620
1748  021462
 (1)                001460
 (1)  021462  104425
1749
1750  021464
 (1)                001461
 (1)  021464  040364
1751  021466
 (1)                001462
 (1)  021466  114452
```

```
        ;FUGITIVE RECEIVE ROUTINES---DON'T FIT IN PAGE
RH1:    LDMA    BR,<INCA!SP15>          ;LD MAR WITH ADDR OF BA
        MICPC=MICPC+1
        <MOVE!LDMAR!BR!<INCA!SP15>>
        BRWRTE  BR,SELA!SP1             ;READ STATUS BYTE
        MICPC=MICPC+1
        <MOVE!WRTEBR!BR!<SELA!SP1>>
        BR1     RH4                     ;NO BUFF ASSIGNED IN MAINT MODE
        MICPC=MICPC+1
        <JUMP!BR1CON!<RH4-INIT&3000*4>!<RH4-INIT&777/2>>
        BRWRTE  IMM!INCMAR,77
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!INCMAR!<77>>
        SP      BR,AANDB,SP5
        MICPC=MICPC+1
        <MOVE!SPX!BR!AANDB!SP5>
        SP      BR!INCMAR,SELB,SP0      ;SAVE BYTE COUNT MASK
        MICPC=MICPC+1
        <MOVE!SPX!BR!INCMAR!SELB!SP0>
        BRWRTE  MEMX!INCMAR,AANDB!SP0   ;GET HIGH BYTE COUNT BITS
        MICPC=MICPC+1
        <MOVE!WRTEBR!MEMX!INCMAR!<AANDB!SP0>>
        CMP     BR,SP5                  ;COMPARE HIGH ORDER BITS OF COUNT
        MICPC=MICPC+1
        <SUBTC!BR!SP5>
        Z       RCLOW                   ;IF EQUAL COMPARE LOW ORDER BITS OF COUNT
        MICPC=MICPC+1
        <JUMP!ZCOND!<RCLOW-INIT&3000*4>!<RCLOW-INIT&777/2>>
RH6:    C       RCFATL                  ;IF CARRY--TOO BIG ERROR
        MICPC=MICPC+1
        <JUMP!CCOND!<RCFATL-INIT&3000*4>!<RCFATL-INIT&777/2>>
RH2:    BRWRTE  IBUS,IOBA1                      ;READ LOW BYTE OF IN BA
        MICPC=MICPC+1
        <MOVE!WRTEBR!IBUS!<IOBA1>>
        BR0     RCVODD                          ;IF SET IS ODD TRANSFER
        MICPC=MICPC+1
        <JUMP!BR0CON!<RCVODD-INIT&3000*4>!<RCVODD-INIT&777/2>>
RH4:    STATE   RCVKE0
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<RCVKE0-INIT&777/2>>
        ALWAYS  REXIT
        MICPC=MICPC+1
        <JUMP!ALCOND!<REXIT-INIT&3000*4>!<REXIT-INIT&777/2>>
RCVODD: STATE   RCVKO1
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<RCVKO1-INIT&777/2>>
        ALWAYS  REXIT
        MICPC=MICPC+1
        <JUMP!ALCOND!<REXIT-INIT&3000*4>!<REXIT-INIT&777/2>>
;
RCLOW:  CMP     MEMX,SP4                        ;COMPARE LOW ORDER BITS OF COUNT
        MICPC=MICPC+1
        <SUBTC!MEMX!SP4>
        ALWAYS  RH6
        MICPC=MICPC+1
        <JUMP!ALCOND!<RH6-INIT&3000*4>!<RH6-INIT&777/2>>
```

```
1752  021470
 (1)              001463
 (1)  021470  010151     RCFATL: LDMA    IMM,T
                                 MICPC=MICPC+1
                                 <MOVE!LDMAR!IMM!<T&377>>
1753  021472
 (1)              001464         MEMINC  IMM,2
 (1)  021472  016402             MICPC=MICPC+1
                                 <MOVE!WRMEM!INCMAR!IMM!<2>>
1754  021474
 (1)              001465         MEM     IMM,311
 (1)  021474  002711             MICPC=MICPC+1
                                 <MOVE!WRMEM!IMM!<311>>
1755  021476
 (1)              001466         LDMA    IMM,<<RTHRS+1>>          ;ADDRESS ERROR LINK
 (1)  021476  010175             MICPC=MICPC+1
                                 <MOVE!LDMAR!IMM!<<RTHRS+1>&377>>
1756  021500
 (1)              001467         MEMINC  IBUS,IOBA1
 (1)  021500  036540             MICPC=MICPC+1
                                 <MOVE!WRMEM!INCMAR!IBUS!<IOBA1>>
1757  021502
 (1)              001470         MEMINC  IBUS,IOBA2
 (1)  021502  036560             MICPC=MICPC+1
                                 <MOVE!WRMEM!INCMAR!IBUS!<IOBA2>>
1758  021504
 (1)              001471         BRWRTE  IMM,20
 (1)  021504  000420             MICPC=MICPC+1
                                 <MOVE!WRTEBR!IMM!<20>>
1759  021506
 (1)              001472 RCEXY:  MEMINC  IMM,0
 (1)  021506  016400             MICPC=MICPC+1
                                 <MOVE!WRMEM!INCMAR!IMM!<0>>
1760  021510
 (1)              001473         MEM     BR,SELB
 (1)  021510  062620             MICPC=MICPC+1
                                 <MOVE!WRMEM!BR!<SELB>>
1761  021512
 (1)              001474 RCEXX:  OUTPUT  IMM,<200!ORCVCO>         ;FLUSH INPUT SILO
 (1)  021512  002212             MICPC=MICPC+1
                                 <MOVE!WROUT!IMM!<200!ORCVCO>>
1762  021514
 (1)              001475         TSTATE  TMTA6
 (1)  021514  000404             MICPC=MICPC+1
                                 <MOVE!WRTEBR!IMM!<TMTA6-INIT&777/2>>
 (1)              001476         MICPC=MICPC+1
 (1)  021516  063222             <MOVE!SPX!BR!SELB!SP2>
1763  021520                     SP      IMM,1,SP1        ;SET INIT MODE IN PORT STATUS WORD
 (1)              001477         MICPC=MICPC+1
 (1)  021520  003001             <MOVE!SPX!IMM!1!SP1>
1764  021522                     ALWAYS  NTRS1
 (1)              001500         MICPC=MICPC+1
 (1)  021522  114642             <JUMP!ALCOND!<NTRS1-INIT&3000*4>!<NTRS1-INIT&777/2>>
1765
1766                        ;
1767  021524                     .SBTTL  START HANDLER
 (1)              001501 START:  BRWRTE  DP,<SELA!SP10>           ;READ LINE STATUS WORD
 (1)  021524  060610             MICPC=MICPC+1
                                 <MOVE!WRTEBR!DP!<SELA!SP10>>
1768  021526                     BRSHFT                           ;GET START MODE BIT IN POSITION
 (1)              001502         MICPC=MICPC+1
 (1)  021526  001620             <MOVE!SHFTBR!WRTEBR!SELB>
1769  021530                     BR4     10$                      ;IF IN START MODE SET STACK
 (1)              001503         MICPC=MICPC+1
 (1)  021530  117106             <JUMP!BR4CON!<10$-INIT&3000*4>!<10$-INIT&777/2>>
1770                                                             ;ELSE SET UP START ERROR
1771  021532                     BRWRTE  IMM,200
 (1)              001504         MICPC=MICPC+1
 (1)  021532  000600             <MOVE!WRTEBR!IMM!<200>>
```

```
1772  021534
 (1)            001505
 (1)  021534    104746
1773  021536
 (1)            001506
 (1)  021536    010151
1774  021540
 (1)            001507
 (1)  021540    016407
1775  021542
 (1)            001510
 (1)  021542    000411
1776  021544
 (1)            001511
 (1)  021544    110726
```

```
        ALWAYS  RCVM2
        MICPC=MICPC+1
        <JUMP!ALCOND!<RCVM2-INIT&3000*4>!<RCVM2-INIT&777/2>>
10$:    LDMA    IMM,T                   ;SET UP ADDR OF TYPE FIELD
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<T&377>>
        MEMINC  IMM,7                   ;WRITE STACK TYPE
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<7>>
        BRWRTE  IMM,11                  ;SEND THE UNNUMB MESS
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<11>>
        ALWAYS  SA2
        MICPC=MICPC+1
        <JUMP!ALCOND!<SA2-INIT&3000*4>!<SA2-INIT&777/2>>
```

```
1778  021546
 (1)              001512
 (1)  021546  040757
1779  021550
 (1)              001513
 (1)  021550  107573
1780  021552
 (1)              001514
 (1)  021552  070200
1781  021554
 (1)              001515
 (1)  021554  002400
1782  021556
 (1)              001516
 (1)  021556  010067
1783  021560
 (1)              001517
 (1)  021560  002471
1784  021562
 (1)              001520
 (1)  021562  000543
1785  021564
 (1)              001521
 (1)  021564  060360
1786  021566
 (1)              001522
 (1)  021566  115525
1787  021570
 (1)              001523
 (1)  021570  000406
1788  021572
 (1)              001524
 (1)  021572  062400
1789  021574
 (1)              001525
 (1)  021574  010241
1790  021576
 (1)              001526
 (1)  021576  053223
1791
1792  021600
 (1)              001527
 (1)  021600  016600
1793  021602
 (1)              001530
 (1)  021602  062460
1794  021604
 (1)              001531
 (1)  021604  010241
1795  021606
 (1)              001532
 (1)  021606  002642
1796  021610
 (1)              001533
 (1)  021610  000776
1797  021612
```

```
TDON3:  BRWRTE  MEMX,SUB!SP17                   ;COMPARE RESPONSE TO MSG NO
        MICPC=MICPC+1
        <MOVE!WRTEBR!MEMX!<SUB!SP17>>
        BR7     RH3                             ;IF NEGATIVE EXIT
        MICPC=MICPC+1
        <JUMP!BR7CON!<RH3-INIT&3000*4>!<RH3-INIT&777/2>>
TDON2:  LDMA    BR,SELA!SPO                     ;ADDRESS THE TRANSMITLINK
        MICPC=MICPC+1
        <MOVE!LDMAR!BR!<SELA!SPO>>
        MEM     IMM,0                           ;TURN OF ASSIGNEDAND TMTED BITS IN FLAG
        MICPC=MICPC+1
        <MOVE!WRMEM!IMM!<0>>
        LDMA    IMM,STC
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<STC&377>>
        MEM     IMM,TML1                        ;ASSUME WRAPAROUND
        MICPC=MICPC+1
        <MOVE!WRMEM!IMM!<TML1>>
        BRWRTE  IMM,TML8                        ;WRAPAROUND?
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<TML8>>
        CMP     BR,SPO
        MICPC=MICPC+1
        <SUBTC!BR!SPO>
        Z       TDON4                           ;YES
        MICPC=MICPC+1
        <JUMP!ZCOND!<TDON4-INIT&3000*4>!<TDON4-INIT&777/2>>
        BRWRTE  IMM,6                           ;OFFSET FOR NEXT TMT LINK
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<6>>
        MEM     BR,ADD!SPO                      ;UPDATE THE POINTER
        MICPC=MICPC+1
        <MOVE!WRMEM!BR!<ADD!SPO>>
TDON4:  LDMA    IMM,NXTSP                       ;ADDRESS DONE LINK
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<NXTSP&377>>
        LDMA    MEMX,SELB!SPX!SP3               ;ADDRESS THE LINK,COPYING
        MICPC=MICPC+1
        <MOVE!LDMAR!MEMX!<SELB!SPX!SP3>>
                                                ;ITS ADDRESS TO SPO
        MEMINC  IMM,200                         ;WRITE THE INTERRUPT TYPE
        MICPC=MICPC+1
        <MOVE!WRMEM!INCMAR!IMM!<200>>
        MEM     BR,INCA!SPO                     ;COPY ACTUAL LINK ADDRESS
        MICPC=MICPC+1
        <MOVE!WRMEM!BR!<INCA!SPO>>
        LDMA    IMM,NXTSP                       ;ADDRESS PTR INT STACK
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<NXTSP&377>>
        MEM     IMM,INTSTK                      ;ASSUME WRAP AROUND
        MICPC=MICPC+1
        <MOVE!WRMEM!IMM!<INTSTK>>
        BRWRTE  IMM,<<MMEND-2>>                 ;ADDRESS ENDOFINT STACK
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<<MMEND-2>>>
        CMP     BR,SP3                          ;WRAPAROUND?
```

```
  (1)           001534
  (1)  021612   060363          MICPC=MICPC+1
1798   021614                   <SUBTC!BR!SP3>
                                Z       TDON40                          ;YES---BRANCH
  (1)           001535          MICPC=MICPC+1
  (1)  021614   115540          <JUMP!ZCOND!<TDON40-INIT&3000*4>!<TDON40-INIT&777/2>>
1799   021616                   BRWRTE  IMM,2                           ;OFFSET TO NEXT PAIR
  (1)           001536          MICPC=MICPC+1
  (1)  021616   000402          <MOVE!WRTEBR!IMM!<2>>
1800   021620                   MEM     BR,ADD!SP3                      ;UPDATE POINTER
  (1)           001537          MICPC=MICPC+1
  (1)  021620   062403          <MOVE!WRMEM!BR!<ADD!SP3>>
1801   021622           TDON40: BRWRTE  IMM,20                          ;WRITE INTERUPT PENDING
  (1)           001540          MICPC=MICPC+1
  (1)  021622   000420          <MOVE!WRTEBR!IMM!<20>>
1802   021624                   SP      BR,AORB,SP1                     ;IN PORT STATUS WORD
  (1)           001541          MICPC=MICPC+1
  (1)  021624   063301          <MOVE!SPX!BR!AORB!SP1>
1803
1804   021626           TDON1:  LDMA    IMM,ISP17                       ;GET LAST ACKED
  (1)           001542          MICPC=MICPC+1
  (1)  021626   010153          <MOVE!LDMAR!IMM!<ISP17&377>>
1805   021630                   SP      MEMX,SELB,SP17                  ;STORE IT IN SP17
  (1)           001543          MICPC=MICPC+1
  (1)  021630   043237          <MOVE!SPX!MEMX!SELB!SP17>
1806   021632                   LDMA    IMM,STC                         ;GET START OF TMT CHAIN
  (1)           001544          MICPC=MICPC+1
  (1)  021632   010067          <MOVE!LDMAR!IMM!<STC&377>>
1807   021634                   LDMA    MEMX,SELB!SPBRX!SPO             ;ADDRESS THE LINK
  (1)           001545          MICPC=MICPC+1
  (1)  021634   053620          <MOVE!LDMAR!MEMX!<SELB!SPBRX!SPO>>
1808   021636                   BRWRTE  MEMX!INCMAR,SELB                ;GET THE FLAGS
  (1)           001546          MICPC=MICPC+1
  (1)  021636   054620          <MOVE!WRTEBR!MEMX!INCMAR!<SELB>>
1809   021640                   BR1     TDON3                           ;IF BUFFER ASSIGNED PROCEED
  (1)           001547          MICPC=MICPC+1
  (1)  021640   116512          <JUMP!BR1CON!<TDON3-INIT&3000*4>!<TDON3-INIT&777/2>>
1810   021642                   ALWAYS  RH3                             ;ELSE---EXIT
  (1)           001550          MICPC=MICPC+1
  (1)  021642   104573          <JUMP!ALCOND!<RH3-INIT&3000*4>!<RH3-INIT&777/2>>
1811
1812                            ;
1813   021644           RESEXT: BRWRTE  IMM,4                           ;ADD TO MXT BITS
  (1)           001551          MICPC=MICPC+1
  (1)  021644   000404          <MOVE!WRTEBR!IMM!<4>>
1814   021646                   SP      BR,ADD,SPO
  (1)           001552          MICPC=MICPC+1
  (1)  021646   063000          <MOVE!SPX!BR!ADD!SPO>
1815   021650                   ALWAYS  TH3X
  (1)           001553          MICPC=MICPC+1
  (1)  021650   110574          <JUMP!ALCOND!<TH3X-INIT&3000*4>!<TH3X-INIT&777/2>>
```

```
1817
1818
1819
1820
1821  021652
 (1)              001554
 (1)  021652      023333
1822  021654
 (1)              001555
 (1)  021654      115561
1823  021656
 (1)              001556
 (1)  021656      000406
1824  021660
 (1)              001557
 (1)  021660      060360
1825  021662
 (1)              001560
 (1)  021662      115566
1826  021664
 (1)              001561
 (1)  021664      060601
1827  021666
 (1)              001562
 (1)  021666      106753
1828  021670
 (1)              001563
 (1)  021670      060521
1829  021672
 (1)              001564
 (1)  021672      107345
1830  021674
 (1)              001565
 (1)  021674      104661
1831  021676
 (1)              001566
 (1)  021676      063164
1832  021700
 (1)              001567
 (1)  021700      000750
1833  021702
 (1)              001570
 (1)  021702      104425
1834
1835
1836
1837  021704
 (1)              001571
 (1)  021704      010001
1838  021706
 (1)              001572
 (1)  021706      002401
1839  021710
 (1)              001573
 (1)  021710      010167
1840  021712
```

```
;
; INPUTS:
;       SPO = RECEIVE CHARACTER

PASWRD: SP      IBUS,LNOSW,SP13         ;READ PASSWD SWITCH
        MICPC=MICPC+1
        <MOVE!SPX!IBUS!LNOSW!SP13>
        Z       10$                     ;IF ALL ONES NO RLD ENABLED
        MICPC=MICPC+1
        <JUMP!ZCOND!<10$-INIT&3000*4>!<10$-INIT&777/2>>
        BRWRTE  IMM,6                   ;CHECK FOR ENTER MOP MODE
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<6>>
        CMP     BR,SPO
        MICPC=MICPC+1
        <SUBTC!BR!SPO>
        Z       20$                     ;IF EQUAL===ENTER MOP
        MICPC=MICPC+1
        <JUMP!ZCOND!<20$-INIT&3000*4>!<20$-INIT&777/2>>
10$:    BRWRTE  BR,SELA!SP1             ;READ STATUS BYTE
        MICPC=MICPC+1
        <MOVE!WRTEBR!BR!<SELA!SP1>>
        BR1     RHX                     ;MESSAGE WITH NO BUFFER ASSIGNED
        MICPC=MICPC+1
        <JUMP!BR1CON!<RHX-INIT&3000*4>!<RHX-INIT&777/2>>
        BRWRTE  BR,AA!SP1
        MICPC=MICPC+1
        <MOVE!WRTEBR!BR!<AA!SP1>>
        BR4     RCVMO                   ;DLE RECEIVED IN NORMAL MODE
        MICPC=MICPC+1
        <JUMP!BR4CON!<RCVMO-INIT&3000*4>!<RCVMO-INIT&777/2>>
        ALWAYS  RK3                     ;HANDLE MAINT MODE MESSAGE
        MICPC=MICPC+1
        <JUMP!ALCOND!<RK3-INIT&3000*4>!<RK3-INIT&777/2>>
20$:    SP      BR,DECA,SP4             ;COUNT FOR NUMB OF COMPARES
        MICPC=MICPC+1
        <MOVE!SPX!BR!DECA!SP4>
        STATE   EM2
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<EM2-INIT&777/2>>
        ALWAYS  REXIT
        MICPC=MICPC+1
        <JUMP!ALCOND!<REXIT-INIT&3000*4>!<REXIT-INIT&777/2>>
;
        .ENABL  LSB
;
RCVM1:  LDMA    IMM,NAKST               ;RESET NAKS SENT
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<NAKST&377>>
        MEM     IMM,1                   ;..
        MICPC=MICPC+1
        <MOVE!WRMEM!IMM!<1>>
        LDMA    IMM,BC                  ;ADDRESS ORIGINAL RECV BYTE COUNT
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<BC&377>>
        SP      MEMX!INCMAR,SELB,SP4    ;MOVE BYTE COUNT TO SP4
```

```
                001574
 (1)
 (1)  021712    057224          MICPC=MICPC+1
1841  021714                    <MOVE!SPX!MEMX!INCMAR!SELB!SP4>
                                SP      MEMX!INCMAR,SELB,SP5    ;AND SP5
 (1)            001575          MICPC=MICPC+1
 (1)  021714    057225          <MOVE!SPX!MEMX!INCMAR!SELB!SP5>
1842  021716                    MEM     BR,DECA!SP11                    ;COPY SP11 TO MEMORY
 (1)            001576          MICPC=MICPC+1
 (1)  021716    062571          <MOVE!WRMEM!BR!<DECA!SP11>>
1843  021720                    LDMA    IMM,NXTSP
 (1)            001577          MICPC=MICPC+1
 (1)  021720    010241          <MOVE!LDMAR!IMM!<NXTSP&377>>
1844  021722                    SP      MEMX!LDMAR,SELB,SP3             ;COPY TO SP3
 (1)            001600          MICPC=MICPC+1
 (1)  021722    053223          <MOVE!SPX!MEMX!LDMAR!SELB!SP3>
1845  021724                    MEMINC  IMM,204                 ;RECEIVE DONE IMAGE
 (1)            001601          MICPC=MICPC+1
 (1)  021724    016604          <MOVE!WRMEM!INCMAR!IMM!<204>>
1846  021726                    MEM     BR!LDMAR,SELA!SP15              ;COPY LINK ADDRESS TO NEXT INTER
 (1)            001602          MICPC=MICPC+1
 (1)  021726    072615          <MOVE!WRMEM!BR!LDMAR!<SELA!SP15>>
1847  021730                    MEMINC  IMM,0                   ;ZERO THE FLAGS
 (1)            001603          MICPC=MICPC+1
 (1)  021730    016400          <MOVE!WRMEM!INCMAR!IMM!<0>>
1848  021732                    SP      IMM!INCMAR,SP0,300              ;WRITE A 300 TO SP0
 (1)            001604          MICPC=MICPC+1
 (1)  021732    017300          <MOVE!SPX!IMM!INCMAR!SP0!300>
1849  021734                    BRWRTE  IMM!INCMAR,2                    ;PREPARE TO ADDRESS NEXT
 (1)            001605          MICPC=MICPC+1
 (1)  021734    014402          <MOVE!WRTEBR!IMM!INCMAR!<2>>
1850                                                                    ;INTERRUPT STACK AND INCREMENT
1851                                                                    ;THE MAR
1852  021736                    MEM     MEMX,AANDB!SP0                  ;MASK OFF ORIGINAL HIGH BYTE
 (1)            001606          MICPC=MICPC+1
 (1)  021736    042660          <MOVE!WRMEM!MEMX!<AANDB!SP0>>
1853                                                                    ;OF COUNT SAVING EXTENDED MEM BITS
1854  021740                    MEMINC  MEMX,AORB!SP5                   ;COPY TO MEMORY LINK
 (1)            001607          MICPC=MICPC+1
 (1)  021740    056705          <MOVE!WRMEM!INCMAR!MEMX!<AORB!SP5>>
1855  021742                    MEMINC  BR,SELA!SP4
 (1)            001610          MICPC=MICPC+1
 (1)  021742    076604          <MOVE!WRMEM!INCMAR!BR!<SELA!SP4>>
1856  021744                    LDMA    IMM,NXTSP                       ;ADDRESS NEXT INT STACK
 (1)            001611          MICPC=MICPC+1
 (1)  021744    010241          <MOVE!LDMAR!IMM!<NXTSP&377>>
1857  021746                    MEM     BR,ADD!SP3
 (1)            001612          MICPC=MICPC+1
 (1)  021746    062403          <MOVE!WRMEM!BR!<ADD!SP3>>
1858  021750                    BRWRTE  IMM,<<MMEND-2>>                 ;ADDRESSEND OF INT STACK
 (1)            001613          MICPC=MICPC+1
 (1)  021750    000776          <MOVE!WRTEBR!IMM!<<MMEND-2>>>
1859  021752                    CMP     BR,SP3                          ;WRAP AROUND
 (1)            001614          MICPC=MICPC+1
 (1)  021752    060363          <SUBTC!BR!SP3>
1860  021754                    Z       40$                             ;IF YES-- BRANCH
 (1)            001615          MICPC=MICPC+1
 (1)  021754    115626          <JUMP!ZCOND!<40$-INIT&3000*4>!<40$-INIT&777/2>>
```

```
1861  021756
1862  021756
 (1)            001616
 (1)  021756    000462          20$:    BRWRTE  IMM,RCL7
                                        MICPC=MICPC+1
                                        <MOVE!WRTEBR!IMM!<RCL7>>
1863  021760                            CMP     BR,SP15
 (1)            001617                 MICPC=MICPC+1
 (1)  021760    060375                 <SUBTC!BR!SP15>
1864  021762                            Z       50$
 (1)            001620                 MICPC=MICPC+1
 (1)  021762    115630                 <JUMP!ZCOND!<50$-INIT&3000*4>!<50$-INIT&777/2>>
1865  021764                            BRWRTE  IMM,5
 (1)            001621                 MICPC=MICPC+1
 (1)  021764    000405                 <MOVE!WRTEBR!IMM!<5>>
1866  021766                    30$:    SP      BR,ADD,SP15
 (1)            001622                 MICPC=MICPC+1
 (1)  021766    063015                 <MOVE!SPX!BR!ADD!SP15>
1867  021770                            BRWRTE  IMM,20                  ;MASK FOR INTERUPT PENDING
 (1)            001623                 MICPC=MICPC+1
 (1)  021770    000420                 <MOVE!WRTEBR!IMM!<20>>
1868  021772                            SP      DP,AORB,SP1             ;UPDATE PORT STATUS WORD
 (1)            001624                 MICPC=MICPC+1
 (1)  021772    063301                 <MOVE!SPX!DP!AORB!SP1>
1877  021774                            ALWAYS  FLUSH
 (1)            001625                 MICPC=MICPC+1
 (1)  021774    104415                 <JUMP!ALCOND!<FLUSH-INIT&3000*4>!<FLUSH-INIT&777/2>>
1879  021776                    40$:    MEM     IMM,INTSTK              ;POINT TO START OF INTERRUPT STACK
 (1)            001626                 MICPC=MICPC+1
 (1)  021776    002642                 <MOVE!WRMEM!IMM!<INTSTK>>
1880  022000                            ALWAYS  20$
 (1)            001627                 MICPC=MICPC+1
 (1)  022000    114616                 <JUMP!ALCOND!<20$-INIT&3000*4>!<20$-INIT&777/2>>
1881  022002                    50$:    BRWRTE  IMM,<<RCL1-RCL7>&377>   ;POINT TO START OF QUEUE
 (1)            001630                 MICPC=MICPC+1
 (1)  022002    000742                 <MOVE!WRTEBR!IMM!<<RCL1-RCL7>&377>>
1882  022004                            ALWAYS  30$
 (1)            001631                 MICPC=MICPC+1
 (1)  022004    114622                 <JUMP!ALCOND!<30$-INIT&3000*4>!<30$-INIT&777/2>>
1883                                    .DSABL  LSB
1884  022006                    NTHRES: LDMA    IMM,ST
 (1)            001632                 MICPC=MICPC+1
 (1)  022006    010152                 <MOVE!LDMAR!IMM!<ST&377>>
1885  022010                            SPBR    MEMX,SELB,SPO
 (1)            001633                 MICPC=MICPC+1
 (1)  022010    043620                 <MOVE!SPBRX!MEMX!SELB!SPO>
1886  022012                            BRWRTE  BR,ADD!SPO                      ;SHIFT LEFT
 (1)            001634                 MICPC=MICPC+1
 (1)  022012    060400                 <MOVE!WRTEBR!BR!<ADD!SPO>>
1887  022014                            BR4     OVRRUN
 (1)            001635                 MICPC=MICPC+1
 (1)  022014    103367                 <JUMP!BR4CON!<OVRRUN-INIT&3000*4>!<OVRRUN-INIT&777/2>>
1888  022016                            BRWRTE  IMM,1
 (1)            001636                 MICPC=MICPC+1
 (1)  022016    000401                 <MOVE!WRTEBR!IMM!<1>>
1889  022020                    ERRXX:
1890  022020                    NTRS0:  LDMA    IMM,<<RTHRS+3>>
 (1)            001637                 MICPC=MICPC+1
```

```
                                        <MOVE!LDMAR!IMM!<<RTHRS+3>&377>>
  (1)  022020  010177                   MEMINC  IMM,0
1891   022022
  (1)          001640                   MICPC=MICPC+1
  (1)  022022  016400                   <MOVE!WRMEM!INCMAR!IMM!<0>>
1892   022024                           MEM     BR,SELB
  (1)          001641                   MICPC=MICPC+1
  (1)  022024  062620                   <MOVE!WRMEM!BR!<SELB>>
1893   022026                   NTRS1:  LDMA    IMM,NXTSP
  (1)          001642                   MICPC=MICPC+1
  (1)  022026  010241                   <MOVE!LDMAR!IMM!<NXTSP&377>>
1894   022030                           LDMA    MEMX,SELB!SPX!SPO
  (1)          001643                   MICPC=MICPC+1
  (1)  022030  053220                   <MOVE!LDMAR!MEMX!<SELB!SPX!SPO>>
1895   022032                           MEMINC  IMM,201
  (1)          001644                   MICPC=MICPC+1
  (1)  022032  016601                   <MOVE!WRMEM!INCMAR!IMM!<201>>
1896   022034                           MEM     IMM,<<RTHRS>>
  (1)          001645                   MICPC=MICPC+1
  (1)  022034  002574                   <MOVE!WRMEM!IMM!<<RTHRS>>>
1897   022036                           LDMA    IMM,NXTSP
  (1)          001646                   MICPC=MICPC+1
  (1)  022036  010241                   <MOVE!LDMAR!IMM!<NXTSP&377>>
1898   022040                           MEM     IMM,INTSTK              ;ASSUME QUEUE WRAP AROUND
  (1)          001647                   MICPC=MICPC+1
  (1)  022040  002642                   <MOVE!WRMEM!IMM!<INTSTK>>
1899   022042                           BRWRTE  IMM,<<MMEND-2>>
  (1)          001650                   MICPC=MICPC+1
  (1)  022042  000776                   <MOVE!WRTEBR!IMM!<<MMEND-2>>>
1900   022044                           CMP     BR,SPO
  (1)          001651                   MICPC=MICPC+1
  (1)  022044  060360                   <SUBTC!BR!SPO>
1901   022046                           Z       NTRS2                   ;IT DID WRAP AROUND
  (1)          001652                   MICPC=MICPC+1
  (1)  022046  115655                   <JUMP!ZCOND!<NTRS2-INIT&3000*4>!<NTRS2-INIT&777/2>>
1902   022050                           BRWRTE  IMM,2                   ;OFFSET TO NEXT PAIR
  (1)          001653                   MICPC=MICPC+1
  (1)  022050  000402                   <MOVE!WRTEBR!IMM!<2>>
1903   022052                           MEM     BR,ADD!SPO              ;UPDATE QUEUE POINTER
  (1)          001654                   MICPC=MICPC+1
  (1)  022052  062400                   <MOVE!WRMEM!BR!<ADD!SPO>>
1904   022054                   NTRS2:  BRWRTE  IMM,20
  (1)          001655                   MICPC=MICPC+1
  (1)  022054  000420                   <MOVE!WRTEBR!IMM!<20>>
1905   022056                           SPBR    BR,AORB,SP1
  (1)          001656                   MICPC=MICPC+1
  (1)  022056  063701                   <MOVE!SPBRX!BR!AORB!SP1>
1906   022060                           BRO     TAB1                    ;FLAGGED BY ERROR TYPE
  (1)          001657                   MICPC=MICPC+1
  (1)  022060  116320                   <JUMP!BROCON!<TAB1-INIT&3000*4>!<TAB1-INIT&777/2>>
1907   022062                   SNAK:   LDMA    IMM,ISP11
  (1)          001660                   MICPC=MICPC+1
  (1)  022062  010171                   <MOVE!LDMAR!IMM!<ISP11&377>>
1908   022064                           SP      MEMX,SELB,SP11
  (1)          001661                   MICPC=MICPC+1
  (1)  022064  043231                   <MOVE!SPX!MEMX!SELB!SP11>
1909   022066                           SP      BR,INCA,SP11            ;INCREMENT MSG EXPECTED
```

```
(1)          001662          MICPC=MICPC+1
(1)  022066  063071          <MOVE!SPX!BR!INCA!SP11>
1910 022070          SNAK1:  BRWRTE  IMM,1                   ;UNNUMB PENING BIT TO BR
(1)          001663          MICPC=MICPC+1
(1)  022070  000401          <MOVE!WRTEBR!IMM!<1>>
1911 022072          SNAK2:  SP      BP,AORB,SP10            ;UPDATE LINE STATUS WORD
(1)          001664          MICPC=MICPC+1
(1)  022072  063310          <MOVE!SPX!BR!AORB!SP10>
1912 022074                  ALWAYS  FLUSH
(1)          001665          MICPC=MICPC+1
(1)  022074  104415          <JUMP!ALCOND!<FLUSH-INIT&3000*4>!<FLUSH-INIT&777/2>>
```

```
1914  022076
 (1)            001666
 (1)  022076    000424
1915  022100
 (1)            001667
 (1)  022100    062226
1916  022102
 (1)            001670
 (1)  022102    000400
1917  022104
 (1)            001671
 (1)  022104    062227
1918  022106
 (1)            001672
 (1)  022106    023740
1919  022110
 (1)            001673
 (1)  022110    062222
1920  022112
 (1)            001674
 (1)  022112    000766
1921  022114
 (1)            001675
 (1)  022114    062223
1922  022116
 (1)            001676
 (1)  022116    000421
1923  022120
 (1)            001677
 (1)  022120    061230
1924  022122
 (1)            001700
 (1)  022122    061231
1925  022124
 (1)            001701
 (1)  022124    120620
1926  022126
 (1)            001702
 (1)  022126    103355
1927  022130
 (1)            001703
 (1)  022130    120600
1928  022132
 (1)            001704
 (1)  022132    116301
1929  022134
 (1)            001705
 (1)  022134    020540
1930  022136
 (1)            001706
 (1)  022136    116713
1931  022140
 (1)            001707
 (1)  022140    002026
1932  022142
 (1)            001710
```

```
EMTRIG: BRWRTE  IMM,24
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<24>>
        OUTPUT  BR,<SELB!OBA1>
        MICPC=MICPC+1
        <MOVE!WROUT!BR!<SELB!OBA1>>
        BRWRTE  IMM,0
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<0>>
        OUTPUT  BR,<SELB!OBA2>
        MICPC=MICPC+1
        <MOVE!WROUT!BR!<SELB!OBA2>>
        SPBR    IBUS,BM873,SPO          ;READ BM873 ADDRESS---
        MICPC=MICPC+1
        <MOVE!SPBRX!IBUS!BM873!SPO>
        OUTPUT  BR,SELB!OUTDA1          ;SET UP LOW BYTE OF ADDRESS
        MICPC=MICPC+1
        <MOVE!WROUT!BR!<SELB!OUTDA1>>
        BRWRTE  IMM,366                 ;HIGH BYTE BASE FOR ROM BOOT
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<366>>
EM4:    OUTPUT  BR,SELB!OUTDA2          ;SET UP HIGH BYTE OF ADDRESS
        MICPC=MICPC+1
        <MOVE!WROUT!BR!<SELB!OUTDA2>>
EM6:    BRWRTE  IMM,21                  ;MASK TO START NPR
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<21>>
        OUT     BR,<SELB!ONPR>
        MICPC=MICPC+1
        <MOVE!WROUTX!BR!<SELB!ONPR>>
        OUT     BR,<SELB!OBR>           ;START TIMER
        MICPC=MICPC+1
        <MOVE!WROUTX!BR!<SELB!OBR>>
CKTIME: BRWRTE  IBUS,UBBR               ;CHECK TIMER
        MICPC=MICPC+1
        <MOVE!WRTEBR!IBUS!<UBBR>>
        BR4     HALTED                  ;NPR TIMED OUT DO ACLOW
        MICPC=MICPC+1
        <JUMP!BR4CON!<HALTED-INIT&3000*4>!<HALTED-INIT&777/2>>
EM1:    BRWRTE  IBUS,NPR                ;READ NPR CONTROL
        MICPC=MICPC+1
        <MOVE!WRTEBR!IBUS!<NPR>>
        BRO     CKTIME
        MICPC=MICPC+1
        <JUMP!BROCON!<CKTIME-INIT&3000*4>!<CKTIME-INIT&777/2>>
        BRWRTE  IBUS,IOBA1              ;READ LOW BYTE OF ADDRESS BACK
        MICPC=MICPC+1
        <MOVE!WRTEBR!IBUS!<IOBA1>>
        BR1     EM5                     ;ADDRESS IS 26-DONE
        MICPC=MICPC+1
        <JUMP!BR1CON!<EM5-INIT&3000*4>!<EM5-INIT&777/2>>
        OUTPUT  IMM,<26!OBA1>           ;SET UP LOW BYTE OF ADDRESS
        MICPC=MICPC+1
        <MOVE!WROUT!IMM!<26!OBA1>>
        BRWRTE  IMM,0                   ;CLEAR BR TO CLEAR LOW BYTE OF
        MICPC=MICPC+1
```

```
                START HANDLER

                        <MOVE!WRTEBR!IMM!<0>>
                                                        ;DATA
                OUTPUT  BR,SELB!OUTDA1                  ;ZERO LOW BYTE OF DATA
                MICPC=MICPC+1
                <MOVE!WROUT!BR!<SELB!OUTDA1>>
                ALWAYS  EM4                             ;GO DO NPR
                MICPC=MICPC+1
                <JUMP!ALCOND!<EM4-INIT&3000*4>!<EM4-INIT&777/2>>
EM5:            BRADDR  RM1                                     ;IF NPR DONE
                MICPC=MICPC+1
                <MOVE!WRTEBR!<RM1-INIT&777/2>>
                ALWAYS  ACLOW
                MICPC=MICPC+1
                <JUMP!ALCOND!<ACLOW-INIT&3000*4>!<ACLOW-INIT&777/2>>
TABUPD:         SPBR    IBUS,RCVCON,SPO                 ;READ RECEIVER CONTROL REG
                MICPC=MICPC+1
                <MOVE!SPBRX!IBUS!RCVCON!SPO>
                BRWRTE  BR,ADD!SPO                      ;SHIFT LEFT
                MICPC=MICPC+1
                <MOVE!WRTEBR!BR!<ADD!SPO>>
                BR7     IDLE                            ;RECEIVE ACTIVE--IDLE
                MICPC=MICPC+1
                <JUMP!BR7CON!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
TAB1:           LDMA    IMM,IMG10
                MICPC=MICPC+1
                <MOVE!LDMAR!IMM!<IMG10&377>>
                BRWRTE  IMM,42
                MICPC=MICPC+1
                <MOVE!WRTEBR!IMM!<42>>
                MEMINC  BR,AANDB!SP10                   ;SAVE BITS 1 AND 5 OF SP10
                MICPC=MICPC+1
                <MOVE!WRMEM!INCMAR!BR!<AANDB!SP10>>
                MEMINC  BR,SELA!SP11                    ;SAVE SP11
                MICPC=MICPC+1
                <MOVE!WRMEM!INCMAR!BR!<SELA!SP11>>
                MEMINC  BR,SELA!SP15                    ;SAVE SP15
                MICPC=MICPC+1
                <MOVE!WRMEM!INCMAR!BR!<SELA!SP15>>
                MEMINC  BR,SELA!SP17                    ;SAVE SP17
                MICPC=MICPC+1
                <MOVE!WRMEM!INCMAR!BR!<SELA!SP17>>
                                                        ;MAR NOW POINTS TO BASE
                STATE   RB2                     ;DO NOT CHANGE BR UNTIL RB0
                MICPC=MICPC+1
                <MOVE!WRTEBR!IMM!<RB2-INIT&777/2>>
                LDMA    IMM,TABST               ;POINT TO TABLE UPDATE STATE
                MICPC=MICPC+1
                <MOVE!LDMAR!IMM!<TABST&377>>
                PSTATE  TBU1                            ;NEW PORT STATE ADDRESS
                MEM     IMM,<<TBU1-INIT&777/2>>
                MICPC=MICPC+1
                <MOVE!WRMEM!IMM!<<TBU1-INIT&777/2>>>
                SP      IMM,4,SP4                       ;INITIALIZE COUNT
                MICPC=MICPC+1
                <MOVE!SPX!IMM!4!SP4>
                                                        ;NOTE: FIRST 6 RAM LOCATIONS ARE NOT WRITTEN
```

```
  (1)   022142  000400
1933
1934    022144
  (1)           001711
  (1)   022144  062222
1935    022146
  (1)           001712
  (1)   022146  114675
1936    022150
  (1)           001713
  (1)   022150  000757
1937    022152
  (1)           001714
  (1)   022152  100756
1938    022154
  (1)           001715
  (1)   022154  023640
1939    022156
  (1)           001716
  (1)   022156  060400
1940    022160
  (1)           001717
  (1)   022160  103447
1941    022162
  (1)           001720
  (1)   022162  010154
1942    022164
  (1)           001721
  (1)   022164  000442
1943    022166
  (1)           001722
  (1)   022166  076670
1944    022170
  (1)           001723
  (1)   022170  076611
1945    022172
  (1)           001724
  (1)   022172  076615
1946    022174
  (1)           001725
  (1)   022174  076617
1947
1948    022176
  (1)           001726
  (1)   022176  000461
1949    022200
  (1)           001727
  (1)   022200  010210
1950    022202
  (1)   022202
  (2)           001730
  (2)   022202  002764
1951    022204
  (1)           001731
  (1)   022204  003004
1952
```

```
 1953
 1954  022206                                                    ;TO CORE TABLE.
  (1)            001732          LDMA    IMM,BASE
  (1)  022206    010017          MICPC=MICPC+1
 1955  022210                    <MOVE!LDMAR!IMM!<BASE&377>>
  (1)            001733          ALWAYS  RBO
  (1)  022210    104443          MICPC=MICPC+1
                                 <JUMP!ALCOND!<RBO-INIT&3000*4>!<RBO-INIT&777/2>>
```

E02


```
1957  022212
 (1)            001734
 (1)  022212    000402
1958  022214
 (1)            001735
 (1)  022214    063004
1959  022216
 (1)            001736
 (1)  022216    023140
1960  022220
 (1)            001737
 (1)  022220    062006
1961  022222
 (1)            001740
 (1)  022222    023160
1962  022224
 (1)            001741
 (1)  022224    062107
1963  022226
 (1)            001742
 (1)  022226    101270
1964  022230
 (1)            001743
 (1)  022230    060601
1965  022232
 (1)            001744
 (1)  022232    116347
1966
1967  022234
 (1)            001745
 (1)  022234    070604
1968  022236
 (1)            001746
 (1)  022236    117355
1969  022240
 (1)            001747
 (1)  022240    070604
1970  022242
 (1)            001750
 (1)  022242    117755
1971  022244
 (1)            001751
 (1)  022244    056222
1972  022246
 (1)            001752
 (1)  022246    056223
1973  022250
 (1)            001753
 (1)  022250    123200
1974  022252
 (1)            001754
 (1)  022252    104634
1975  022254
 (1)            001755
 (1)  022254    010210
1976  022256
```

```
EC2:    BRWRTE  IMM,2                           ;INCREMENT COUNT/TEST
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<2>>
        SP      BR,ADD,SP4
        MICPC=MICPC+1
        <MOVE!SPX!BR!ADD!SP4>
        SP      IBUS,IOBA1,SPO                  ;POINT TO NEXT ADDRESS
        MICPC=MICPC+1
        <MOVE!SPX!IBUS!IOBA1!SPO>
        OUTPUT  BR,ADD!OBA1
        MICPC=MICPC+1
        <MOVE!WROUT!BR!<ADD!OBA1>>
        SP      IBUS,IOBA2,SPO
        MICPC=MICPC+1
        <MOVE!SPX!IBUS!IOBA2!SPO>
        OUTPUT  BR,AC!OBA2
        MICPC=MICPC+1
        <MOVE!WROUT!BR!<AC!OBA2>>
        C       TABMXT
        MICPC=MICPC+1
        <JUMP!CCOND!<TABMXT-INIT&3000*4>!<TABMXT-INIT&777/2>>
ECX:    BRWRTE  BR,SELA!SP1                     ;READ PORT STATUS
        MICPC=MICPC+1
        <MOVE!WRTEBR!BR!<SELA!SP1>>
        BRO     30$                             ;INIT MODE, WRITE OUT 200 BYTES
        MICPC=MICPC+1
        <JUMP!BROCON!<30$-INIT&3000*4>!<30$-INIT&777/2>>
                                                ;OTHERWISE ONLY WRITE OUT ERROR COUNTERS
        BRWRTE  BR!LDMAR,SELA!SP4               ;READ COUNTER
        MICPC=MICPC+1
        <MOVE!WRTEBR!BR!LDMAR!<SELA!SP4>>
        BR4     20$                             ;ALL DONE
        MICPC=MICPC+1
        <JUMP!BR4CON!<20$-INIT&3000*4>!<20$-INIT&777/2>>
30$:    BRWRTE  BR!LDMAR,SELA!SP4               ;READ CTR
        MICPC=MICPC+1
        <MOVE!WRTEBR!BR!LDMAR!<SELA!SP4>>
        BR7     20$                             ;ALL DONE
        MICPC=MICPC+1
        <JUMP!BR7CON!<20$-INIT&3000*4>!<20$-INIT&777/2>>
        OUTPUT  MEMX!INCMAR,SELB!OUTDA1         ;STORE COUNTS OF ERRORS
        MICPC=MICPC+1
        <MOVE!WROUT!MEMX!INCMAR!<SELB!OUTDA1>>
        OUTPUT  MEMX!INCMAR,SELB!OUTDA2
        MICPC=MICPC+1
        <MOVE!WROUT!MEMX!INCMAR!<SELB!OUTDA2>>
        SP      IBUS,NPR,SPO
        MICPC=MICPC+1
        <MOVE!SPX!IBUS!NPR!SPO>
        ALWAYS  RK8
        MICPC=MICPC+1
        <JUMP!ALCOND!<RK8-INIT&3000*4>!<RK8-INIT&777/2>>
20$:    LDMA    IMM,TABST
        MICPC=MICPC+1
        <MOVE!LDMAR!IMM!<TABST&377>>
        PSTATE  I3
```

START HANDLER

```
  (1)  022256
  (2)         001756
  (2)  022256 002455
1980   022260
1981   022260
  (1)         001757
  (1)  022260 000400
1982   022262
  (1)         001760
  (1)  022262 104425
1983
1984   022264
  (1)         001761
  (1)  022264 060530
1985   022266
  (1)         001762
  (1)  022266 103373
1986   022270
  (1)         001763
  (1)  022270 107422
1987
1988   022272
  (1)         001764
  (1)  022272 100447
1990   022274
  (1)         001765
  (1)  022274 000600
1991   022276
  (1)         001766
  (1)  022276 063301
1992   022300
  (1)         001767
  (1)  022300 000604
1993   022302
  (1)         001770
  (1)  022302 063230
1994   022304
  (1)         001771
  (1)  022304 000400
  (1)         001772
  (1)  022306 063222
1995   022310
  (1)         001773
  (1)  022310 100662
2004   022312
  (1)         001774
  (1)  022312 023640
2005   022314
  (1)         001775
  (1)  022314 060400
2006   022316
  (1)         001776
  (1)  022316 103451
2007   022320
  (1)         001777
  (1)  022320 110402
```

```
        MEM     IMM,<<I3-INIT&777/2>>
        MICPC=MICPC+1
        <MOVE!WRMEM!IMM!<<I3-INIT&777/2>>>
RM1:
        STATE   RCVA
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<RCVA-INIT&777/2>>
        ALWAYS  REXIT
        MICPC=MICPC+1
        <JUMP!ALCOND!<REXIT-INIT&3000*4>!<REXIT-INIT&777/2>>
        .
ACK:    BRWRTE  BR,AA!SP10                      ;READ LINE STATUS SHIFTING LEFT
        MICPC=MICPC+1
        <MOVE!WRTEBR!BR!<AA!SP10>>
        BR4     ACK1                            ;IF START RCVD--CLEAR START MODE
        MICPC=MICPC+1
        <JUMP!BR4CON!<ACK1-INIT&3000*4>!<ACK1-INIT&777/2>>
        BR7     FLUSHA                          ;IF HALF DUPLEX-SEND AN ACK
        MICPC=MICPC+1
        <JUMP!BR7CON!<FLUSHA-INIT&3000*4>!<FLUSHA-INIT&777/2>>
                                                ;IF NOTHING ELSE TO SEND
        ALWAYS IDLE
        MICPC=MICPC+1
        <JUMP!ALCOND!<IDLE-INIT&3000*4>!<IDLE-INIT&777/2>>
CBOOT:  BRWRTE  IMM,200                         ;MASK FOR BOOT MODE
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<200>>
        SP      BR,AORB,SP1                     ;IN PORT STATUS WORD
        MICPC=MICPC+1
        <MOVE!SPX!BR!AORB!SP1>
        BRWRTE  IMM,204                         ;MASK FOR OK TOSEND AND LINE IDLE
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<204>>
        SP      BR,SELB,SP10                    ;IN LINE STATUS
        MICPC=MICPC+1
        <MOVE!SPX!BR!SELB!SP10>
        TSTATE  TMTA
        MICPC=MICPC+1
        <MOVE!WRTEBR!IMM!<TMTA-INIT&777/2>>
        MICPC=MICPC+1
        <MOVE!SPX!BR!SELB!SP2>
        ALWAYS  INS12
        MICPC=MICPC+1
        <JUMP!ALCOND!<INS12-INIT&3000*4>!<INS12-INIT&777/2>>
RCVCK:  SPBR    IBUS,RCVCON,SP0
        MICPC=MICPC+1
        <MOVE!SPBRX!IBUS!RCVCON!SP0>
        BRWRTE  BR,ADD!SP0
        MICPC=MICPC+1
        <MOVE!WRTEBR!BR!<ADD!SP0>>
        BR7     I1
        MICPC=MICPC+1
        <JUMP!BR7CON!<I1-INIT&3000*4>!<I1-INIT&777/2>>
        ALWAYS  TA1
        MICPC=MICPC+1
        <JUMP!ALCOND!<TA1-INIT&3000*4>!<TA1-INIT&777/2>>
```

```
DDCGAH.MAC    14-DEC-77 15:07          START HANDLER

  2009          000001                          .END

. ABS.  022322      000

 ERRORS DETECTED:  0

 .DDCMP/CRF/DS:CRF←DMCNEW,HI1,DDCGAH
RUN-TIME:  6 10 0 SECONDS
RUN-TIME RATIO: 77/16=4.6
CORE USED:  6K  (11 PAGES)

  1671
```

```
                                        ;*************************** TEST 1 ****************************
                                        ;*THIS IS A SPECIAL TEST WHICH IS ONLY EXECUTED  ONE TIME.
                                        ;*THE FIRST PASS AFTER THE DIAGNOSTIC IS LOADED. IT TYPES ON
                                        ;*THE CONSOLE THE PART NUMBERS OF THE CROMS WHICH THIS
                                        ;*REVISION SUPPORTS. TO FORCE A TYPE OUT PATCH LOCATION
                                        ;*ROMNUM: TO A ZERO.
                                        ;:************************************************************

                                        ;  TEST 1
                                        ;--------------
022324  012737  000001  001226  TST1:   MOV     #1,TSTNO
022332  012737  022762  001216          MOV     #TST2,NEXT
                                                                ;R1 CONTAINS BASE DMC11 ADDRESS
022340  005737  022362                  TST     ROMNUM          ;FIRST TIME HERE?
022344  001002                          BNE     1$              ;SKIP IF NOT
022346  104402  022364                  TYPE,   ROM1            ;TYPE PART NUMBERS
022352  012737  177777  022362  1$:     MOV     #-1,ROMNUM      ;SET FLAG TO ONLY TYPE ONCE
022360  104400                          SCOPE                   ;
022362  000000                  ROMNUM: 0
022364  005377  055103  046504  ROM1:   .ASCII  <377><12>/CZDMG-D SUPPORTS THE FOLLOWING CROM PART NUMBERS:/
  (1)   022447     377  042012  041515          .ASCII  <377><12>/DMC11-AR  (M8200-YA)/
  (1)   022475     377  031462  033055          .ASCII  <377>/23-630A9/<15><12>/23-631A9/<15><12>/23-632A9/
  (1)   022532  005015  031462  033055          .ASCII  <15><12>/23-632A9/<15><12>/23-634A9/<15><12>/23-635A9/
  (1)   022570  005015  031462  033055          .ASCII  <15><12>/23-636A9/<15><12>/23-637A9/
  (1)   022614  005377  046504  030503          .ASCII  <377><12>/DMC11-AL  (M8200-YB)/
  (1)   022642  031377  026463  031066          .ASCII  <377>/23-622A9/<15><12>/23-623A9/<15><12>/23-624A9/
  (1)   022677     015  031012  026463          .ASCII  <15><12>/23-625A9/<15><12>/23-626A9/<15><12>/23-627A9/
  (1)   022735     015  031012  026463          .ASCII  <15><12>/23-628A9/
  (1)   022747     015  031012  026463          .ASCIZ  <15><12>/23-629A9/
  (1)                                   .EVEN
  (3)

                                        ;*************************** TEST 2 ****************************
                                        ;*THIS IS A SPECIAL TEST WHICH WILL RUN ON A KMC (DMC WITH
                                        ;*WRITTABLE CONTROL STORE) TO LOAD THE CRAM WITH THE DDCMP
                                        ;*MICRO-CODE. FIRST BE SURE BIT1 OF STAT3 IS SET UP AS FOLLOWS
                                        ;*1=LOCAL HIGH SPEED CODE,  0=REMOTE LOW SPEED CODE  THE STATUS
                                        ;*OF STAT3 BIT1 DETERMINES WHICH MICRO-CODE WILL
                                        ;*BE LOADED IN THE KMC. LOOP ON THIS TEST FOR A FEW SECONDS
                                        ;*TO LOAD THE KMC.
                                        ;:************************************************************

                                        ;  TEST 2
                                        ;--------------
022762  012737  000002  001226  TST2:   MOV     #2,TSTNO
022770  012737  023054  001216          MOV     #TST3,NEXT
                                                                ;R1 CONTAINS BASE DMC11 ADDRESS
022776  004737  027014                  JSR     PC,MAPCK        ;CHECK FOR HI OR LO
023002  032737  100000  001366          BIT     #BIT15,STAT1    ;BE SURE DMC HAS CRAM
023010  001420                          BEQ     2$              ;SKIP IF NO CRAM
023012  005000                          CLR     R0              ;R0=CRAM ADDRESS
023014  013702  012322                  MOV     ROMMAP,R2       ;R2 POINTS TO ROMMAP
023020  012711  002000          1$:     MOV     #BIT10,(R1)     ;SET ROM0
```

```
1717  023024  010061  000004                  MOV     R0,4(R1)        ;LOAD CRAM ADDRESS
1718  023030  012261  000006                  MOV     (R2)+,6(R1)     ;LOAD WORD TO BE WRITTEN
1719  023034  052711  020000                  BIS     #BIT13,(R1)     ;WRITE IT!
1720  023040  005200                          INC     R0              ;NEXT ADDRESS
1721  023042  022700  002000                  CMP     #2000,R0        ;DONE YET?
1722  023046  001364                          BNE     1$              ;BR IF NO
1723  023050  005011                          CLR     (R1)            ;CLEAR SELO
1724  023052  104400                  2$:     SCOPE                   ;SCOPE THIS TEST
1725
1726
1727                                          ;*************************** TEST 3 ****************************
1728                                          ;*TEST OF BR RIGHT SHIFT
1729                                          ;*VERIFY THAT A DEST OF BR RSH (011) OF A MICRO-INSTRUCTION
1730                                          ;*SHIFTS THE RESULTING BR DATA RIGHT ONCE.
1731                                          ;:***********************************************************
1732
1733                                          ;  TEST 3
1734                                          ;--------------
1735  023054  012737  000003  001226  TST3:   MOV     #3,TSTNO
1736  023062  012737  023170  001216          MOV     #TST4,NEXT
1737                                                                  ;R1 CONTAINS BASE DMC11 ADDRESS
1738  023070  104412                          MSTCLR                  ;MASTER CLEAR DMC11
1739  023072  104412                          MSTCLR                  ;MASTER CLEAR DMC11
1740  023074  013701  001404                  MOV     DMCSR,R1        ;R1 = DMC BASE ADDRESS
1741  023100  005011                          CLR     (R1)            ;CLEAR SELO
1742  023102  012705  052525                  MOV     #52525,R5       ;START WITH 125
1743  023106  010561  000004                  MOV     R5,4(R1)        ;PORT4←125
1744  023112  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1745  023114  120500                          120500                  ;BR ← PORT4
1746  023116  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1747  023120  061620                          061620                  ;BR RSH←BR, SHIFT BR RIGHT
1748  023122  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1749  023124  061225                          061225                  ;PORT5←BR
1750  023126  006005                          ROR     R5              ;R5 = "EXPECTED"
1751  023130  116104  000005                  MOVB    5(R1),R4        ;R4 = "FOUND"
1752  023134  120504                          CMPB    R5,R4           ;DID BR SHIFT RIGHT ONCE?
1753  023136  001401                          BEQ     1$              ;BR IF YES
1754  023140  104012                          HLT     12              ;BR RIGHT SHIFT ERROR
1755  023142                          1$:
1756  023142  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1757  023144  061620                          061620                  ;BR RSH←BR, SHFT BR RIGHT AGAIN
1758  023146  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1759  023150  061225                          061225                  ;PORT5←BR
1760  023152  006005                          ROR     R5              ;R5 = "EXPECTED"
1761  023154  116104  000005                  MOVB    5(R1),R4        ;R4 = "FOUND"
1762  023160  120504                          CMPB    R5,R4           ;DID BR SHIFT RIGHT?
1763  023162  001401                          BEQ     2$              ;BR IF YES
1764  023164  104012                          HLT     12              ;BR RIGHT SHIFT ERROR
1765  023166  104400                  2$:     SCOPE                   ;SCOPE THIS TEST
1766
1767
1768                                          ;*************************** TEST 4 ****************************
1769                                          ;*CROM READ TEST
1770                                          ;*THIS TEST READS EACH ROM LOCATION AND COMPARES
1771                                          ;*IT TO A SOFTWARE DUPLICATE OF THE CROM. THIS TEST
1772                                          ;*ALSO TESTS THE JUMP(I) MICRO-PROCESSOR INSTRUCTION.
```

```
                                        ;*IF THIS TEST FAILS CHECK YOUR CROM PART NUMBERS.
                                        ;*CZDMG-D SUPPORTS THE FOLLOWING PART NUMBERS:
                                        ;*
                                        ;*DMC11-AR  (M8200-YA)
                                        ;*       23-630A9
                                        ;*       23-631A9
                                        ;*       23-632A9
                                        ;*       23-633A9
                                        ;*       23-634A9
                                        ;*       23-635A9
                                        ;*       23-636A9
                                        ;*       23-637A9
                                        ;*
                                        ;*DMC11-AL  (M8200-YB)
                                        ;*       23-622A9
                                        ;*       23-623A9
                                        ;*       23-624A9
                                        ;*       23-625A9
                                        ;*       23-626A9
                                        ;*       23-627A9
                                        ;*       23-628A9
                                        ;*       23-629A9
                                        ;:*****************************************************************

                                        ;  TEST 4
                                        ;--------------
023170  012737  000004  001226  TST4:   MOV     #4,TSTNO
023176  012737  023364  001216          MOV     #TST5,NEXT
023204  012737  023242  001220          MOV     #1$,LOCK
                                                                ;R1 CONTAINS BASE DMC11 ADDRESS
023212  104412                         MSTCLR                  ;MASTER CLEAR DMC11
023214  032737  100000  001366          BIT     #BIT15,STAT1    ;IS IT RAM OR ROM
023222  001057                         BNE     4$              ;SKIP TEST IF CRAM
023224  004737  027014                 JSR     PC,MAPCK        ;CHECK FOR HI OR LO
023230  005011                         CLR     (R1)            ;CLEAR RUN
023232  013700  012322                 MOV     ROMMAP,R0       ;R0 POINTS TO SOFTWARE ROM MAP
023236  005002                         CLR     R2              ;R2 CONTAINS ROM ADDRESS BITS 0-7
023240  005003                         CLR     R3              ;R3 CONTAINS ROM ADDRESS BITS 8&9 IN BITS 11&12
023242  042737  014377  023262  1$:     BIC     #14377,2$       ;CLEAR ADDRESS FIELDS OF INSTRUCTION
023250  050237  023262                 BIS     R2,2$           ;ADD BITS 0-7 TO INSTRUCTION
023254  050337  023262                 BIS     R3,2$           ;ADD BITS 11&12 TO INSTRUCTION
023260  104414                         ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
023262  100400                 2$:     100400                  ;JUMP(I) TO ROM ADDRESS IN R2 & R3
023264  012711  002000                 MOV     #BIT10,(R1)     ;SET ROMO
023270  011005                         MOV     (R0),R5         ;PUT "EXPECTED" IN R5
023272  016104  000006                 MOV     6(R1),R4        ;PUT "FOUND" IN R4
023276  020504                         CMP     R5,R4           ;COMPARE ROM CONTENTS TO SOFT DUP
023300  001414                         BEQ     3$              ;BR IF OK
023302  010337  001252                 MOV     R3,TEMP3        ;PUT ROM ADDRESS IN TEMP3
023306  000241                         CLC                     ;FOR ERROR TYPEOUT
023310  006037  001252                 ROR     TEMP3
023314  006037  001252                 ROR     TEMP3
023320  006037  001252                 ROR     TEMP3
023324  050237  001252                 BIS     R2,TEMP3        ;TEMP3 NOW CONTAINS CORRECT ADDRESS
023330  104004                         HLT     4               ;ROM READ ERROR
023332  104401                 3$:     SCOP1                   ;LOOP TO 1$ IF SW09=1
```

```
1829  023334  005720                          TST     (R0)+           ;BUMP SOFT POINTER
1830  023336  005202                          INC     R2              ;BUMP ROM ADDRESS
1831  023340  022702  000400                  CMP     #400,R2         ;IS R2 TO MAX YET?
1832  023344  001336                          BNE     1$              ;BR IF NO
1833  023346  005002                          CLR     R2              ;YES, RESET R2 TO 0
1834  023350  062703  004000                  ADD     #4000,R3        ;INC TO NEXT PAGE OF ROM
1835  023354  022703  020000                  CMP     #20000,R3       ;DONE YET?
1836  023360  001330                          BNE     1$              ;BR IF NO
1837  023362  104400                  4$:     SCOPE                   ;SCOPE THIS TEST
1838
1839
1840                                          ;*************************** TEST 5 ****************************
1841                                          ;*CROM TEST OF JUMP(I) NEVER MICRO-PROCESSOR INSTRUCTION.
1842                                          ;*PERFORM THE JUMP INSTRUCTION
1843                                          ;*VERIFY THAT THE JUMP DID NOT OCCUR BY READING
1844                                          ;*THE CONTENTS OF THE NEW ROM PC(IT SHOULD INCREMENT BY ONE).
1845                                          ;:***************************************************************
1846
1847                                          ;  TEST 5
1848                                          ;--------------
1849  023364  012737  000005  001226  TST5:   MOV     #5,TSTNO
1850  023372  012737  023560  001216          MOV     #TST6,NEXT
1851  023400  012737  023424  001220          MOV     #1$,LOCK
1852                                                                  ;R1 CONTAINS BASE DMC11 ADDRESS
1853  023406  104412                          MSTCLR                  ;MASTER CLEAR DMC11
1854  023410  032737  100000  001366          BIT     #BIT15,STAT1    ;IS IT CRAM?
1855  023416  001057                          BNE     6$+2            ;SKIP TEST IF YES
1856  023420  004737  027014                  JSR     PC,MAPCK        ;CHECK FOR HI OR LO
1857  023424                          1$:
1858  023424  004737  026660                  JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
1859  023430  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1860  023432  100400                          100400                  ;START AT ROM PC=0
1861  023434  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1862  023436  114377                          114377!<400*0>          ;JUMP TO ROM PC OF 1777
1863  023440  004737  026752                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
1864  023444  000002                          2                       ;INDEX
1865  023446  020504                          CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
1866  023450  001401                          BEQ     2$              ;BR IF YES
1867  023452  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
1868  023454  104401                  2$:     SCOP1                   ;LOOP TO 1$ IF SW09=1
1869  023456  012737  023464  001220          MOV     #3$,LOCK        ;NEW SCOP1
1870  023464                          3$:
1871  023464  004737  026660                  JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
1872  023470  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1873  023472  100403                          100403                  ;START AT ROM PC=3
1874  023474  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1875  023476  100000                          100000!<400*0>  ;JUMP TO ROM PC OF 0
1876  023500  004737  026752                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
1877  023504  000010                          10                      ;INDEX
1878  023506  020504                          CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
1879  023510  001401                          BEQ     4$              ;BR IF YES
1880  023512  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
1881  023514  104401                  4$:     SCOP1                   ;LOOP TO 3$ IF SW09=1
1882  023516  012737  023524  001220          MOV     #5$,LOCK        ;NEW SCOP1
1883  023524                          5$:
1884  023524  004737  026660                  JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
```

```
1885  023530  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1886  023532  100406                        100406                  ;START AT ROM PC=6
1887  023534  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1888  023536  104125                        104125!<400*0>          ;JUMP TO ROM PC OF 525
1889  023540  004737  026752                JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
1890  023544  000016                        16                      ;INDEX
1891  023546  020504                        CMP     R5,R4           ;ARE NEW ROM PC CONTENTS CORRECT?
1892  023550  001401                        BEQ     6$              ;BR IF YES
1893  023552  104006                        HLT     6               ;ERROR, CROM PC IS WRONG
1894  023554  104401              6$:       SCOP1                   ;LOOP TO 5$ IF SW59=1
1895  023556  104400                        SCOPE                   ;SCOPE THIS TEST
1896
1897
1898                                        ;*************************** TEST 6 ****************************
1899                                        ;*CROM TEST OF JUMP(I) ALWAYS MICRO-PROCESSOR INSTRUCTION.
1900                                        ;*PERFORM THE JUMP INSTRUCTION
1901                                        ;*VERIFY THE JUMP BY READING THE CONTENTS OF THE NEW ROM PC
1902                                        ;:***********************************************************
1903
1904                                        ;  TEST 6
1905                                        ;--------------
1906  023560  012737  000006  001226  TST6: MOV     #6,TSTNO
1907  023566  012737  023740  001216        MOV     #TST7,NEXT
1908  023574  012737  023620  001220        MOV     #1$,LOCK
1909                                                                ;R1 CONTAINS BASE DMC11 ADDRESS
1910  023602  104412                        MSTCLR                  ;MASTER CLEAR DMC11
1911  023604  032737  100000  001366        BIT     #BIT15,STAT1    ;IS IT CRAM?
1912  023612  001051                        BNE     6$+2            ;SKIP TEST IF YES
1913  023614  004737  027014                JSR     PC,MAPCK        ;CHECK FOR HI OR LO
1914  023620                        1$:
1915  023620  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1916  023622  100400                        100400                  ;START AT ROM PC=0
1917  023624  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1918  023626  114777                        114377!<400*1>          ;JUMP TO ROM PC OF 1777
1919  023630  004737  026752                JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
1920  023634  003776                        3776                    ;INDEX
1921  023636  020504                        CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
1922  023640  001401                        BEQ     2$              ;BR IF YES
1923  023642  104006                        HLT     6               ;ERROR, CROM PC IS WRONG
1924  023644  104401              2$:       SCOP1                   ;LOOP TO 1$ IF SW09=1
1925  023646  012737  023654  001220        MOV     #3$,LOCK        ;NEW SCOP1
1926  023654                        3$:
1927  023654  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1928  023656  100403                        100403                  ;START AT ROM PC=3
1929  023660  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1930  023662  100400                        100000!<400*1>  ;JUMP TO ROM PC OF 0
1931  023664  004737  026752                JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
1932  023670  000000                        0                       ;INDEX
1933  023672  020504                        CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
1934  023674  001401                        BEQ     4$              ;BR IF YES
1935  023676  104006                        HLT     6               ;ERROR, CROM PC IS WRONG
1936  023700  104401              4$:       SCOP1                   ;LOOP TO 3$ IF SW09=1
1937  023702  012737  023710  001220        MOV     #5$,LOCK        ;NEW SCOP1
1938  023710                        5$:
1939  023710  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1940  023712  100406                        100406                  ;START AT ROM PC=6
```

```
1941  023714  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1942  023716  104525                        104125!<400*1>          ;JUMP TO ROM PC OF 525
1943  023720  004737  026752                JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
1944  023724  001252                        1252                    ;INDEX
1945  023726  020504                        CMP     R5,R4           ;ARE NEW ROM PC CONTENTS CORRECT?
1946  023730  001401                        BEQ     6$              ;BR IF YES
1947  023732  104006                        HLT     6               ;ERROR, CROM PC IS WRONG
1948  023734  104401              6$:       SCOP1                   ;LOOP TO 5$ IF SW59=1
1949  023736  104400                        SCOPE                   ;SCOPE THIS TEST
1950
1951
1952                                        ;*************************** TEST 7 ****************************
1953                                        ;*CROM TEST OF JUMP(I) ON C BIT SET MICRO-PROCESSOR INSTRUCTION.
1954                                        ;*SET THE C BIT, PERFORM THE JUMP INSTRUCTION,
1955                                        ;*VERIFY THE JUMP BY READING THE CONTENTS OF THE NEW ROM PC
1956                                        ;:***********************************************************
1957
1958                                        ;  TEST 7
1959                                        ;--------------
1960  023740  012737  000007  001226  TST7: MOV     #7,TSTNO
1961  023746  012737  024134  001216        MOV     #TST10,NEXT
1962  023754  012737  024000  001220        MOV     #1$,LOCK
1963                                                                ;R1 CONTAINS BASE DMC11 ADDRESS
1964  023762  104412                        MSTCLR                  ;MASTER CLEAR DMC11
1965  023764  032737  100000  001366        BIT     #BIT15,STAT1    ;IS IT CRAM?
1966  023772  001057                        BNE     6$+2            ;SKIP TEST IF YES
1967  023774  004737  027014                JSR     PC,MAPCK        ;CHECK FOR HI OR LO
1968  024000                        1$:
1969  024000  004737  026726                JSR     PC,SETC ;SET THE C BIT'
1970  024004  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1971  024006  100400                        100400                  ;START AT ROM PC=0
1972  024010  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1973  024012  115377                        114377!<400*2>          ;JUMP TO ROM PC OF 1777
1974  024014  004737  026752                JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
1975  024020  003776                        3776                    ;INDEX
1976  024022  020504                        CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
1977  024024  001401                        BEQ     2$              ;BR IF YES
1978  024026  104006                        HLT     6               ;ERROR, CROM PC IS WRONG
1979  024030  104401              2$:       SCOP1                   ;LOOP TO 1$ IF SW09=1
1980  024032  012737  024040  001220        MOV     #3$,LOCK        ;NEW SCOP1
1981  024040                        3$:
1982  024040  004737  026726                JSR     PC,SETC ;SET THE C BIT'
1983  024044  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1984  024046  100403                        100403                  ;START AT ROM PC=3
1985  024050  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1986  024052  101000                        100000!<400*2>  ;JUMP TO ROM PC OF 0
1987  024054  004737  026752                JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
1988  024060  000000                        0                       ;INDEX
1989  024062  020504                        CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
1990  024064  001401                        BEQ     4$              ;BR IF YES
1991  024066  104006                        HLT     6               ;ERROR, CROM PC IS WRONG
1992  024070  104401              4$:       SCOP1                   ;LOOP TO 3$ IF SW09=1
1993  024072  012737  024100  001220        MOV     #5$,LOCK        ;NEW SCOP1
1994  024100                        5$:
1995  024100  004737  026726                JSR     PC,SETC ;SET THE C BIT'
1996  024104  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
```

```
1997  024106  100406                        100406                  ;START AT ROM PC=6
1998  024110  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1999  024112  105125                        104125!<400*2>          ;JUMP TO ROM PC OF 525
2000  024114  004737  026752                JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2001  024120  001252                        1252                    ;INDEX
2002  024122  020504                        CMP     R5,R4           ;ARE NEW ROM PC CONTENTS CORRECT?
2003  024124  001401                        BEQ     6$              ;BR IF YES
2004  024126  104006                        HLT     6               ;ERROR, CROM PC IS WRONG
2005  024130  104401              6$:       SCOP1                   ;LOOP TO 5$ IF SW59=1
2006  024132  104400                        SCOPE                   ;SCOPE THIS TEST
2007
2008
2009                                        ;*************************** TEST 10 ****************************
2010                                        ;*CROM TEST OF JUMP(I) ON Z BIT SET MICRO-PROCESSOR INSTRUCTION.
2011                                        ;*SET THE Z BIT, PERFORM THE JUMP INSTRUCTION,
2012                                        ;*VERIFY THE JUMP BY READING THE CONTENTS OF THE NEW ROM PC
2013                                        ;:************************************************************
2014
2015                                        ;  TEST 10
2016                                        ;--------------
2017  024134  012737  000010  001226  TST10: MOV     #10,TSTNO
2018  024142  012737  024330  001216        MOV     #TST11,NEXT
2019  024150  012737  024174  001220        MOV     #1$,LOCK
2020                                                                ;R1 CONTAINS BASE DMC11 ADDRESS
2021  024156  104412                        MSTCLR                  ;MASTER CLEAR DMC11
2022  024160  032737  100000  001366        BIT     #BIT15,STAT1    ;IS IT CRAM?
2023  024166  001057                        BNE     6$+2            ;SKIP TEST IF YES
2024  024170  004737  027014                JSR     PC,MAPCK        ;CHECK FOR HI OR LO
2025  024174                        1$:
2026  024174  004737  026744                JSR     PC,SETZ ;SET THE Z BIT'
2027  024200  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2028  024202  100400                        100400                  ;START AT ROM PC=0
2029  024204  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2030  024206  115777                        114377!<400*3>          ;JUMP TO ROM PC OF 1777
2031  024210  004737  026752                JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2032  024214  003776                        3776                    ;INDEX
2033  024216  020504                        CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2034  024220  001401                        BEQ     2$              ;BR IF YES
2035  024222  104006                        HLT     6               ;ERROR, CROM PC IS WRONG
2036  024224  104401              2$:       SCOP1                   ;LOOP TO 1$ IF SW09=1
2037  024226  012737  024234  001220        MOV     #3$,LOCK        ;NEW SCOP1
2038  024234                        3$:
2039  024234  004737  026744                JSR     PC,SETZ ;SET THE Z BIT'
2040  024240  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2041  024242  100403                        100403                  ;START AT ROM PC=3
2042  024244  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2043  024246  101400                        100000!<400*3>  ;JUMP TO ROM PC OF 0
2044  024250  004737  026752                JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2045  024254  000000                        0                       ;INDEX
2046  024256  020504                        CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2047  024260  001401                        BEQ     4$              ;BR IF YES
2048  024262  104006                        HLT     6               ;ERROR, CROM PC IS WRONG
2049  024264  104401              4$:       SCOP1                   ;LOOP TO 3$ IF SW09=1
2050  024266  012737  024274  001220        MOV     #5$,LOCK        ;NEW SCOP1
2051  024274                        5$:
2052  024274  004737  026744                JSR     PC,SETZ ;SET THE Z BIT'
```

```
2053  024300  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2054  024302  100406                        100406                  ;START AT ROM PC=6
2055  024304  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2056  024306  105525                        104125!<400*3>          ;JUMP TO ROM PC OF 525
2057  024310  004737  026752                JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2058  024314  001252                        1252                    ;INDEX
2059  024316  020504                        CMP     R5,R4           ;ARE NEW ROM PC CONTENTS CORRECT?
2060  024320  001401                        BEQ     6$              ;BR IF YES
2061  024322  104006                        HLT     6               ;ERROR, CROM PC IS WRONG
2062  024324  104401              6$:       SCOP1                   ;LOOP TO 5$ IF SW59=1
2063  024326  104400                        SCOPE                   ;SCOPE THIS TEST
2064
2065
2066                                        ;************************** TEST 11 **************************
2067                                        ;*CROM TEST OF JUMP(I) ON BRO SET MICRO-PROCESSOR INSTRUCTION.
2068                                        ;*SET THE BRO BIT, PERFORM THE JUMP INSTRUCTION,
2069                                        ;*VERIFY THE JUMP BY READING THE CONTENTS OF THE NEW ROM PC
2070                                        ;:**********************************************************************
2071
2072                                        ;  TEST 11
2073                                        ;--------------
2074  024330  012737  000011  001226  TST11: MOV     #11,TSTNO
2075  024336  012737  024524  001216        MOV     #TST12,NEXT
2076  024344  012737  024370  001220        MOV     #1$,LOCK
2077                                                                ;R1 CONTAINS BASE DMC11 ADDRESS
2078  024352  104412                        MSTCLR                  ;MASTER CLEAR DMC11
2079  024354  032737  100000  001366        BIT     #BIT15,STAT1    ;IS IT CRAM?
2080  024362  001057                        BNE     6$+2            ;SKIP TEST IF YES
2081  024364  004737  027014                JSR     PC,MAPCK        ;CHECK FOR HI OR LO
2082  024370                        1$:
2083  024370  004737  026676                JSR     PC,SETBRO       ;SET THE BRO BIT'
2084  024374  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2085  024376  100400                        100400                  ;START AT ROM PC=0
2086  024400  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2087  024402  116377                        114377!<400*4>          ;JUMP TO ROM PC OF 1777
2088  024404  004737  026752                JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2089  024410  003776                        3776                    ;INDEX
2090  024412  020504                        CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2091  024414  001401                        BEQ     2$              ;BR IF YES
2092  024416  104006                        HLT     6               ;ERROR, CROM PC IS WRONG
2093  024420  104401              2$:       SCOP1                   ;LOOP TO 1$ IF SW09=1
2094  024422  012737  024430  001220        MOV     #3$,LOCK        ;NEW SCOP1
2095  024430                        3$:
2096  024430  004737  026676                JSR     PC,SETBRO       ;SET THE BRO BIT'
2097  024434  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2098  024436  100403                        100403                  ;START AT ROM PC=3
2099  024440  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2100  024442  102000                        100000!<400*4>  ;JUMP TO ROM PC OF 0
2101  024444  004737  026752                JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2102  024450  000000                        0                       ;INDEX
2103  024452  020504                        CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2104  024454  001401                        BEQ     4$              ;BR IF YES
2105  024456  104006                        HLT     6               ;ERROR, CROM PC IS WRONG
2106  024460  104401              4$:       SCOP1                   ;LOOP TO 3$ IF SW09=1
2107  024462  012737  024470  001220        MOV     #5$,LOCK        ;NEW SCOP1
2108  024470                        5$:
```

```
2109  024470  004737  026676              JSR     PC,SETBRO       ;SET THE BRO BIT'
2110  024474  104414                      ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2111  024476  100406                      100406                  ;START AT ROM PC=6
2112  024500  104414                      ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2113  024502  106125                      104125!<400*4>          ;JUMP TO ROM PC OF 525
2114  024504  004737  026752              JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2115  024510  001252                      1252                    ;INDEX
2116  024512  020504                      CMP     R5,R4           ;ARE NEW ROM PC CONTENTS CORRECT?
2117  024514  001401                      BEQ     6$              ;BR IF YES
2118  024516  104006                      HLT     6               ;ERROR, CROM PC IS WRONG
2119  024520  104401              6$:     SCOP1                   ;LOOP TO 5$ IF SW59=1
2120  024522  104400                      SCOPE                   ;SCOPE THIS TEST
2121
2122
2123                                      ;************************** TEST 12 **************************
2124                                      ;*CROM TEST OF JUMP(I) ON BR1 SET MICRO-PROCESSOR INSTRUCTION.
2125                                      ;*SET THE BR1 BIT, PERFORM THE JUMP INSTRUCTION,
2126                                      ;*VERIFY THE JUMP BY READING THE CONTENTS OF THE NEW ROM PC
2127                                      ;:************************************************************
2128
2129                                      ;  TEST 12
2130                                      ;--------------
2131  024524  012737  000012  001226  TST12:  MOV     #12,TSTNO
2132  024532  012737  024720  001216          MOV     #TST13,NEXT
2133  024540  012737  024564  001220          MOV     #1$,LOCK
2134                                                          ;R1 CONTAINS BASE DMC11 ADDRESS
2135  024546  104412                      MSTCLR              ;MASTER CLEAR DMC11
2136  024550  032737  100000  001366      BIT     #BIT15,STAT1    ;IS IT CRAM?
2137  024556  001057                      BNE     6$+2            ;SKIP TEST IF YES
2138  024560  004737  027014              JSR     PC,MAPCK        ;CHECK FOR HI OR LO
2139  024564                      1$:
2140  024564  004737  026704              JSR     PC,SETBR1       ;SET THE BR1 BIT'
2141  024570  104414                      ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2142  024572  100400                      100400                  ;START AT ROM PC=0
2143  024574  104414                      ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2144  024576  116777                      114377!<400*5>          ;JUMP TO ROM PC OF 1777
2145  024600  004737  026752              JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2146  024604  003776                      3776                    ;INDEX
2147  024606  020504                      CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2148  024610  001401                      BEQ     2$              ;BR IF YES
2149  024612  104006                      HLT     6               ;ERROR, CROM PC IS WRONG
2150  024614  104401              2$:     SCOP1                   ;LOOP TO 1$ IF SW09=1
2151  024616  012737  024624  001220      MOV     #3$,LOCK        ;NEW SCOP1
2152  024624                      3$:
2153  024624  004737  026704              JSR     PC,SETBR1       ;SET THE BR1 BIT'
2154  024630  104414                      ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2155  024632  100403                      100403                  ;START AT ROM PC=3
2156  024634  104414                      ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2157  024636  102400                      100000!<400*5>  ;JUMP TO ROM PC OF 0
2158  024640  004737  026752              JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2159  024644  000000                      0                       ;INDEX
2160  024646  020504                      CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2161  024650  001401                      BEQ     4$              ;BR IF YES
2162  024652  104006                      HLT     6               ;ERROR, CROM PC IS WRONG
2163  024654  104401              4$:     SCOP1                   ;LOOP TO 3$ IF SW09=1
2164  024656  012737  024664  001220      MOV     #5$,LOCK        ;NEW SCOP1
```

```
2165  024664
2166  024664  004737  026704  5$:     JSR     PC,SETBR1       ;SET THE BR1 BIT'
2167  024670  104414                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2168  024672  100406                  100406                  ;START AT ROM PC=6
2169  024674  104414                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2170  024676  106525                  104125!<400*5>          ;JUMP TO ROM PC OF 525
2171  024700  004737  026752          JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2172  024704  001252                  1252                    ;INDEX
2173  024706  020504                  CMP     R5,R4           ;ARE NEW ROM PC CONTENTS CORRECT?
2174  024710  001401                  BEQ     6$              ;BR IF YES
2175  024712  104006                  HLT     6               ;ERROR, CROM PC IS WRONG
2176  024714  104401          6$:     SCOP1                   ;LOOP TO 5$ IF SW59=1
2177  024716  104400                  SCOPE                   ;SCOPE THIS TEST
2178
2179
2180                                  ;************************* TEST 13 *************************
2181                                  ;*CROM TEST OF JUMP(I) ON BR4 SET MICRO-PROCESSOR INSTRUCTION.
2182                                  ;*SET THE BR4 BIT, PERFORM THE JUMP INSTRUCTION,
2183                                  ;*VERIFY THE JUMP BY READING THE CONTENTS OF THE NEW ROM PC
2184                                  ;:*************************************************************
2185
2186                                  ;  TEST 13
2187                                  ;--------------
2188  024720  012737  000013  001226  TST13:  MOV     #13,TSTNO
2189  024726  012737  025114  001216          MOV     #TST14,NEXT
2190  024734  012737  024760  001220          MOV     #1$,LOCK
2191                                                          ;R1 CONTAINS BASE DMC11 ADDRESS
2192  024742  104412                  MSTCLR                  ;MASTER CLEAR DMC11
2193  024744  032737  100000  001366  BIT     #BIT15,STAT1    ;IS IT CRAM?
2194  024752  001057                  BNE     6$+2            ;SKIP TEST IF YES
2195  024754  004737  027014          JSR     PC,MAPCK        ;CHECK FOR HI OR LO
2196  024760                  1$:
2197  024760  004737  026712          JSR     PC,SETBR4       ;SET THE BR4 BIT'
2198  024764  104414                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2199  024766  100400                  100400                  ;START AT ROM PC=0
2200  024770  104414                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2201  024772  117377                  114377!<400*6>          ;JUMP TO ROM PC OF 1777
2202  024774  004737  026752          JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2203  025000  003776                  3776                    ;INDEX
2204  025002  020504                  CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2205  025004  001401                  BEQ     2$              ;BR IF YES
2206  025006  104006                  HLT     6               ;ERROR, CROM PC IS WRONG
2207  025010  104401          2$:     SCOP1                   ;LOOP TO 1$ IF SW09=1
2208  025012  012737  025020  001220  MOV     #3$,LOCK        ;NEW SCOP1
2209  025020                  3$:
2210  025020  004737  026712          JSR     PC,SETBR4       ;SET THE BR4 BIT'
2211  025024  104414                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2212  025026  100403                  100403                  ;START AT ROM PC=3
2213  025030  104414                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2214  025032  103000                  100000!<400*6>  ;JUMP TO ROM PC OF 0
2215  025034  004737  026752          JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2216  025040  000000                  0                       ;INDEX
2217  025042  020504                  CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2218  025044  001401                  BEQ     4$              ;BR IF YES
2219  025046  104006                  HLT     6               ;ERROR, CROM PC IS WRONG
2220  025050  104401          4$:     SCOP1                   ;LOOP TO 3$ IF SW09=1
```

CZDMGD.P11 05-JAN-78 11:08 CROM JUMP TESTS

```
2221  025052  012737  025060  001220          MOV     #5$,LOCK        ;NEW SCOP1
2222  025060                          5$:
2223  025060  004737  026712                  JSR     PC,SETBP4       ;SET THE BR4 BIT'
2224  025064  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2225  025066  100406                          100406                  ;START AT ROM PC=6
2226  025070  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2227  025072  107125                          104125!<400*6>          ;JUMP TO ROM PC OF 525
2228  025074  004737  026752                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2229  025100  001252                          1252                    ;INDEX
2230  025102  020504                          CMP     R5,R4           ;ARE NEW ROM PC CONTENTS CORRECT?
2231  025104  001401                          BEQ     6$              ;BR IF YES
2232  025106  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
2233  025110  104401                  6$:     SCOP1                   ;LOOP TO 5$ IF SW9=1
2234  025112  104400                          SCOPE                   ;SCOPE THIS TEST
2235
2236
2237                                          ;************************** TEST 14 **************************
2238                                          ;*CROM TEST OF JUMP(I) ON BR7 SET MICRO-PROCESSOR INSTRUCTION.
2239                                          ;*SET THE BR7 BIT, PERFORM THE JUMP INSTRUCTION,
2240                                          ;*VERIFY THE JUMP BY READING THE CONTENTS OF THE NEW ROM PC
2241                                          ;:************************************************************
2242
2243                                          ;  TEST 14
2244                                          ;--------------
2245  025114  012737  000014  001226  TST14:  MOV     #14,TSTNO
2246  025122  012737  025310  001216          MOV     #TST15,NEXT
2247  025130  012737  025154  001220          MOV     #1$,LOCK
2248                                                                  ;R1 CONTAINS BASE DMC11 ADDRESS
2249  025136  104412                          MSTCLR                  ;MASTER CLEAR DMC11
2250  025140  032737  100000  001366          BIT     #BIT15,STAT1    ;IS IT CRAM?
2251  025146  001057                          BNE     6$+2            ;SKIP TEST IF YES
2252  025150  004737  027014                  JSR     PC,MAPCK        ;CHECK FOR HI OR LO
2253  025154                          1$:
2254  025154  004737  026720                  JSR     PC,SETBR7       ;SET THE BR7 BIT'
2255  025160  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2256  025162  100400                          100400                  ;START AT ROM PC=0
2257  025164  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2258  025166  117777                          114377!<400*7>          ;JUMP TO ROM PC OF 1777
2259  025170  004737  026752                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2260  025174  003776                          3776                    ;INDEX
2261  025176  020504                          CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2262  025200  001401                          BEQ     2$              ;BR IF YES
2263  025202  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
2264  025204  104401                  2$:     SCOP1                   ;LOOP TO 1$ IF SW09=1
2265  025206  012737  025214  001220          MOV     #3$,LOCK        ;NEW SCOP1
2266  025214                          3$:
2267  025214  004737  026720                  JSR     PC,SETBR7       ;SET THE BR7 BIT'
2268  025220  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2269  025222  100403                          100403                  ;START AT ROM PC=3
2270  025224  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2271  025226  103400                          100000!<400*7>  ;JUMP TO ROM PC OF 0
2272  025230  004737  026752                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2273  025234  000000                          0                       ;INDEX
2274  025236  020504                          CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2275  025240  001401                          BEQ     4$              ;BR IF YES
2276  025242  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
```

```
2277  025244  104401                  4$:   SCOP1                   ;LOOP TO 3$ IF SW09=1
2278  025246  012737  025254  001220        MOV     #5$,LOCK        ;NEW SCOP1
2279  025254                          5$:
2280  025254  004737  026720                JSR     PC,SETBR7       ;SET THE BR7 BIT'
2281  025260  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2282  025262  100406                        100406                  ;START AT ROM PC=6
2283  025264  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2284  025266  107525                        104125!<400*7>          ;JUMP TO ROM PC OF 525
2285  025270  004737  026752                JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2286  025274  001252                        1252                    ;INDEX
2287  025276  020504                        CMP     R5,R4           ;ARE NEW ROM PC CONTENTS CORRECT?
2288  025300  001401                        BEQ     6$              ;BR IF YES
2289  025302  104006                        HLT     6               ;ERROR, CROM PC IS WRONG
2290  025304  104401                  6$:   SCOP1                   ;LOOP TO 5$ IF SW59=1
2291  025306  104400                        SCOPE                   ;SCOPE THIS TEST
2292
2293
2294                                        ;************************** TEST 15 **************************
2295                                        ;*CROM TEST OF JUMP(I) ON C BIT SET MICRO-PROCESSOR INSTRUCTION.
2296                                        ;*CLEAR THE C BIT, PERFORM THE JUMP INSTRUCTION,
2297                                        ;*VERIFY THAT THE JUMP DID NOT OCCUR BY READING
2298                                        ;*THE CONTENTS OF THE NEW ROM PC(IT SHOULD INCREMENT BY ONE).
2299                                        ;:************************************************************
2300
2301                                        ;  TEST 15
2302                                        ;--------------
2303  025310  012737  000015  001226  TST15: MOV     #15,TSTNO
2304  025316  012737  025504  001216        MOV     #TST16,NEXT
2305  025324  012737  025350  001220        MOV     #1$,LOCK
2306                                                                ;R1 CONTAINS BASE DMC11 ADDRESS
2307  025332  104412                        MSTCLR                  ;MASTER CLEAR DMC11
2308  025334  032737  100000  001366        BIT     #BIT15,STAT1    ;IS IT CRAM?
2309  025342  001057                        BNE     6$+2            ;SKIP TEST IF YES
2310  025344  004737  027014                JSR     PC,MAPCK        ;CHECK FOR HI OR LO
2311  025350                          1$:
2312  025350  004737  026660                JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2313  025354  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2314  025356  100400                        100400                  ;START AT ROM PC=0
2315  025360  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2316  025362  115377                        114377!<400*2>          ;JUMP TO ROM PC OF 1777
2317  025364  004737  026752                JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2318  025370  000002                        2                       ;INDEX
2319  025372  020504                        CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2320  025374  001401                        BEQ     2$              ;BR IF YES
2321  025376  104006                        HLT     6               ;ERROR, CROM PC IS WRONG
2322  025400  104401                  2$:   SCOP1                   ;LOOP TO 1$ IF SW09=1
2323  025402  012737  025410  001220        MOV     #3$,LOCK        ;NEW SCOP1
2324  025410                          3$:
2325  025410  004737  026660                JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2326  025414  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2327  025416  100403                        100403                  ;START AT ROM PC=3
2328  025420  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2329  025422  101000                        100000!<400*2>  ;JUMP TO ROM PC OF 0
2330  025424  004737  026752                JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2331  025430  000010                        10                      ;INDEX
2332  025432  020504                        CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
```

```
2333  025434  001401                        BEQ     4$              ;BR IF YES
2334  025436  104006                        HLT     6               ;ERROR, CROM PC IS WRONG
2335  025440  104401                4$:     SCOP1                   ;LOOP TO 3$ IF SW09=1
2336  025442  012737  025450  001220        MOV     #5$,LOCK        ;NEW SCOP1
2337  025450                        5$:
2338  025450  004737  026660                JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2339  025454  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2340  025456  100406                        100406                  ;START AT ROM PC=6
2341  025460  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2342  025462  105125                        104125!<400*2>          ;JUMP TO ROM PC OF 525
2343  025464  004737  026752                JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2344  025470  000016                        16                      ;INDEX
2345  025472  020504                        CMP     R5,R4           ;ARE NEW ROM PC CONTENTS CORRECT?
2346  025474  001401                        BEQ     6$              ;BR IF YES
2347  025476  104006                        HLT     6               ;ERROR, CROM PC IS WRONG
2348  025500  104401                6$:     SCOP1                   ;LOOP TO 5$ IF SW59=1
2349  025502  104400                        SCOPE                   ;SCOPE THIS TEST
2350
2351
2352                                        ;************************** TEST 16 **************************
2353                                        ;*CROM TEST OF JUMP(I) ON Z BIT SET MICRO-PROCESSOR INSTRUCTION.
2354                                        ;*CLEAR THE Z BIT, PERFORM THE JUMP INSTRUCTION,
2355                                        ;*VERIFY THAT THE JUMP DID NOT OCCUR BY READING
2356                                        ;*THE CONTENTS OF THE NEW ROM PC(IT SHOULD INCREMENT BY ONE).
2357                                        ;:**********************************************************************
2358
2359                                        ;  TEST 16
2360                                        ;--------------
2361  025504  012737  000016  001226  TST16: MOV     #16,TSTNO
2362  025512  012737  025700  001216        MOV     #TST17,NEXT
2363  025520  012737  025544  001220        MOV     #1$,LOCK
2364                                                                ;R1 CONTAINS BASE DMC11 ADDRESS
2365  025526  104412                        MSTCLR                  ;MASTER CLEAR DMC11
2366  025530  032737  100000  001366        BIT     #BIT15,STAT1    ;IS IT CRAM?
2367  025536  001057                        BNE     6$+2            ;SKIP TEST IF YES
2368  025540  004737  027014                JSR     PC,MAPCK        ;CHECK FOR HI OR LO
2369  025544                        1$:
2370  025544  004737  026660                JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2371  025550  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2372  025552  100400                        100400                  ;START AT ROM PC=0
2373  025554  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2374  025556  115777                        114377!<400*3>          ;JUMP TO ROM PC OF 1777
2375  025560  004737  026752                JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2376  025564  000002                        2                       ;INDEX
2377  025566  020504                        CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2378  025570  001401                        BEQ     2$              ;BR IF YES
2379  025572  104006                        HLT     6               ;ERROR, CROM PC IS WRONG
2380  025574  104401                2$:     SCOP1                   ;LOOP TO 1$ IF SW09=1
2381  025576  012737  025604  001220        MOV     #3$,LOCK        ;NEW SCOP1
2382  025604                        3$:
2383  025604  004737  026660                JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2384  025610  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2385  025612  100403                        100403                  ;START AT ROM PC=3
2386  025614  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2387  025616  101400                        100000!<400*3>  ;JUMP TO ROM PC OF 0
2388  025620  004737  026752                JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
```

```
2389  025624  000010                        10                      ;INDEX
2390  025626  020504                        CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2391  025630  001401                        BEQ     4$              ;BR IF YES
2392  025632  104006                        HLT     6               ;ERROR, CROM PC IS WRONG
2393  025634  104401              4$:       SCOP1                   ;LOOP TO 3$ IF SW09=1
2394  025636  012737  025644  001220        MOV     #5$,LOCK        ;NEW SCOP1
2395  025644                      5$:
2396  025644  004737  026660                JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2397  025650  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2398  025652  100406                        100406                  ;START AT ROM PC=6
2399  025654  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2400  025656  105525                        104125!<400*3>          ;JUMP TO ROM PC OF 525
2401  025660  004737  026752                JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2402  025664  000016                        16                      ;INDEX
2403  025666  020504                        CMP     R5,R4           ;ARE NEW ROM PC CONTENTS CORRECT?
2404  025670  001401                        BEQ     6$              ;BR IF YES
2405  025672  104006                        HLT     6               ;ERROR, CROM PC IS WRONG
2406  025674  104401              6$:       SCOP1                   ;LOOP TO 5$ IF SW59=1
2407  025676  104400                        SCOPE                   ;SCOPE THIS TEST
2408
2409
2410                                        ;************************** TEST 17 **************************
2411                                        ;*CROM TEST OF JUMP(I) ON BRO SET MICRO-PROCESSOR INSTRUCTION.
2412                                        ;*CLEAR THE BRO BIT, PERFORM THE JUMP INSTRUCTION,
2413                                        ;*VERIFY THAT THE JUMP DID NOT OCCUR BY READING
2414                                        ;*THE CONTENTS OF THE NEW ROM PC(IT SHOULD INCREMENT BY ONE).
2415                                        ;:**********************************************************
2416
2417                                        ;  TEST 17
2418                                        ;--------------
2419  025700  012737  000017  001226  TST17: MOV     #17,TSTNO
2420  025706  012737  026074  001216        MOV     #TST20,NEXT
2421  025714  012737  025740  001220        MOV     #1$,LOCK
2422                                                                ;R1 CONTAINS BASE DMC11 ADDRESS
2423  025722  104412                        MSTCLR                  ;MASTER CLEAR DMC11
2424  025724  032737  100000  001366        BIT     #BIT15,STAT1    ;IS IT CRAM?
2425  025732  001057                        BNE     6$+2            ;SKIP TEST IF YES
2426  025734  004737  027014                JSR     PC,MAPCK        ;CHECK FOR HI OR LO
2427  025740                      1$:
2428  025740  004737  026660                JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2429  025744  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2430  025746  100400                        100400                  ;START AT ROM PC=0
2431  025750  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2432  025752  116377                        114377!<400*4>          ;JUMP TO ROM PC OF 1777
2433  025754  004737  026752                JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2434  025760  000002                        2                       ;INDEX
2435  025762  020504                        CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2436  025764  001401                        BEQ     2$              ;BR IF YES
2437  025766  104006                        HLT     6               ;ERROR, CROM PC IS WRONG
2438  025770  104401              2$:       SCOP1                   ;LOOP TO 1$ IF SW09=1
2439  025772  012737  026000  001220        MOV     #3$,LOCK        ;NEW SCOP1
2440  026000                      3$:
2441  026000  004737  026660                JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2442  026004  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2443  026006  100403                        100403                  ;START AT ROM PC=3
2444  026010  104414                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
```

CZDMGD.P11 05-JAN-78 11:08 CROM JUMP TESTS

```
2445  026012  102000                    100000!<400*4>  ;JUMP TO ROM PC OF 0
2446  026014  004737  026752            JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2447  026020  000010                    10                      ;INDEX
2448  026022  020504                    CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2449  026024  001401                    BEQ     4$              ;BR IF YES
2450  026026  104006                    HLT     6               ;ERROR, CROM PC IS WRONG
2451  026030  104401            4$:     SCOP1                   ;LOOP TO 3$ IF SW09=1
2452  026032  012737  026040  001220    MOV     #5$,LOCK        ;NEW SCOP1
2453  026040                    5$:
2454  026040  004737  026660            JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2455  026044  104414                    ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2456  026046  100406                    100406                  ;START AT ROM PC=6
2457  026050  104414                    ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2458  026052  106125                    104125!<400*4>          ;JUMP TO ROM PC OF 525
2459  026054  004737  026752            JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2460  026060  000016                    16                      ;INDEX
2461  026062  020504                    CMP     R5,R4           ;ARE NEW ROM PC CONTENTS CORRECT?
2462  026064  001401                    BEQ     6$              ;BR IF YES
2463  026066  104006                    HLT     6               ;ERROR, CROM PC IS WRONG
2464  026070  104401            6$:     SCOP1                   ;LOOP TO 5$ IF SW59=1
2465  026072  104400                    SCOPE                   ;SCOPE THIS TEST
2466
2467
2468                                    ;*************************** TEST 20 ****************************
2469                                    ;*CROM TEST OF JUMP(I) ON BR1 SET MICRO-PROCESSOR INSTRUCTION.
2470                                    ;*CLEAR THE BR1 BIT, PERFORM THE JUMP INSTRUCTION,
2471                                    ;*VERIFY THAT THE JUMP DID NOT OCCUR BY READING
2472                                    ;*THE CONTENTS OF THE NEW ROM PC(IT SHOULD INCREMENT BY ONE).
2473                                    ;:************************************************************
2474
2475                                    ;  TEST 20
2476                                    ;--------------
2477  026074  012737  000020  001226    TST20:  MOV     #20,TSTNO
2478  026102  012737  026270  001216    MOV     #TST21,NEXT
2479  026110  012737  026134  001220    MOV     #1$,LOCK
2480                                                            ;R1 CONTAINS BASE DMC11 ADDRESS
2481  026116  104412                    MSTCLR                  ;MASTER CLEAR DMC11
2482  026120  032737  100000  001366    BIT     #BIT15,STAT1    ;IS IT CRAM?
2483  026126  001057                    BNE     6$+2            ;SKIP TEST IF YES
2484  026130  004737  027014            JSR     PC,MAPCK        ;CHECK FOR HI OR LO
2485  026134                    1$:
2486  026134  004737  026660            JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2487  026140  104414                    ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2488  026142  100400                    100400                  ;START AT ROM PC=0
2489  026144  104414                    ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2490  026146  116777                    114377!<400*5>          ;JUMP TO ROM PC OF 1777
2491  026150  004737  026752            JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2492  026154  000002                    2                       ;INDEX
2493  026156  020504                    CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2494  026160  001401                    BEQ     2$              ;BR IF YES
2495  026162  104006                    HLT     6               ;ERROR, CROM PC IS WRONG
2496  026164  104401            2$:     SCOP1                   ;LOOP TO 1$ IF SW09=1
2497  026166  012737  026174  001220    MOV     #3$,LOCK        ;NEW SCOP1
2498  026174                    3$:
2499  026174  004737  026660            JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2500  026200  104414                    ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
```

CZDMGD.P11 05-JAN-78 11:08 CROM JUMP TESTS

```
2501  026202  100403                          100403                  ;START AT ROM PC=3
2502  026204  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2503  026206  102400                          100000!<400*5>  ;JUMP TO ROM PC OF 0
2504  026210  004737  026752                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2505  026214  000010                          10                      ;INDEX
2506  026216  020504                          CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2507  026220  001401                          BEQ     4$              ;BR IF YES
2508  026222  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
2509  026224  104401                  4$:     SCOP1                   ;LOOP TO 3$ IF SW09=1
2510  026226  012737  026234  001220          MOV     #5$,LOCK        ;NEW SCOP1
2511  026234                          5$:
2512  026234  004737  026660                  JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2513  026240  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2514  026242  100406                          100406                  ;START AT ROM PC=6
2515  026244  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2516  026246  106525                          104125!<400*5>          ;JUMP TO ROM PC OF 525
2517  026250  004737  026752                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2518  026254  000016                          16                      ;INDEX
2519  026256  020504                          CMP     R5,R4           ;ARE NEW ROM PC CONTENTS CORRECT?
2520  026260  001401                          BEQ     6$              ;BR IF YES
2521  026262  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
2522  026264  104401                  6$:     SCOP1                   ;LOOP TO 5$ IF SW59=1
2523  026266  104400                          SCOPE                   ;SCOPE THIS TEST
2524
2525                                          ;************************** TEST 21 **************************
2526                                          ;*CROM TEST OF JUMP(I) ON BR4 SET MICRO-PROCESSOR INSTRUCTION.
2527                                          ;*CLEAR THE BR4 BIT, PERFORM THE JUMP INSTRUCTION,
2528                                          ;*VERIFY THAT THE JUMP DID NOT OCCUR BY READING
2529                                          ;*THE CONTENTS OF THE NEW ROM PC(IT SHOULD INCREMENT BY ONE).
2530                                          ;:************************************************************
2531
2532
2533                                          ;   TEST 21
2534                                          ;--------------
2535  026270  012737  000021  001226  TST21:  MOV     #21,TSTNO
2536  026276  012737  026464  001216          MOV     #TST22,NEXT
2537  026304  012737  026330  001220          MOV     #1$,LOCK
2538                                                                  ;R1 CONTAINS BASE DMC11 ADDRESS
2539  026312  104412                          MSTCLR                  ;MASTER CLEAR DMC11
2540  026314  032737  100000  001366          BIT     #BIT15,STAT1    ;IS IT CRAM?
2541  026322  001057                          BNE     6$+2            ;SKIP TEST IF YES
2542  026324  004737  027014                  JSR     PC,MAPCK        ;CHECK FOR HI OR LO
2543  026330                          1$:
2544  026330  004737  026660                  JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2545  026334  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2546  026336  100400                          100400                  ;START AT ROM PC=0
2547  026340  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2548  026342  117377                          114377!<400*6>          ;JUMP TO ROM PC OF 1777
2549  026344  004737  026752                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2550  026350  000002                          2                       ;INDEX
2551  026352  020504                          CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2552  026354  001401                          BEQ     2$              ;BR IF YES
2553  026356  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
2554  026360  104401                  2$:     SCOP1                   ;LOOP TO 1$ IF SW09=1
2555  026362  012737  026370  001220          MOV     #3$,LOCK        ;NEW SCOP1
2556  026370                          3$:
```

```
2557  026370  004737  026660          JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2558  026374  104414                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2559  026376  100403                  100403                  ;START AT ROM PC=3
2560  026400  104414                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2561  026402  103000                  100000!<400*6>  ;JUMP TO ROM PC OF 0
2562  026404  004737  026752          JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2563  026410  000010                  10                      ;INDEX
2564  026412  020504                  CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2565  026414  001401                  BEQ     4$              ;BR IF YES
2566  026416  104006                  HLT     6               ;ERROR, CROM PC IS WRONG
2567  026420  104401          4$:     SCOP1                   ;LOOP TO 3$ IF SW09=1
2568  026422  012737  026430  001220  MOV     #5$,LOCK        ;NEW SCOP1
2569  026430                  5$:
2570  026430  004737  026660          JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2571  026434  104414                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2572  026436  100406                  100406                  ;START AT ROM PC=6
2573  026440  104414                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2574  026442  107125                  104125!<400*6>          ;JUMP TO ROM PC OF 525
2575  026444  004737  026752          JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2576  026450  000016                  16                      ;INDEX
2577  026452  020504                  CMP     R5,R4           ;ARE NEW ROM PC CONTENTS CORRECT?
2578  026454  001401                  BEQ     6$              ;BR IF YES
2579  026456  104006                  HLT     6               ;ERROR, CROM PC IS WRONG
2580  026460  104401          6$:     SCOP1                   ;LOOP TO 5$ IF SW59=1
2581  026462  104400                  SCOPE                   ;SCOPE THIS TEST
2582
2583
2584                                  ;************************** TEST 22 **************************
2585                                  ;*CROM TEST OF JUMP(I) ON BR7 SET MICRO-PROCESSOR INSTRUCTION.
2586                                  ;*CLEAR THE BR7 BIT, PERFORM THE JUMP INSTRUCTION,
2587                                  ;*VERIFY THAT THE JUMP DID NOT OCCUR BY READING
2588                                  ;*THE CONTENTS OF THE NEW ROM PC(IT SHOULD INCREMENT BY ONE).
2589                                  ;;**********************************************************
2590
2591                                  ;  TEST 22
2592                                  ;--------------
2593  026464  012737  000022  001226  TST22:  MOV     #22,TSTNO
2594  026472  012737  003364  001216          MOV     #.EOP,NEXT
2595  026500  012737  026524  001220          MOV     #1$,LOCK
2596                                                          ;R1 CONTAINS BASE DMC11 ADDRESS
2597  026506  104412                  MSTCLR                  ;MASTER CLEAR DMC11
2598  026510  032737  100000  001366  BIT     #BIT15,STAT1    ;IS IT CRAM?
2599  026516  001057                  BNE     6$+2            ;SKIP TEST IF YES
2600  026520  004737  027014          JSR     PC,MAPCK        ;CHECK FOR HI OR LO
2601  026524                  1$:
2602  026524  004737  026660          JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2603  026530  104414                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2604  026532  100400                  100400                  ;START AT ROM PC=0
2605  026534  104414                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2606  026536  117777                  114377!<400*7>          ;JUMP TO ROM PC OF 1777
2607  026540  004737  026752          JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2608  026544  000002                  2                       ;INDEX
2609  026546  020504                  CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2610  026550  001401                  BEQ     2$              ;BR IF YES
2611  026552  104006                  HLT     6               ;ERROR, CROM PC IS WRONG
2612  026554  104401          2$:     SCOP1                   ;LOOP TO 1$ IF SW09=1
```

```
2613  026556  012737  026564  001220          MOV     #3$,LOCK        ;NEW SCOP1
2614  026564                          3$:
2615  026564  004737  026660                  JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2616  026570  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2617  026572  100403                          100403                  ;START AT ROM PC=3
2618  026574  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2619  026576  103400                          100000!<400*7>  ;JUMP TO ROM PC OF 0
2620  026600  004737  026752                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2621  026604  000010                          10                      ;INDEX
2622  026606  020504                          CMP     R5,R4           ;ARE NEW PC CONTENTS CORRECT?
2623  026610  001401                          BEQ     4$              ;BR IF YES
2624  026612  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
2625  026614  104401                  4$:     SCOP1                   ;LOOP TO 3$ IF SW09=1
2626  026616  012737  026624  001220          MOV     #5$,LOCK        ;NEW SCOP1
2627  026624                          5$:
2628  026624  004737  026660                  JSR     PC,CLRALL       ;CLEAR ALL  CONDITIONS
2629  026630  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2630  026632  100406                          100406                  ;START AT ROM PC=6
2631  026634  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2632  026636  107525                          104125!<400*7>          ;JUMP TO ROM PC OF 525
2633  026640  004737  026752                  JSR     PC,ROMDAT       ;R5=EXPECTED ROM DATA,R4=ACTUAL ROM DATA
2634  026644  000016                          16                      ;INDEX
2635  026646  020504                          CMP     R5,R4           ;ARE NEW ROM PC CONTENTS CORRECT?
2636  026650  001401                          BEQ     6$              ;BR IF YES
2637  026652  104006                          HLT     6               ;ERROR, CROM PC IS WRONG
2638  026654  104401                  6$:     SCOP1                   ;LOOP TO 5$ IF SW59=1
2639  026656  104400                          SCOPE                   ;SCOPE THIS TEST
2640
2641
2642                                  ;SUBROUTINES
2643                                  ;-----------
2644
2645  026660                          CLRALL:
2646                                  ;THIS SUBROUTINE CLEARS THE C&Z BITS AND THE BR
2647
2648  026660  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2649  026662  000400                          000400                  ;BR←0
2650  026664  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2651  026666  063220                          063220                  ;SP(0)←BR
2652  026670  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2653  026672  060400                          060400                  ;BR←SP(0)+BR
2654  026674  000207                          RTS     PC
2655
2656
2657  026676                          SETBR0:
2658                                  ;THIS SUBROUTINE SETS BR0 BIT
2659
2660  026676  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2661  026700  000401                          000401                  ;BR←001
2662  026702  000207                          RTS     PC
2663
2664
2665  026704                          SETBR1:
2666                                  ;THIS SUBROUTINE SETS BR1 BIT
2667
2668  026704  104414                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
```

```
2669  026706  000402                    000402                  ;BR←002
2670  026710  000207                    RTS     PC
2671
2672
2673  026712                    SETBR4:
2674                                    ;THIS SUBROUTINE SETS BR4 BIT
2675
2676  026712  104414                    ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2677  026714  000420                    000420                  ;BR←020
2678  026716  000207                    RTS     PC
2679
2680
2681  026720                    SETBR7:
2682                                    ;THIS SUBROUTINE SETS BR7 BIT
2683
2684  026720  104414                    ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2685  026722  000600                    000600                  ;BR←200
2686  026724  000207                    RTS     PC
2687
2688
2689  026726                    SETC:
2690                                    ;THIS SUBROUTINE SETS THE C BIT
2691
2692  026726  104414                    ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2693  026730  000777                    000777                  ;BR←377
2694  026732  104414                    ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2695  026734  063220                    063220                  ;SP(0)←BR
2696  026736  104414                    ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2697  026740  060400                    060400                  ;BR←SP(0)+BR
2698  026742  000207                    RTS     PC
2699
2700
2701  026744                    SETZ:
2702                                    ;THIS SUBROUTINE SETS THE Z BIT
2703
2704  026744  104414                    ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2705  026746  000777                    000777                  ;BR←377
2706  026750  000207                    RTS     PC
2707
2708
2709  026752                    ROMDAT:
2710                                    ;THIS SUBROUTINE LOADS R5 WITH EXPECTED ROM CONTENTS
2711                                    ;AND LOADS R4 WITH ACTUAL ROM CONTENTS
2712
2713  026752  017600  000000           MOV     @(SP),R0        ;INDEX FOR COMPARE
2714  026756  062716  000002           ADD     #2,(SP)         ;ADJUST STACK
2715  026762  012711  002000           MOV     #BIT10,(R1)     ;SET ROM0
2716  026766  016005  012324           MOV     LOMAP(R0),R5    ;PUT EXPECTED IN R5 (LOSPEED)
2717  026772  032737  000002  001372   BIT     #BIT1,STAT3     ;LOW OR HIGH SPEED?
2718  027000  001402                    BEQ     1$              ;BR IF LOW SPEED
2719  027002  016005  016324           MOV     HIMAP(R0),R5    ;PUT EXPECTED IN R5 (HISPEED)
2720  027006  016104  000006    1$:    MOV     6(R1),R4        ;PUT "FOUND" IN R4
2721  027012  000207                    RTS     PC              ;RETURN
2722
2723  027014                    MAPCK:
2724                                    ;THIS SUBROUTINE CHECKS THE STATUS TABLE AND LOADS
```

```
2725                                            ;THE ROMMAP POINTER TO POINT TO EITHER THE HIGH OR
2726                                            ;LOW SPEED MICRO-CODE.
2727
2728  027014  012737  012324  012322            MOV     #LOMAP,ROMMAP   ;LOAD POINTER TO LOW SPEED
2729  027022  032737  000002  001372            BIT     #BIT1,STAT3     ;CHECK STATUS TABLE
2730  027030  001403                            BEQ     1$              ;BR IF LOW SPEED
2731  027032  012737  016324  012322            MOV     #HIMAP,ROMMAP   ;LOAD POINTER TO HIGH SPEED
2732  027040  000207                    1$:     RTS     PC              ;RETURN
2733
2734
        027042  041777  040522  020115  EM1:    .ASCIZ  <377>/CRAM DATA ERROR/
        027063     377  051103  046501  EM2:    .ASCIZ  <377>/CRAM DUAL ADDRESSING ERROR/
        027117     377  051103  046517  EM3:    .ASCIZ  <377>/CROM DATA ERROR/
        027140  045377  046525  020120  EM4:    .ASCIZ  <377>/JUMP ERROR/
        027154  047777  052104  042440  EM5:    .ASCIZ  <377>/ODT ERROR IN IBUS* REG10/
        027206  044777  050117  046440  EM7:    .ASCIZ  <377>/ICP MAR TEST/
        027224  041377  020122  044522  EM10:   .ASCIZ  <377>/BR RIGHT SHIFT TEST/
        027251     377  042522  042503  EM11:   .ASCIZ  <377>/RECEIVE DATA ERROR/
        027275     377  051106  042505  EM12:   .ASCIZ  <377>/FREE RUNNING ERROR/
        027321     377  047503  052116  EM13:   .ASCIZ  <377>/CONTROL OUT ERROR/

        027344  042777  050130  041505  DH1:    .ASCIZ  <377>/EXPECTED  FOUND  ADDRESS/
        027376  042777  050130  041505  DH2:    .ASCIZ  <377>/EXPECTED  FOUND/
        027417     377  051440  046105  DH3:    .ASCIZ  <377>/ SEL4      SEL6/
                                            .EVEN

        027440  000003                    DT1:    3
        027442     006     004                    .BYTE   6,4
        027444  001264                            $AVR2
        027446     006     004                    .BYTE   6,4
        027450  001270                            $AVR4
        027452     004     002                    .BYTE   4,2
        027454  001260                            $AVR0
        027456  000003                    DT2:    3
        027460     006     004                    .BYTE   6,4
        027462  001272                            $AVR5
        027464     006     004                    .BYTE   6,4
        027466  001270                            $AVR4
        027470     004     002                    .BYTE   4,2
        027472  001264                            $AVR2
        027474  000003                    DT3:    3
        027476     006     004                    .BYTE   6,4
        027500  001272                            $AVR5
        027502     006     004                    .BYTE   6,4
        027504  001270                            $AVR4
        027506     004     002                    .BYTE   4,2
        027510  001252                            TEMP3
        027512  000002                    DT4:    2
        027514     003     007                    .BYTE   3,7
        027516  001272                            $AVR5
        027520     003     002                    .BYTE   3,2
        027522  001270                            $AVR4
        027524  000002                    DT5:    2
        027526     006     004                    .BYTE   6,4
        027530  001272                            $AVR5
        027532     006     002                    .BYTE   6,2
```

```
027534  001270                      SAVR4
027536  000003              DT7:    3
027540     003      010             .BYTE   3,10
027542  001272                      SAVR5
027544     003      004             .BYTE   3,4
027546  001270                      SAVR4
027550     004      002             .BYTE   4,2
027552  001264                      SAVR2
027554  000003              DT10:   3
027556     003      007             .BYTE   3,7
027560  001272                      SAVR5
027562     003      004             .BYTE   3,4
027564  001270                      SAVR4
027566     006      002             .BYTE   6,2
027570  001252                      TEMP3
027572  000002              DT11:   2
027574     006      004             .BYTE   6,4
027576  001252                      TEMP3
027600     006      002             .BYTE   6,2
027602  001254                      TEMP4

027604                      .ERRTAB:
027604  000000                      0
027606  000000                      0
027610  000000                      0
027612  027042              EM1
027614  027344              DH1     ;HLT    1
027616  027440              DT1
027620  027063              EM2
027622  027344              DH1     ;HLT    2
027624  027440              DT1
027626  027042              EM1
027630  027344              DH1     ;HLT    3
027632  027456              DT2
027634  027117              EM3
027636  027344              DH1     ;HLT    4
027640  027474              DT3
027642  027140              EM4
027644  027376              DH2     ;HLT    5
027646  027512              DT4
027650  027140              EM4
027652  027376              DH2     ;HLT    6
027654  027524              DT5
027656  027154              EM5
027660  027376              DH2     ;HLT    7
027662  027512              DT4
027664  000000              0
027666  000000              0
027670  000000              0
027672  027206              EM7
027674  027344              DH1     ;HLT    11
027676  027536              DT7
027700  027224              EM10
027702  027376              DH2     ;HLT    12
027704  027512              DT4
027706  027251              EM11
```

```
027710  027344          DH1     ;HLT    13
027712  027554          DT10
027714  027275          EM12
027716  000000          0       ;HLT    14
027720  000000          0
027722  027275          EM12
027724  027376          DH2     ;HLT    15
027726  027524          DT5
027730  027321          EM13
027732  027417          DH3     ;HLT    16
027734  027572          DT11

027736                  CORMAX:
        000001          .END
```

CZDMGD.P11 05-JAN-78 11:08 CROSS REFERENCE TABLE -- USER SYMBOLS

```
ADRCNT= 004373   879*   915*   924#
AUDONE  003024   569    608    613    659#
AUSTRT  002446   568#   663
AUTO.S  010514   526   1368#
BINWRD  004714   965*   968*   969   1006#
BIT0   = 000001    95#  1155   1156
BIT1   = 000002    94#   531   1149   1155   1156   1449   1532   1545   2717   2729
BIT10  = 002000    85#  1513   1524   1716   1816   2715
BIT11  = 004000    84#
BIT12  = 010000    83#  1464   1543
BIT13  = 020000    82#  1467   1516   1547   1719
BIT14  = 040000    81#   781   1478   1480   1547   1558
BIT15  = 100000    80#   485    572    575    596    599   1451   1519   1712   1804   1854   1911   1965
                  2022  2079   2136   2193   2250   2308   2366   2424   2482   2540   2598
BIT2   = 000004    93#   531    712   1156
BIT3   = 000010    92#  1549   1556
BIT4   = 000020    91#  1139   1162   1164
BIT5   = 000040    90#  1661
BIT6   = 000100    89#  1144   1145   1551
BIT7   = 000200    88#  1145   1265   1661
BIT8   = 000400    87#  1523   1538   1540   1553   1555   1561   1564   1622
BIT9   = 001000    86#  1521   1523   1535   1561   1564   1620   1622
BM      007055  1187#  1492
BRLVL   012254  1617   1627   1635   1647#
BRW     003730   718    804#
BRX     003732   719    805#
CHRCNT  004712   963*   966    970    986*  1004#  1005
CKSWR   007610   512    779    811   1026   1212#
CKSWR1  007670  1225#  1237
CKSWR2  007702  1228#
CKSWR3  007706  1230#
CKSWR4  007712  1231#  1239   1246
CKSWR5  010016  1213   1220   1255#
CLKX    001242   171#
CLRALL  026660  1858   1871   1884   2312   2325   2338   2370   2383   2396   2428   2441   2454   2486
                2499   2512   2544   2557   2570   2602   2615   2628   2645#
CNERR   007300   626   1187#
CNT.MA  001702   196    364#   484    486    488   1293
CNVRT  = 104411   233#   623    738    740    742    744   1060   1062   1122   1228
CONERR  007224   621   1187#
CONN    007115  1187#  1469
CONTAB  002776   630    643#
CONVRT= 104410   231#   543    629   1076   1391
CORMAX  027736  2734#
CRAM    006607  1187#  1430
CREAM   001320   195#   483*  1289*  1290   1292*  1296
CSR     006511  1187#  1395
CSRMAP  010516  1370#
CYCLE   010062   720    761    762   1280#
DATABP  005216  1049*  1052   1074   1077#
DATACL= 104415   241#
DATAHD  005204  1048*  1070   1073#
DELAY  = 104413   237#
DEVADR  004370   877*   912    922#
DEVTAB  003010   583    648#
DH1     027344  2734#
```

```
DH2     027376          2734#
DH3     027417          2734#
DISPLA  001200           142#   498*   504*   735*
DISPRE  000174           128#   504
DMACTV  001306           189#   521    676*   677    1280   1294   1376*  1577*  1583*  1584*  1588   1613
DMCM    007321           634   1187#
DMCR00  001500           279#
DMCR01  001510           284#
DMCR02  001520           289#
DMCR03  001530           294#
DMCR04  001540           299#
DMCR05  001550           304#
DMCR06  001560           309#
DMCR07  001570           314#
DMCR10  001600           319#
DMCR11  001610           324#
DMCR12  001620           329#
DMCR13  001630           334#
DMCR14  001640           339#
DMCR15  001650           344#
DMCR16  001660           349#
DMCR17  001670           354#
DMCSR   001404           262#   568*   602    607*   612*   647    765    802    1094   1117   1163   1298*  1307
                        1356   1740
DMCSRH  001406           263#  1144*  1145*  1149*  1155*  1156*  1162*  1164*  1307*  1308*  1309
DMCTL   001410           264#  1309*  1310*  1311
DMNUM   001310           190#   479    751   1374*  1389*  1568*  1569   1578   1580
DMP04   001412           265#  1133*  1139   1176   1181   1311*  1312*  1313
DMP06   001414           266#  1150*  1313*  1314*
DMRLVL  001376           259#  1316*  1317*  1318
DMRVEC  001374           258#   768   1299*  1300*  1316
DMS100  001502           280#
DMS101  001512           285#
DMS102  001522           290#
DMS103  001532           295#
DMS104  001542           300#
DMS105  001552           305#
DMS106  001562           310#
DMS107  001572           315#
DMS110  001602           320#
DMS111  001612           325#
DMS112  001622           330#
DMS113  001632           335#
DMS114  001642           340#
DMS115  001652           345#
DMS116  001662           350#
DMS117  001672           355#
DMS200  001504           281#
DMS201  001514           286#
DMS202  001524           291#
DMS203  001534           296#
DMS204  001544           301#
DMS205  001554           306#
DMS206  001564           311#
DMS207  001574           316#
DMS210  001604           321#
```

F04


```
DMS211  001614          326#
DMS212  001624          331#
DMS213  001634          336#
DMS214  001644          341#
DMS215  001654          346#
DMS216  001664          351#
DMS217  001674          356#
DMS300  001506          282#
DMS301  001516          287#
DMS302  001526          292#
DMS303  001536          297#
DMS304  001546          302#
DMS305  001556          307#
DMS306  001566          312#
DMS307  001576          317#
DMS310  001606          322#
DMS311  001616          327#
DMS312  001626          332#
DMS313  001636          337#
DMS314  001646          342#
DMS315  001656          347#
DMS316  001666          352#
DMS317  001676          357#
DMTLVL  001402          261#  1320*  1321*
DMTVEC  001400          260#  1318*  1319*  1320
DM.END  001700          359#  1372
DM.MAP  001500          195    278#   483    536    546    662   1290   1292   1370   1375   1605
DONE    003734          784    786*   806#  1216*
DT1     027440         2734#
DT10    027554         2734#
DT11    027572         2734#
DT2     027456         2734#
DT3     027474         2734#
DT4     027512         2734#
DT5     027524         2734#
DT7     027536         2734#
EM1     027042         2734#
EM10    027224         2734#
EM11    027251         2734#
EM12    027275         2734#
EM13    027321         2734#
EM2     027063         2734#
EM3     027117         2734#
EM4     027140         2734#
EM5     027154         2734#
EM7     027206         2734#
ERCT00  001704          366#
ERCT01  001710          369#
ERCT02  001714          372#
ERCT03  001720          375#
ERCT04  001724          378#
ERCT05  001730          381#
ERCT06  001734          384#
ERCT07  001740          387#
ERCT10  001744          390#
ERCT11  001750          393#
```

CZDMGD.P11 05-JAN-78 11:08 CROSS REFERENCE TABLE -- USER SYMBOLS

```
ERCT12  001754   396#
ERCT13  001760   399#
ERCT14  001764   402#
ERCT15  001770   405#
ERCT16  001774   408#
ERCT17  002000   411#
ERR     002700   592    618#   622
ERRCNT  001232   163#   747    774    1087*  1305*
ERRFLG  001325   202#   481*   733*   795*   1037*  1050   1064*  1123*
ERRMSG  005172  1047*  1065   1068#
ERRPC   002770   624    640#
ERTABO  005322  1062   1097#
EXIT  = 000205    96#
EXITER  005252  1082   1087#
FLOAT   002536   585#   591
FY      002566   594#   605    609    615
HALTS   005222  1033   1079#
HILIM   004366   876*   903    921#
HIMAP   016324  1670#  2719   2731
ICOUNT  001222   159#   793    798*
INBUF   007504   846    882   1202#
INCHAR  010022  1231   1259#
INIFLG  001324   201#   507    527    534*
INSTER= 104404   223#   897
INSTR = 104403   221#  1329   1381   1394   1403   1482   1491
INSTR2  004166   853    865#
INTTY   012270  1414   1431   1441   1454   1470   1655#
KMCM    007331   637   1187#
LIMITS  004314   892    903#
LINE    007017  1187#  1483
LOBITS  004372   878#   907    923#   924
LOCK    001220   158#   797*   814    816    1056   1801*  1851*  1869*  1882*  1908*  1925*  1937*  1962*
                1980*  1993*  2019*  2037*  2050*  2076*  2094*  2107*  2133*  2151*  2164*  2190*  2208*
                2221*  2247*  2265*  2278*  2305*  2323*  2336*  2363*  2381*  2394*  2421*  2439*  2452*
                2479*  2497*  2510*  2537*  2555*  2568*  2595*  2613*  2626*
LOKFLG  001326   203#
LOLIM   004364   875*   905    920#
LOMAP   012324  1667#  2716   2728
LPCNT   001224   160#   792*   793    796*
LSTERR  001234   164#   490*   732*   1034   1036*  1124*
MAPCK   027014  1711   1806   1856   1913   1967   2024   2081   2138   2195   2252   2310   2368   2426
                2484   2542   2600   2723#
MASKX   001244   172#
MASTEK  006143  1058   1187#
MCRLF   005672   831    954    1054   1055   1063   1187#  1328   1390
MCSRX   006073   737   1187#
MDATA   007546   984    994    1204#
MEMLIM  001304   188#   700*
MEPASS  005733   736   1187#
MERRPC  006220  1061   1187#
MERRX   006120   743   1187#
MERR2   005761  1187#  1590
MERR3   006006   673   1187#
MILK    001322   196#   484*   745    1288*  1293*  1297
MLOCK   006044   714   1187#
MNEW    006145   668   1187#
```

CZDMGD.P11 05-JAN-78 11:08 CROSS REFERENCE TABLE -- USER SYMBOLS

```
MODU    006705          1187*   1453
MPASSX  006107           741    1187*
MPFAIL  005675          1121    1187*
MQM     005666           861    1187*   1352    1427    1437    1447    1462    1476
MR      005756           723    1187*   1346
MRESET= 004000            96*
MSTCLR= 104412           235*   1126    1738    1739    1803    1853    1910    1964    2021    2078    2135    2192    2249
                        2307    2365    2423    2481    2539    2597
MTITLE  001000           136*    511
MTSTN   006131          1059    1187*   1330
MTSTPC  006032          1187*
MVECX   006101           739    1187*
NEXT    001216           157*    799    1092    1686*   1709*   1736*   1800*   1850*   1907*   1961*   2018*   2075*   2132*
                        2189*   2246*   2304*   2362*   2420*   2478*   2536*   2594*
NOACT   007155           523    1187*   1282
NODEV   002674           577     616*
NUM     006451          1187*   1382
OK      002646           603     610*    639
ONE     001302           187*
PACT00  001702           365*
PACT01  001706           368*
PACT02  001712           371*
PACT03  001716           374*
PACT04  001722           377*
PACT05  001726           380*
PACT06  001732           383*
PACT07  001736           386*
PACT10  001742           389*
PACT11  001746           392*
PACT12  001752           395*
PACT13  001756           398*
PACT14  001762           401*
PACT15  001766           404*
PACT16  001772           407*
PACT17  001776           410*
PARAM = 104405           225*   1331    1383    1396    1405    1484    1493
PARAM1  004234           881*    898
PARBIT= 040000            96*
PARERR  004310           884     886     888     897*    904     906     908
PASCNT  001230           162*    734*    735     746     771    1304*
PERFOR= 004537            96*
PFTAB   005430          1122    1128*
POPRO = 012600            72*   1086
POP1SP= 005726            70*
POP2SP= 022626            74*    801
PRIO    006550          1187*   1413
PS      177776            63*    476*    711*   1617*   1627*
PUSHRO= 010046            71*   1083
PUSH1S= 005746            69*
PUSH2S= 024646            73*
QV.FLG  001327           204*    482*    750*    790
RESREG  005220          1075    1078*
RESTAR  005350          1108    1114*
RESTRT  003534           749     753    761*
RESOS = 104407           229*   1078
RETURN  001214           156*    492*    720*    724     761*    799*    803    1092*   1095    1127    1345*   1355*   1357
```

```
ROMCLK= 104414          239*  1134  1137  1174  1179  1744  1746  1748  1756  1758  1814  1859  1861
                       1872  1874  1885  1887  1915  1917  1927  1929  1939  1941  1970  1972  1983
                       1985  1996  1998  2027  2029  2040  2042  2053  2055  2084  2086  2097  2099
                       2110  2112  2141  2143  2154  2156  2167  2169  2198  2200  2211  2213  2224
                       2226  2255  2257  2268  2270  2281  2283  2313  2315  2326  2328  2339  2341
                       2371  2373  2384  2386  2397  2399  2429  2431  2442  2444  2455  2457  2487
                       2489  2500  2502  2513  2515  2545  2547  2558  2560  2571  2573  2603  2605
                       2616  2618  2629  2631  2648  2650  2652  2660  2668  2676  2684  2692  2694
                       2696  2704
ROMDAT  026752         1863  1876  1889  1919  1931  1943  1974  1987  2000  2031  2044  2057  2088
                       2101  2114  2145  2158  2171  2202  2215  2228  2259  2272  2285  2317  2330
                       2343  2375  2388  2401  2433  2446  2459  2491  2504  2517  2549  2562  2575
                       2607  2620  2633  2709#
ROMMAP  012322         1665#  1715  1808  2728*  2731*
ROMNUM  022362         1688  1691*  1693#
ROM1    022364         1690  1694#
RUN     001316          193#   485*  1286*  1287*  1294
SAVACT  001312          191#   671  1588*
SAVNUM  001314          192#   479*   748*   751*  1581*
SAVPC   001276          185#   622*   642   929*  1099
SAVR0   001260          178#   938*   943  2734
SAVR1   001262          179#   628*   937*   944
SAVR2   001264          180#   936*   945  2734
SAVR3   001266          181#   935*   946
SAVR4   001270          182#   934*   947  2734
SAVR5   001272          183#   933*   948  2734
SAVSP   001274          184#
SAVOS = 104406          227#  1038
SCOPE = 104400          215#  1692  1724  1765  1837  1895  1949  2006  2063  2120  2177  2234  2291
                       2349  2407  2465  2523  2581  2639
SCOP1 = 104401          217#  1828  1868  1881  1894  1924  1936  1948  1979  1992  2005  2036  2049
                       2062  2093  2106  2119  2150  2163  2176  2207  2220  2233  2264  2277  2290
                       2322  2335  2348  2380  2393  2406  2438  2451  2464  2496  2509  2522  2554
                       2567  2580  2612  2625  2638
SETBR0  026676         2083  2096  2109  2657#
SETBR1  026704         2140  2153  2166  2665#
SETBR4  026712         2197  2210  2223  2673#
SETBR7  026720         2254  2267  2280  2681#
SETC    026726         1969  1982  1995  2689#
SETZ    026744         2026  2039  2052  2701#
SKIP    002632          573   576   597   600   606#
SOFTSW  010054         1229  1268#
SPACNT= 004713          964*   988   991*  1005#
SPEED   007341         1187#  1440
STACK = 001200           64#   477   690  1093  1116
STAT    001240          170#
STAT1   001366          251#  1301*  1712  1804  1854  1911  1965  2022  2079  2136  2193  2250  2308
                       2366  2424  2482  2540  2598
STAT2   001370          252#  1302*
STAT3   001372          253#  1303*  2717  2729
STRTSW  001236          169#   513*   516*   517   519   529   531   666   712   721  1323  1347*  1377
                       1601
SVOS    004402          933#
SWFLG   010020          480*   825  1224*  1251*  1257#
SWMES   007206         1187#  1227
SWMES1  007216         1187#  1230
```

```
SWR     001202  143#  497*  499   503*  513   671   676   781   788   812   827   1027  1032
                1081  1088  1090  1152  1212  1250*
SWREG   000176  129#  503   1212  1270
SW00  = 000001   45#  517   1377  1601
SW01  = 000002   44#  721   1323  1347
SW02  = 000004   43#
SW03  = 000010   42#  666
SW04  = 000020   41#
SW05  = 000040   40#
SW06  = 000100   39#  1152
SW07  = 000200   38#
SW08  = 000400   37#  1088
SW09  = 001000   36#  812
SW10  = 002000   35#  1090
SW11  = 004000   34#  788
SW12  = 010000   33#  827   1027
SW13  = 020000   32#  1032
SW14  = 040000   31#
SW15  = 100000   30#
TEMP    001416  271#  971   1118* 1119* 1160* 1165* 1171* 1183*
TEMP1   001246  173#  537*  1189  1613* 1614*
TEMP2   001250  174#  538*  1191
TEMP3   001252  175#  540*  566*  618   627*  1193  1386  1389  1501* 1821* 1823* 1824* 1825*
                1826* 2734
TEMP4   001254  176#  541*  1195  1399  1402  1408  1411  1487  1490  1496  1499  2734
TEMP5   001256  177#  542*  1197  1380* 1393* 1599
TIMER = 104416  243#
TKCSR   001204  148#  848   1214  1259  1655
TKDBR   001206  149#  850   856   1217  1219  1261  1657
TLAST = 026464 1350   2640#
TPCSR   001210  150#  832   854   1029  1262  1658
TPDBR   001212  151#  834*  856*  1031* 1264* 1660*
TRPOK   004730 1017#
TSTNO   001226  161#  491*  1102  1130  1334  1341  1343  1685* 1708* 1735* 1799* 1849* 1906*
                1960* 2017* 2074* 2131* 2188* 2245* 2303* 2361* 2419* 2477* 2535* 2593*
TST1    022324 1337   1355  1685#
TST10   024134 1961   2017#
TST11   024330 2018   2074#
TST12   024524 2075   2131#
TST13   024720 2132   2188#
TST14   025114 2189   2245#
TST15   025310 2246   2303#
TST16   025504 2304   2361#
TST17   025700 2362   2419#
TST2    022762 1686   1708#
TST20   026074 2420   2477#
TST21   026270 2478   2535#
TST22   026464 2536   2593#  2640
TST23 = ****** U 2594
TST3    023054 1709   1735#
TST4    023170 1736   1799#
TST5    023364 1800   1849#
TST6    023560 1850   1906#
TST7    023740 1907   1960#
TTST    003612  715*  716*  718*  719*  782#
TWOSYN= 010000   96#
```

```
TYPDAT  005206          1053    1071    1074#
TYPE   = 104402          219#    511     523     535     620     625     633     636     668     673     714     723     736
                         737     739     741     743     831     844     861     954     994     1054    1055    1058    1059
                        1061    1063    1067    1072    1121    1227    1230    1282    1328    1346    1352    1390    1412
                        1426    1429    1436    1439    1446    1452    1461    1468    1475    1590    1690
TYPMSG  005106          1051    1054#
VEC     006527          1187#   1404
VECMAP  012012          1589    1601#
WHICH   012004          1392    1597#
WRDCNT  004710           962#    995#   1003#
WRKO.F  005174          1066    1069#
XBX     005000          1028    1030    1032#
XCSR    003546           738     763#
XERR    003570           744     772#
XHEAD   006225           535    1187#
XLOC    003022           593#    611#    614     645     658#
XPASS   003562           742     769#
XSTATQ  007456           544    1187#
XTSTN   005330          1060    1100#
XVEC    003554           740     766#
ZERO    001300           186#
$COD   = ****** U          1
$CRAP  = 177777            1#   1673#   1676    1681#   1694#   1697    1704#   1725#   1728    1731#   1766#   1769    1795#
                        1838#   1841    1845#   1896#   1899    1902#   1950#   1953    1956#   2007#   2010    2013#   2064#
                        2067    2070#   2121#   2124    2127#   2178#   2181    2184#   2235#   2238    2241#   2292#   2295
                        2299#   2350#   2353    2357#   2408#   2411    2415#   2466#   2469    2473#   2524#   2527    2531#
                        2582#   2585    2589#
$ENDAD  003522           123     509     755#   1079
$N     = 000022            1#   1673    1681    1683    1688#   1694    1704    1706    1711#   1725    1731    1733    1738
                        1739#   1766    1795    1797    1803    1804#   1838    1845    1847    1853    1854#   1896    1902
                        1904    1910    1911#   1950    1956    1958    1964    1965#   2007    2013    2015    2021    2022#
                        2064    2070    2072    2078    2079#   2121    2127    2129    2135    2136#   2178    2184    2186
                        2192    2193#   2235    2241    2243    2249    2250#   2292    2299    2301    2307    2308#   2350
                        2357    2359    2365    2366#   2408    2415    2417    2423    2424#   2466    2473    2475    2481
                        2482#   2524    2531    2533    2539    2540#   2582    2589    2591    2597    2598#   2640#
$S     = 000024            1#   1686    1688#   1709    1711#   1736    1739#   1800    1804#   1850    1854#   1907    1911#
                        1961    1965#   2018    2022#   2075    2079#   2132    2136#   2189    2193#   2246    2250#   2304
                        2308#   2362    2366#   2420    2424#   2478    2482#   2536    2540#   2594    2598#
$Y     = 000017            1#    207#    215     217#    219#    221#    223#    225#    227#    229#    231#    233#    235#
                         237#   239#    241#    243#    245#
.      = 027736          108#    109     112#    119#    124#    127#    131#    135#    137#    189#    190#    191#    192#
                         272#   277#    279#    280#    281#    282#    284#    285#    286#    287#    289#    290#    291#
                         292#   294#    295#    296#    297#    299#    300#    301#    302#    304#    305#    306#    307#
                         309#   310#    311#    312#    314#    315#    316#    317#    319#    320#    321#    322#    324#
                         325#   326#    327#    329#    330#    331#    332#    334#    335#    336#    337#    339#    340#
                         341#   342#    344#    345#    346#    347#    349#    350#    351#    352#    354#    355#    356#
                         357#   515     525     657#    675     1110    1120    1168#   1187#   1203#   1205#   1260    1263
                        1284    1378    1422    1542    1546    1593    1624    1656    1659
.BEGIN  003152           690#
.CNVRT  004472           234     955#
.CONVR  004466           232     954#
.DATAC  005552           242    1159#
.DELAY  005436           238    1132#
.EOP    003364           731#   2594
.ERRTA  027604          1046    2734#
.HLT    004750           115    1026#
```

.INSTE	004154	224	861#		
.INSTR	004050	222	840#		
.INST1	004070	844#	864		
.MSG	004072	842*	845#		
.MSTCL	005466	236	1143#		
.PARAM	004174	226	872#		
.PFAIL	005336	113	478	1107#	1115
.RESOS	004434	230	943#		
.ROMCL	005504	240	1148#		
.SAVOS	004374	228	929#		
.SCOPE	003576	216	779#		
.SCOP1	003736	218	811#		
.START	002002	132	476#	492	1247
.TIMER	005616	244	1170#		
.TRPSR	004716	117	1014#		
.TRPTA	001330	214#	1019		
.TYPE	003766	220	822#		

M04


```
DMEND        1#    725
DMFRNT       1#
HLT         75#   1754   1764   1827   1867   1880   1893   1923   1935   1947   1978   1991   2004   2035   2048
          2061   2092   2105   2118   2149   2162   2175   2206   2219   2232   2263   2276   2289   2321   2334
          2347   2379   2392   2405   2437   2450   2463   2495   2508   2521   2553   2566   2579   2611   2624
          2637
$AUTO        1#    547
$BRRSH       1#   1725
$BUFFE       1#   1199
$COMP        1#
$CRAM        1#   1694
$CYCLE       1#   1271
$EOP         1#    725
$FINI        1#   2640
$GETPA       1#
$HEADE       1#
$JUMP        1#   1838   1896   1950   2007   2064   2121   2178   2235   2292   2350   2408   2466   2524   2582
$MARHI       1#
$MOCK        1#
$MSG         1#   1187
$PFAIL       1#   1103
$QUEST       1#   1381   1394   1403   1482   1491
$RAMCL       1#   1131
$RCLK        1#   1134   1137   1174   1179   1744   1746   1748   1755   1758   1814   1859   1861   1872   1874
          1885   1887   1915   1917   1927   1929   1939   1941   1970   1972   1983   1985   1996   1998   2027
          2029   2040   2042   2053   2055   2084   2086   2097   2099   2110   2112   2141   2143   2154   2156
          2167   2169   2198   2200   2211   2213   2224   2226   2255   2257   2268   2270   2281   2283   2313
          2315   2326   2328   2339   2341   2371   2373   2384   2386   2397   2399   2429   2431   2442   2444
          2455   2457   2487   2489   2500   2502   2513   2515   2545   2547   2558   2560   2571   2573   2603
          2605   2616   2618   2629   2631   2648   2650   2652   2660   2668   2676   2684   2692   2694   2696
          2704
$ROMNU       1#   1673
$ROMRD       1#   1766
$SCOPE       1#    775
$SIMBC       1#
$SOFTC       1#   1207
$TRPDE       1#    215    217    219    221    223    225    227    229    231    233    235    237    239    241
           243
$TSTN        1#   1683   1706   1733   1797   1847   1904   1958   2015   2072   2129   2186   2243   2301   2359
          2417   2475   2533   2591
$VARIA       1#    134
$XZ          1#   1673   1681   1694   1704   1725   1731   1766   1795   1838   1845   1896   1902   1950   1956
          2007   2013   2064   2070   2121   2127   2178   2184   2235   2241   2292   2299   2350   2357   2408
          2415   2466   2473   2524   2531   2582   2589

. ABS.  027736     000

 ERRORS DETECTED:  0

 CZDMGD,CZDMGD/SOL/CRF←IPLUT1,CZDMGD
 RUN-TIME: 11 16 1 SECONDS
 RUN-TIME RATIO: 496/28=17.3
 CORE USED:  21K  (41 PAGES)
```