

DR11W

DR11-W GEN NPR INTFC
CZDRLA0

AH-E780A-MC
COPYRIGHT 1980
FICHE 1 OF 1

JAN 1980
digital
MADE IN USA

This microfiche card contains a grid of frames. The frames are arranged in approximately 10 rows and 10 columns. Each frame contains a small amount of data, likely a page of text or a table. The data is too small to read clearly, but it appears to be organized in a structured format. The frames are separated by thin lines, and the overall layout is a standard microfiche format.

.REM 2

IDENTIFICATION

PRODUCT CODE: AC-E779A-MC
 PRODUCT NAME: CZDRLAO DP11-W GEN NPR INTFC
 DATE RELEASED: AUG. 1979
 MAINTAINER: DIAGNOSTIC ENGINEERING
 AUTHOR: JOHN W. CIKAJ

THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION. DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR ANY ERRORS THAT MAY APPEAR IN THIS DOCUMENT.

NO RESPONSIBILITY IS ASSUMED FOR THE USE OR RELIABILITY OF SOFTWARE ON EQUIPMENT THAT IS NOT SUPPLIED BY DIGITAL OR ITS AFFILIATED COMPANIES.

COPYRIGHT (C) 1979 BY DIGITAL EQUIPMENT CORPORATION.

THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION

DIGITAL	PDP	JNIBUS	MASSBUS
DEC	DECUS	DECTAPE	

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37

TABLE OF CONTENTS

39		
40		
41		
42		
43		
44		
45	1.0	ABSTRACT
46	2.0	REQUIREMENTS
47		2.1 EQUIPMENT
48		2.2 STORAGE
49		
50	3.0	TESTING MODES
51		3.1 DEFINITION
52		3.2 IMPLEMENTATION
53		
54	4.0	LOAD AND START PROLEDURE
55	5.0	SOFTWARE SWITCH REGISTER
56		5.1 OPTIONS
57		5.2 IMPLEMENTATION
58		5.3 CONTROL
59		5.4 PROGRAM AND/OR OPERATOR ACTION
60		
61	6.0	ERROR REPORTING
62	7.0	OPERATING MODES
63		7.1 MANUAL MODE
64		7.1.1 MANUAL MODE W/MULTIPLE BOARD FEATURE
65		7.1.2 MANUAL MODE W/BURST DATA LATE CALIBR.
66		
67		7.2 AUTO MODE
68		
69		
70		
71		
72		
73		
74		
75		
76	8.0	MISCELLANEOUS
77		8.1 RESTARTING PROGRAM IN CORE
78		8.2 POWER FAIL
79		
80	9.0	EXECUTION TIMES
81	10.0	SUBROUTINE ABSTRACTS
82	11.0	EXAMPLES
83		11.1 USING MULTIPLE BOARD FEATURE
84		
85		
86		
87		
88		
89	12.0	PROGRAM SUMMARY OF TESTS
90		

92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148

1.0 ABSTRACT

THIS DIAGNOSTIC PROGRAM IS CAPABLE OF TESTING EITHER THE DR11W OR THE DR11B NPR GENERAL INTERFACE.

IT HAS THE FOLLOWING FEATURES:

1. ACT11/XXDP COMPATIBLE
2. MULTIPLE BOARD TESTING USING TABLE CREATED BY USER
3. BURST DATA LATE CALIBRATION
4. INDEPENDANT 'LOGIC WRAP-AROUND' AND 'CABLE WRAP-AROUND' TESTING

2.0 REQUIREMENTS

2.1 [EQUIPMENT]

1. PDP11 STANDARD COMPUTER
2. I/O TYPE TERMINAL
3. DR11-W
4. LOOP BACK CABLE (OPTIONAL)

2.2 [STORAGE]

THE PROGRAM USES 4600 WORDS OF MEMORY

3.0 TESTING MODES

3.1 [DEFINITION]

THE DR11W DIAGNOSTIC PROVIDES TWO INDEPENDANT MODES OF LOGIC TESTING:

1. MAINTENANCE MODE TESTING
2. CABLE MODE TESTING

IN CONTEXT OF THE DIAGNOSTIC, 'MAINTENANCE MODE TESTING' IS DEFINED AS TESTS WHICH ARE PERFORMED WITH BIT 12 OF THE CSR SET. THIS RESULTS IN 'LOGIC WRAP-AROUND' WHICH ALLOWS TESTING WITHOUT A USER DEVICE OR PHYSICAL CABLE. LOGIC WRAP-AROUND CHECKS ALL LOGIC FUNCTIONALITY EXCEPT MODULE CONNECTORS, CABLES, AND I/O DRIVERS AND RECEIVERS. 'CABLE MODE TESTING', THEREFORE, CHECKS ALL THESE THAT THE LOGIC WRAP-AROUND DOESN'T.

3.2 [IMPLEMENTATION]

THE TWO MODES OF TESTING ARE INVOKED AS FOLLOWS:

1. THE PROGRAM WILL ALWAYS DO THE MAINTENANCE MODE TESTS (LOGIC WRAP-AROUND)

149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205

2. IF SWITCH REGISTERS 9 AND 10 ARE 0, THEN THE PROGRAM WILL AUTOMATICALLY CHECK IF THE PHYSICAL WRAP-AROUND CABLE IS IN PLACE. IF IT IS, 'CABLE TESTS' WILL BE DONE. IF NOT, CABLE TESTS ARE SKIPPED.

NOTE: STEP 2 CAN BE ALTERED BY USING SWITCH REGISTERS 9 OR 10. SEE SECT. 5.1

4.0 LOAD AND START PROCEDURE

1. LOAD PROGRAM INTO MEMORY
2. LOAD STARTING ADDRESS 200
3. PRESS START-SEE SECT 7.0 FOR OPERATING MODFS

5.0 SOFTWARE SWITCH REGISTER

5.1 [OPTIONS]

SWITCH	OCTAL	FUNCTION
-----	-----	-----
SW15=1	100000	HALT ON ERROR
SW14=1	040000	LOOP ON TEST
SW13=1	020000	INHIBIT ERROR TYPEOUTS
SW11=1	004000	INHIBIT ITERATIONS
SW10=1	002000	PERFORM MAINT. MODE (INTERNAL LOGIC WRAP-AROUND) TESTS ONLY.
SW09=1	001000	PERFORM CABLE MODE TESTS UNCONDITIONALLY (DO NOT ALLOW PROGRAM TO DECIDE WHETHER CABLE IS IN)
SW08=1	000400	AFTER 1ST OCCURRENCE OF ERROR DURING A PASS OF PROGRAM, JUMP TO THE NEXT BOARD SPECIFIED IN TABLE FOR TESTING.

5.2 [IMPLEMENTATION]

IF THE DIAGNOSTIC IS RUN ON A CPU WITHOUT A SWITCH REGISTER THEN A SOFTWARE SWITCH REGISTER IS USED WHICH ALLOWS THE USER THE SAME SWITCH OPTIONS AS THE HARDWARE SWITCH REGISTER. IF THE HARDWARE SWITCH REGISTER DOES NOT EXIST OR IF ONE DOES AND IT CONTAINS ALL ONES (177777) THEN THE SOFTWARE SWITCH REGISTER (LOC. 176) IS USED.

5.3 [CONTROL]

THIS PROGRAM SUPPORTS THE DYNAMIC LOADING OF THE SOFTWARE SWITCH REGISTER (LOC 176) FROM THE TTY. THIS IS ACCOMPLISHED AS FOLLOWS:

1. TYPE CONTROL G <^G>; THIS ALLOWS THE TTY TO ENTER DATA INTO LOC. 176 AT SELECTED POINTS WITHIN THE PROGRAM.
2. THE MACHINE WILL TYPE: SWR=XXXXXX NEW (XXXXXX IS THE OCTAL CONTENTS OF THE SOFTW. SWITCH REGISTER)
3. AFTER THE 'NEW' THE OPERATOR CAN DO ONE OF THE FOLLOWING:

206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262

A. TYPE A NUMBER TO BE LOADED INTO LOC. 176 FOLLOWED BY A <CR>
(ONLY NUMBERS BETWEEN 0-7 WILL BE ACCEPTED AND ONLY 6 NUMBERS
WILL BE ALLOWED).
IF A <CR> IS THE FIRST ENTRY THE SOFTWARE SWITCH REGISTER WILL
NOT BE CHANGED.

B. IF A CONTROL U <^U> IS DEPRESSED THE PROGRAM WILL GO BACK
TO STEP 2.

5.4 [PROGRAM AND/OR OPERATOR ACTION]

LOADING AND STARTING AT 200 WITH ALL SWITCHES DOWN
IS NORMAL LOGIC TESTING. IF AN ERROR
IS DETECTED HERE, THERE WILL BE A PRINTOUT. WHEN
AN ERROR IS DETECTED AND IT IS NECESSARY TO SCOPE
ON IT, PLACE SW15 UP TO HALT ON ERROR, THEN SW14 UP
TO LOOP ON ERROR, THEN SW13 UP TO DELETE PRINTOUTS.

6.0 ERROR REPORTING

DEPENDING ON WHICH BOARD IS BEING TESTED, THE STATUS PRINTOUT WILL
REFLECT EITHER THE DR11W OR THE DR11B.

ALL ERRORS ARE ACCOMPANIED WITH THE FOLLOWING PRINTOUT:

TESTNO	ERRPC	PSW	DR11W/B STATUS
--------	-------	-----	----------------

WHERE:

TESTNO	-TEST NUMBER WHERE ERROR OCCURRED
ERRPC	- LISTING ADDRESS WHERE THE ERROR WAS DETECTED
PSW	-PROCESSOR STATUS WORD AT ERROR TIME
DR11W/B STATUS	-CONTENTS OF DR11W/B CONTROL STATUS REGISTER AT ERROR TIME

7.0 OPERATING MODES

7.1 [MANUAL MODE]

DEFINED AS NON-AUTOMATIC USE OF DIAGNOSTIC.FOR EXAMPLE:

A.LOADING PROGRAM VIA PAPER TAPE
B.LOADING FROM XXDP PACKAGE VIA KEYBOARD
C.REQUESTING A PROGRAM FROM ACT11 VIA ACT11 STATION SWITCHES

ONLY UNDER MANUAL MODE DOES THE DIAGNOSTIC OFFER 'BURST
DATA LATE' CALIBRATION AND 'MULTIPLE BOARD ' DIALOGUE.
AFTER LOADING PROGRAM, DEPOSITING SA 200, AND PRESSING
START, THE PROGRAM TYPES THE FOLLOWING(NOTE:USER INPUT
IS UNDERSCORED):

CZDRL DR11-W GEN NPR INTFC

263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319

DO YOU WISH BURST DATA LATE CALIBRATION?
(Y=YES,N=NO): N

DO YOU WISH MULTIPLE BOARD DIALOGUE?
(Y=YES,N=NO): N

IF SOFTWARE SWITCH REGISTER IS INVOKED, THEN THE FOLLOWING WILL BE PRINTED:

SWR XXXXXX NEW=

(REFER TO SECT. 5.0 FOR OPERATOR OPTIONS)

ANSWERING 'NO' TO THE ABOVE QUESTIONS (AND ASSUMING THAT MULTIPLE BOARD SIZING HAD NOT BEEN PREVIOUSLY DONE) THE PROGRAM DEFAULTS TO ONE BOARD TESTING SPECIFIED BY REGISTER ADDRESS 'DEFRAD'(172410) AND VECTOR ADDRESS 'DEFVAD'(124). THE PROGRAM THEN TYPES:

*BRD ADDR.: 172410

IF NO SWITCH REGISTER OPTIONS ARE SELECTED THE PROGRAM WILL 'END PASS' THE FIRST TIME WITH NO ITERATIONS. SUBSEQUENT 'END PASS' WILL BE WITH ITERATIONS.

THE 'END PASS' TYPEOUT IS AS FOLLOWS (IN THE CASE OF MAINT MODE TESTING):

TEST MODE: MAINT MODE ONLY
END PASS # XXXXXX

7.1.1 [MANUAL MODE W/MULTIPLE BOARD FEATURE]

THE DIAGNOSTIC CONTAINS AN INTERACTIVE MONITOR WHICH ALLOWS THE USER TO CREATE AND MODIFY A TABLE DESCRIBING UP TO 16 BOARD REGISTER AND VECTOR ADDRESSES FROM WHICH THE PROGRAM WILL SUBSEQUENTLY RUN. THE PROGRAM WILL SEQUENTIALLY RUN THRU EACH BOARD DESCRIBED IN THE TABLE FOR 'ENDPAS #1' AND THEN DO THE SAME FOR 'ENDPAS #2', ETC. ALL BOARDS FOR 'ENDPAS #1' EXPERIENCE TESTING WITH NO ITERATIONS. THUS, TYPING 'Y' TO THE QUESTION 'DO YOU WISH MULTIPLE BOARD DIALOGUE?' ENTERS THE USER INTO THE MULTIPLE BOARD MONITOR WITH THE PROMPT '' TO WHICH THE FOLLOWING COMMANDS MAY BE TYPED:

' L ' - LIST PRESENT MULTIPLE BOARD TABLE
' E ' - EDIT PRESENT MULT BRD TBL
' R ' - START PROGRAM TESTING

AT ANY POINT DURING PROGRAM RUN A 'RUN SUMMARY' MAY BE RECEIVED BY TYPING A 'CONTROL S' <^S>. THE PROGRAM WILL DETECT

320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376

THE ^S AND PRINT:

RUN SUMMARY.....

BOARD	PASCNT	ERRCNT
XXXXXX	XXX	XXX
XXXXXX	XXX	XXX
ETC.	ETC.	ETC.

WHERE:

BOARD	- LISTING OF ALL BOARD REGISTER ADDRESSES SPECIFIED IN MULTIPLE BOARD TABLE
PASCNT	- # PASSES DONE ON EACH BOARD SPECIFIED
ERRCNT	- # OF PASCNT'S WHERE ONE OR MORE ERRORS WERE DETECTED

7.1.c [MANUAL MODE W/BURST DATA LATE CALIBR.]

THE DIAGNOSTIC CONTAINS A ROUTINE WHICH AIDS THE USER IN 'BURST DATA LATE' CALIBRATION. AS STATED IN THE DR11-W ENGINEERING SPECIFICATION, THE 'BURST DLT' MULTIVIBRATOR TIME OUT MUST BE CALIBRATED SO AS TO BE COMPATIBLE WITH THE USER DEFINED TRANSFER RATE IN BURST MODE OPERATION. EFFECTIVELY, THE PROGRAM SOFTWARE ROUTINE SETS THE CYCLE BIT IN THE CSR OF TH DR11W, FOLLOWS WITH A DELAY, AND THEN CLEARS THE CYCLE BIT. THIS CONTINUES REPEATEDLY UNTIL THE USER HALTS THE CPU. THUS, WHEN THE USER SUBSEQUENTLY ATTACHES A SCOPE PROBE TO E88-6 ON THE DR11-W (REFER TO PRINT SET M8716-0-1) A POSITIVE PULSE WILL BE OBSERVED. THE PULSE SHOULD BE SET BETWEEN 5-30 US. BY ADJUSTING POT. R88.

THEREFORE, ANSWERING 'Y' TO THE QUESTION: 'DO YOU WISH BURST DATA LATE CALIBRATION?' WILL RESULT IN THE FOLLOWING PRINTOUT:

```
BURST DATA LATE CALIBRATION IN PROGRESS..
ATTACH SCOPE PROBE...
TO ABORT THIS ROUTINE, HALT PROCESSOR...
```

THE CALIBRATION ROUTINE WILL FUNCTION ON ONE BOARD SPECIFIED BY 'DEFRAD'(172410) AND 'DEFVAD'(124).

7.2 [AUTO MODE]

DEFINED AS USING THE PROGRAM IN A SITUATION OF SEQUENTIAL EXECUTION OF TWO OR MORE PROGRAMS ACCORDING TO A PREDETERMINED SEQUENCE AND WITHOUT OPERATOR INTERVENTION.

AUTOMATIC MODES ARE:

A. ACT11 QUICK VERIFY
B. ACT11 AUTO ACCEPT
C. XXDP CHAIN MODE

377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433

THE DR11W DIAGNOSTIC HAS THE FOLLOWING RUN CHARACTERISTICS WHEN OPERATING IN AUTO MODE:

A. THE PROGRAM WILL TEST ONLY ONE BOARD SPECIFIED BY 'DEFRAD' AND 'DEFVAD'.

B. THE FOLLOWING WILL NOT BE OFFERED TO THE USER:

1. BURST DATA LATE CALIBRATION
2. MULTIPLE BOARD DIALOGUE
3. RUN SUMMARY <^S>

8.0 MISCELLANEOUS

8.1 [RESTARTING PROGRAM IN CORE]

WHENEVER THE PROGRAM IS HALTED AND RESTARTED AT SA 200, ALL HISTORY OF PREVIOUS TESTING IS LOST. HOWEVER, THE MULTIPLE BOARD TABLE AS PREVIOUSLY EDITED IS LEFT INTACT. IT WILL REMAIN INTACT UNTIL:

1. ANOTHER PROGRAM IS LOADED INTO CORE
2. THE USER RE-EDITS THE TABLE

8.2 [POWER FAIL]

THERE IS NO RESTART MESSAGE NOR PROVISIONS FOR AUTOMATICALLY RESTARTING PROGRAM UPON POWER UP. IF NON VOLATILE MEMORY, THEN RESTART AT SA 200.

9.0 EXECUTION TIMES

ON A PDP11/04:

1. MAINT MODE ONLY/NO ITER.: APPROX. 5 SEC.
2. MAINT MODE ONLY/WITH ITER.: APPROX. 35 SEC.
3. MAINT & CABLE MODE/NO ITER.: APPROX. 7 SEC.
4. MAINT & CABLE MODE/WITH ITER.: APPROX. 60 SEC..

10.0. SUBROUTINE ABSTRACTS

10.1 SCOPE

THIS SUBROUTINE CALL IS PLACED BETWEEN EACH SUBTEST IN THE INSTRUCTION SECTION. IT RECORDS THE STARTING ADDRESS OF EACH SUB-TEST AS IT IS BEING ENTERED. IF A SCOPE LOOP IS REQUESTED, IT WILL JUMP TO THE START OF THE SUBTEST THAT THE SCOPE LOOP IS REQUESTED FOR. IF SCOPE LOOP IS NOT REQUESTED, THERE

434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490

WILL BE EITHER A FIXED OR RANDOM NUMBER OF ITERATIONS ON THAT SUBTEST BEFORE THE NEXT SUBTEST IS ENTERED. SWITCH 11 ON A 1 INHIBITS ITERATION OF SUBTESTS. NOTE: SUPPORTS ^G ROUTINE FOR DYNAMIC LOADING OF SOFTWARE SWITCH REGISTER

10.2 HALT

IS A ROUTINE THAT PRINTS-OUT AN ADDRESS THAT TAGS THE FAILING SUBTEST, THE CP STATUS REGISTER AND THE DR11W STATUS REGISTER AT THE TIME OF FAILURE. NOTE: SUPPORTS ^G ROUTINE FOR DYNAMIC LOADING OF SOFTWARE SWITCH REGISTER

10.3 LODBUF

THE INBUF BUFFER IS LOADED WITH AN INCREMENTING PATTERN (0,1,2,3,...) BEGINNING AT THE STARTING ADDRESS OF INBUF. THE NUMBER OF WORDS LOADED IS DETERMINED BY THE CONTENTS OF BUFLN.

10.4 CHKBFF

THE CHKBUFF BUFFER IS LOADED WITH A MODIFIED INCREMENTING PATTERN (0,0,2,2,4,4,6,6,...) BEGINNING AT THE STARTING ADDRESS OF CHKBUFF. THE NUMBER OF WORDS LOADED IS DETERMINED BY THE CONTENTS OF BUFLN. THIS BUFFER IS LOADED ONLY FOR TESTS WHICH USE THE MAINTENANCE MODE OF THE DR11-W WHICH HAS A SPECIAL ALTERNATING DATI-DATO SEQUENCE OF OPERATION.

10.5 INTA

THE IE BIT IS CLEARED IN THE CSR THEN THE CSR IS CHECKED FOR THE ABSENCE OF AN ERROR AND THE PRESENCE OF READY. THE WCR IS CHECKED TO SEE THAT IT IS EQUAL TO ZERO. THE CORRECT CONTENTS OF THE BAR ARE CALCULATED AND CHECKED. THERE IS A JSR TO THE NORMAL SUB-ROUTINE BEFORE THIS ROUTINE IS EXITED. THE PROGRAM WILL HALT IF ERROR IS SET, READY IS CLEAR, OR READY AND ERROR ARE CLEAR.

10.6 DATCHK

THIS ROUTINE IS ENTERED TO CHECK INBUF AFTER A MAINTENANCE MODE OPERATION. THE CONTENTS OF INBUF AND THE CONTENTS OF CHKBUFF ARE CHECKED TO SEE THAT THEY ARE THE SAME. THE NUMBER OF COMPARISONS MADE IS DETERMINED BY THE CONTENTS OF BUFLN.

10.7 NORMAL

THE ROUTINE IS ENTERED FROM INTA AND FROM SOME TESTS WHICH DON'T USE INTA. THE NUMBER OF

491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547

THE DRINV+2 IS PUT INTO DRINV AND THE DRVS IS
CLEARED. IF THE DR11-W INTERRUPTS UNDER THESE
CONDITIONS THE PDP-11 WILL HALT AT DRVS. THE
PROCESSOR STATUS WORD IS RESTORED TO LEVEL 7 AND
THE ROUTINE IS EXITED.

10.8 DATOCK

AFTER A STRING OF DATO'S HAS BEEN COMPLETED THIS
ROUTINE CHECKS THAT THE CORRECT DATA PATTERN
(52525) WAS TRANSFERRED TO INBUF. THE NUMBER
OF COMPARISONS MADE IS DETERMINED BY THE CONTENTS
OF BUFLN. AN ADDITIONAL CHECK IS MADE ON BUFLN+2
TO INSURE THAT TOO MANY WORDS WEREN'T TRANSFERRED.

10.9 ERRCHK

THIS ROUTINE CLEARS IE AND HALTS IF ERROR IS SET.

10.10 TRAPCATCHER

THIS IS A SERIES OF INSTRUCTIONS STARTING AT LOCATION 0
DESIGNED TO DETECT AND ISOLATE UNEXPECTED TRAPS AND
INTERRUPTS TO THE TRAP AND INTERRUPT VECTOR AREA OF
MEMORY.

EACH VECTOR ENTRANCE ADDRESS IS LOADED WITH THE ADDRESS
OF THE NEXT LOCATION. THE NEXT LOCATION IS LOADED WITH
A HALT (00000). THUS AN ILLEGAL TRAP OR INTERRUPT WILL
CAUSE A HALT AT THE TRAP LOCATION PLUS TWO.

IF A HALT OCCURS IN THE TRAP OR INTERRUPT AREA, EXAMINE
REGISTER SIX, IT WILL CONTAIN THE CURRENT STACK ADDRESS.
THE CONTENTS OF THE CURRENT STACK ADDRESS IS THE VALUE OF
THE LOCATION COUNTER WHEN THE TRAP OR INTERRUPT OCCURRED.

11.0 EXAMPLES

11.1 [USING MULTIPLE BOARD FEATURE (REFER TO SECT. 7.1.1)]

(USER LOADS PROGRAM AND STARTS AT 200)

CZDRL DR11-W GEN NPR INTFC

DO YOU WISH BURST DATA LATE CALIBRATION?
(Y=YES, N=NO): N

-

DO YOU WISH MULTIPLE BOARD DIALOGUE?
(Y=YES, N=NO): Y

-

L

-

548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604

NO. BOARDS REQUESTED FOR TESTING: 1

BOARD#	REGSTR.	VECTOR
1	172410	000124

- E

HOW MANY BOARDS DO YO WANT TO TEST? 3 <CR>

BOARD 1
 REGISTER ADDRESS: 172410 <CR> <---(USER INPUTS <CR>:CONTENTS STAY SAME)
 VECTOR ADDRESS: 000124 <CR>

BOARD 2
 REGISTER ADDRESS: 000000 172430 <CR> <---(0 CONTENTS ARE CHANGED)
 VECTOR ADDRESS: 000000 170 <CR>

BOARD 3
 REGISTER ADDRESS: 000000 172450 <CR>
 VECTOR ADDRESS: 000000 174 <CR>

- L

BOARD #	REGSTR.	VECTOR
1	172410	000124
2	172430	000170
3	172450	000174

- R

(PROGRAM NOW RUNS USING MULT. BOARD TABLE)

*BRD ADDR.: 172410
 TEST MODE: MAINT & CABLE
 END PASS # 1

*BRD ADDR.: 172430
 TEST MODE: MAINT & CABLE
 END PASS # 1

*BRD ADDR.: 172450
 TEST MODE: MAINT & CABLE
 END PASS # 1

*BRD ADDR.: 172410
 TEST MODE: MAINT & CABLE

605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647

END PASS # 2
*BRD ADDR.: 172430
TEST MODE: MAINT & CABLE
END PASS # 2
*BRD ADDR.: 172450
TEST MODE: MAINT & CABLE
END PASS # 2

.
.
.
ETC.

(AT ANY TIME USER MAY REQUEST 'RUN SUMMARY' BY TYPING ^S)

^S

RUN SUMMARY.....

BOARD	PASCNT	ERRCNT
172410	10	0
172430	10	0
172450	9	0

(AFTER RUN SUMMARY, PROGRAM RESTARTS
AUTOMATICALLY AT SA 200)

CZDRL DR11-W GEN NPR INTFC

DO YOU WISH BURST DATA LATE CALIBRATION?
(Y=YES, N=NO): N

-
.
.
ETC.

• 649
650
651
652
653
654

12.0 PROGRAM SUMMARY

z

672

```

.TITLE CZDRLAO-DR11W GEN NPR INTFC
.*COPYRIGHT (C) 1979
.*DIGITAL EQUIPMENT CORP.
.*MAYNARD, MASS. 01754
.*
.*PROGRAM BY JOHN W. CIUKAJ
.*
.*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
.*PACKAGE (MAINDEC-11-DZQAC(-C3), JAN 19, 1977.
.*

```

```

000001
160000
673 000240
674 000004
675 104000
676 000001
677 000002
678 000004
679 000010
680 000020
681 000040
682 000100
683 000200
684 000400
685 001000
686 002000
687 004000
688 010000
689 020000
690 040000
691 100000
692
693 000004
694 000000
695 000001
696 000002
697 000003
698 000004
699 000005
700 000006
701 000007
702 000007
703 000006
704
705 000001
706 000002
707 000004
708 000010
709 000020
710 000040
711 000100
712 000200
713 000400
714 001000
715 002000
716 004000
717 010000

```

```

$TN 1
$SWR 160000 ::HALT ON ERROR, LOOP ON TEST, INHIBIT ERROR TYP0UT
NOP=000240
SCOPE=IOT
HLT=EMT
BIT0=000001
BIT1=000002
BIT2=000004
BIT3=000010
BIT4=000020
BIT5=000040
BIT6=000100
BIT7=000200
BIT8=000400
BIT9=001000
BIT10=002000
BIT11=004000
BIT12=010000
BIT13=020000
BIT14=040000
BIT15=100000
BUSERR=000004
R0=%0
R1=%1
R2=%2
R3=%3
R4=%4
R5=%5
R6=%6
R7=%7
PC %7
SP %6
GO=1
FNCT1=2
FNCT2=4
FNCT3=10
XBA16=20
XBA17=40
IE=100
READY=200
CYCLE=400
DSTATA=1000
DSTATB=2000
DSTATC=4000
MAINT=10000

```

```

718      020000      ATTN=20000
719      040000      NEX=40000
720      100000      ERROR=100000
721      000200      CRLF=200
722      022404      STACK=BUFF
723      000034      TRAPVEC=34
724      000004      ERRVEC=4
725      177570      DSWR=177570
726      177570      DDISP=177570
727      000060      TKVEC=60
728
729      ;LOAD TRAP CATCHER INTO 0 THRU 777.
730
731 000000      .ASECT
732      000000      .=0
739      000020      .=20
740 000020      014030      SCOPEC
741 000022      000340      340
742      000030      .-30
743 000030      013314      PRINT
744 000032      000340      340
745      000034      .-34
746 000034      017176      $TRAP
747 000036      000340      340
748      000046      .=46
749 000046      013250      $ENDAD
750      000052      .=52
751 000052      000000      0
752      000174      .=174
753 000174      000000      DISPREG:      0
754 000176      000000      SWREG:      0
755      000200      . 200
756 000200      012706      022404      MOV      #STACK,SP
757 000204      005077      000572      CLR      @PSW
758 000210      005737      000042      TST      @#42
759 000214      001411      BEQ      1$      ;ARE WE IN AUTO MODE
760
761 000216      023737      000042      000046      CMP      @#42,@#46
762 000224      001415      BEQ      2$      ;IF NOT,DO TITLE,CALIBRATION
763 000226      012702      014646      MOV      #TITLE,R2      ;AND MULTIPLE BOARD SIZING
764 000232      004757      015372      JSR      PC,TTOUT      ;IS THIS ACT11 AUTO MODE
765 000236      000410      BR      2$      ;IF SO ,DON'T PRINT TITLE
766 000240      012702      014646      1$:      MOV      #TITLE,R2      ;PRINT THE TITLE
767
768 000244      004737      015372      JSR      PC,TTOUT
769 000250      004737      020540      JSR      PC,BRSTD1      ;BURST DATA LATE CALIBRATION?
770 000254      004737      017256      JSR      PC,MBD      ;MULTIPLE BOARD DIALOGUE
771 000260      005037      001200      2$:      CLR      FLAG
772 000264      000137      001364      JMP      $SWP
773      001000      .=1000
774 001000      177570      SR:      177570
775 001002      177776      PSW:      177776
776 001004      172410      DEFRAD: 172410      ;DEFAULT REGISTER ADDRESS
777 001006      000124      DEFVAD: 124      ;DEFAULT VECTOR ADDRESS
778 001010      000000      PNTRAD: 0      ;POINTER INTO MULTIPLE BOARD TABLE SPECIFYING
779
780 001012      000000      PNTVAD: 0      ;CURRENT REGISTER ADDRESS BEING TESTED
;POINTER FOR VECTOR ADDRESS

```



```

781 001014 000000      PNTPAS: 0          ;POINTER INTO PASCNT TABLE
782 001016 000000      PNTERR: 0         ;POINTER INTO ERROR COUNT TABLE
783 001020 000000      PRGCYC: 0         ;COUNTER TO TRACK WHEN ALL BOARDS ARE TESTED
784 001022 000001      QTYBRD: 1         ;TOTAL # DR11W BOARDS BEING TESTED
785 001024 172410      REGADR: 172410
786 001026             .BLKW 15.      ;TOTAL:16 LOCATIONS FOR BOARD ADDRESSES
787
788 001064 172410      WCR: 172410
789 001066 172412      BAR: 172412
790 001070 172414      CSR: 172414
791 001072 172416      BDR: 172416
792 001074 000240      DRINL: 240
793 001076 000124      VECADR: 124
794 001100             .BLKW 15.      ;TOTAL:16 LOCATIONS FOR VECTOR ADDRESSES
795 001136 000124      DRINV: 124
796 001140 000126      DRVS: 126
797 001142 000000      EIR: 0          ;TEMP STORAGE FOR CSR #2 (ERROR IND. REG)
798 001144 000000      BORW: 0
799 001146 052525      NPR1: 52525
800 001150 173000      DIOMEM: 173000
801 001152 022406      INBUF: XINBUF
802 001154 023410      CHKBUF: XCHKBU
803 001156 000000      BUFLN: HALT
804 001160 000000      LENCHK: HALT
805 001162 000000      BRWAIT: HALT
806 001164 000000      WCLEN: HALT
807 001166 000000      RDYCHK: HALT
808 001170 177560      TKS: 177560
809 001172 177562      TKB: 177562
810 001174 177564      TPS: 177564
811 001176 177566      TPB: 177566
812 001200 000000      FLAG: 0
813 001202 000000      FNCCNT: HALT
814 001204 000000      INBUF1: HALT
815 001206 000000      PASCNT: 0        ;NUMBER OF PASSES COMPLETED: 16 BOARDS
816 001210             .BLKW 15.
817 001246 000000      ERRCNT: 0        ;ERROR COUNT TABLE:16 BOARDS
818 001250             .BLKW 15.
819 001306 000000      TEMP: 0          ;TEMPORARY STORAGE
820 001310 000000      TEMP2: 0         ;TEMPORARY STORAGE
821 001312 000000      TIME: 0         ;GENERAL PURPOSE TIMER
822 001314 000000      CABLE: 0        ;CABLE MODE TO BE DONE?:0-NO,1-YES
823 001316 000000      HLTDET: 0       ;FLAG THAT AN ERROR OCCURRED DURING PROGRAM PASS
824 001320 000000      TESTNO: 0      ;SPECIFIES TEST # FOR ERROR PRINT
825 001322 000000      LOOP: 0        ;GEN'L PURPOSE LOOP COUNTER
826 001324 177570      DISPLAY: .WORD  DDISP
827 001326 177570      SWR: 177570
828
829 001330 177560      $TKS: 177560
830 001332 177562      $TKB: 177562
831 001334 177564      $TPS: 177564
832 001336 177566      $TPB: 177566
833 001340 000        $NULL: .BYTE 0
834 001341 002        $FILLS: .BYTE 2
835 001342 012        $FILLC: .BYTE 12
836 001343 000        $TPFLG: .BYTE 0
837 001344 207        $BELL: .ASCII <207><377><377>

```

377

377

DDISP <207><377><377>


```

876 001626 012737 001024 001010      MOV      #REGADR,PNTRAD
877 001634 012737 001076 001012      MOV      #VECADR,PNTVAD
878 001642 012737 001206 001014      MOV      #PASCNT,PNTPAS
879 001650 012737 001246 001016      MOV      #ERRCNT,PNTERR
880
881
882                                     ;SET UP BOARD ADDRESSES OF CURRENT BOARD FROM
883                                     ;MULTIPLE BOARD TABLE
884
885 001656 005237 001020      BRDSET: INC      PRGCYC
886 001662 005737 000042      TST      @#42          ;ARE WE IN MANUAL MODE?
887 001666 001411      BEQ      LOAD          ;YES;CHECK MULT BRD TABLE FOR NEXT BOARD ADDRESS
888 001670 013737 001004 001024      BEGAUT: MOV      DEFRAD,REGADR      ;NO;USE DEFAULT REGISTER ADDRESS
889 001676 013737 001006 001076      MOV      DEFRAD,REGADR
890 001704 012737 000001 001022      MOV      #1,QTYBRD
891 001712 012700 001064      LOAD:  MOV      #WCR,R0 ;LOAD WRD CNT,BUS ADDRESS,STATUS,& DATA BUFFER ADDRESSES
892 001716 017701 177066      MOV      @PNTRAD,R1
893 001722 010120      2$:    MOV      R1,(R0)+
894 001724 062701 000002      ADD      #2,R1
895 001730 022700 001074      CMP      #BDR+2,R0
896 001734 001372      BNE      2$
897 001736 012700 001136      MOV      #DRINV,R0      ;LOAD INTERRUPT VECTOR ADDRESSES
898 001742 017701 177044      MOV      @PNTVAD,R1
899 001746 010120      3$:    MOV      R1,(R0)+
900 001750 062701 000002      ADD      #2,R1
901 001754 022700 001142      CMP      #DRVS+2,R0
902 001760 001372      BNE      3$
903 001762 104401 001770      TYPE     ,65$          ;;TYPE ASCIZ STRING
      001766 000410      BR       64$          ;;GET OVER THE ASCIZ
      002010
904 002010 013746 001064      ;;65$: .ASCIZ <CRLF><CRLF>/ *BRD ADDR.: /
      002014 104402
905 002016 104401 002024      MOV      WCR,-(SP)      ;;SAVE WCR FOR TYPEOUT
      002022 000401      TYPOC      ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
      TYPE     ,67$      ;;TYPE ASCIZ STRING
      BR       66$      ;;GET OVER THE ASCIZ
      ;;67$: .ASCIZ <CRLF>
      66$:
      1$:
906 002026
907 002026 005037 001314      CLR      CABLE
908 002032 012706 022404      MOV      #BUFF,SP
909 002036 012777 000340 176736      MOV      #340,@PSW      ;PROC. AT LEVEL #7
910 002044 013737 002070 014126      MOV      T1STR,RETURN
911 002052 005037 014124      CLR      SCOPEF
912 002056 005037 001320      CLR      TESTNO
913 002062 005037 014122      CLR      ICOUNT

```

915
916
917

.SBTTL
.SBTTL
.SBTTL

MAINTENANCE MODE TESTING

```

919 .SBTTL TEST 1 CAN ALL DR11W REG BE ADDRESSED WITHOUT ERROR?
920 :*****
921 : TEST 1 CAN ALL DR11-W REG BE ADDRESSED WITHOUT ERROR?
922 :*****
923
924
925 002066 000004 SCOPE
926 002070 012737 002000 014122 T1STR: MOV #2000,ICOUNT ;
927 002076 012737 002204 000004 MOV #CPUERR,BUSERR ;CPU ERROR VECTOR TO CPUERR
928 002104 013746 001064 CKWCR: MOV WCR,-(SP) ;SAVE ADDRESS FOR TYPE
929 002110 005777 176750 TST @WCR ;ACCESS REGISTER
930 002114 005726 TST (SP)+ ;OK;READJUST STACK
931 002116 013746 001066 CKBAR: MOV BAR,-(SP)
932 002122 005777 176740 TST @BAR
933 002126 005726 TST (SP)+
934 002130 013746 001070 CKCSR: MOV CSR,-(SP)
935 002134 005777 176730 TST @CSR
936 002140 005726 TST (SP)+
937 002142 013746 001072 CKBDR: MOV BDR,-(SP)
938 002146 005777 176720 TST @BDR
939 002152 005726 TST (SP)+
940 002154 013746 001136 CKDRIN: MOV DRINV,-(SP)
941 002160 013777 001140 176750 MOV DRVS,@DRINV
942 002166 005726 TST (SP)+
943 002170 013746 001140 CKDRVS: MOV DRVS,-(SP)
944 002174 005077 176740 CLR @DRVS
945 002200 005726 TST (SP)+
946 002202 000455 BR CKFIN
947 002204 022626 CPUERR: CMP (SP)+,(SP)+
948 002206 104401 002214 TYPE ,65$ ;;TYPE ASCIZ STRING
002212 000423 BR 64$ ;;GET OVER THE ASCIZ
002262 64$: .ASCIZ <CRLF>/TESTNO 1-CPU ERROR WHILE ADRESSING /
949 002262 104402 TYPOC
950 002264 104401 001354 TYPE ,SENULL
951 002270 012737 000006 000004 MOV #6,BUSERR ;RESTORE #6 TO BUS ERROR VECTOR
952 002276 104401 002304 TYPE ,67$ ;;TYPE ASCIZ STRING
002302 000413 BR 66$ ;;GET OVER THE ASCIZ
002332 66$: .ASCIZ <CRLF>/SKIPPING ALL TESTS /
953 002332 000137 013132 JMP END
954 002336 012737 000006 000004 CKFIN: MOV #6,BUSERR ;RESTORE #6 TO BUS ERROR VECTOR
955
956
957
    
```

959
960
961
962
963 002344 000004
964 002346 005037 014122
965 002352 005077 176512
966 002356 052777 100000 176504
967 002364 017737 176500 001144
968 002372 032737 000001 001144
969 002400 001403
970 002402 104401 015310
971 002406 000402
972 002410 104401 015230

```
                  .SBTTL TEST 2 ARE WE TESTING A DR11W OR A DR11B?  
:*****  
:                  TEST 2 ARE WE TESTING A DR11W OR A DR11B?  
:*****  
                  SCOPE  
                  CLR       ICOUNT  
                  CLR       @CSR               ;FORCE TO BE CSR #1  
                  BIS       #BIT15,@CSR  
                  MOV       @CSR,BORW  
                  BIT       #BIT0,BORW       ;ARE WE A B OR A W?  
                  BEQ       1$               ;WE ARE A B  
                  TYPE       ,MSG9  
                  BR        T3  
1$:               TYPE       ,MSG8
```

```

974
975
976
977
978 002414 000004
979 002416 005077 176446
980 002422 012737 000010 014122
981 002430 012777 177777 174426
982 002436 004737 014144
983 002442 052777 010000 176420
984 002450 042777 010000 176412
985 002456 005777 176402
986 002462 001401
987 002464 104000
  
```

```

          .SBTTL TEST 3 THAT DEVICE INIT CLEARS WCR
          :*****
          : TEST 3 THAT DEVICE INIT CLEARS WCR
          :*****
T3:      SCOPE
          CLR      @CSR      ;FORCE TO BE CSR #1
          MOV      #10,IOUNT
          MOV      #-1,@WCR  ;ALL ONES TO WCR
          JSR      PC,CKSWR
          BIS      #BIT12,@CSR ;DEVICE
          BIC      #BIT12,@CSR ;INIT
          TST      @WCR      ;IS Z BIT SET?
          BEQ      .+4       ;DID WCR GET CLEARED
          HLT
          ;WCR NOT CLEAR
  
```

```
989  
990  
991  
992  
993 002466 000004  
994 002470 005077 176374  
995 002474 012777 177777 176364  
996 002502 004737 014144  
997 002506 052777 010000 176354  
998 002514 042777 010000 176346  
999 002522 005777 176340  
1000 002526 001401  
1001 002530 104000
```

```
          .SBTTL TEST 4 THAT DEVICE INIT CLEARS BAR  
:*****  
:          TEST 4 THAT DEVICE INIT CLEARS BAR  
:*****  
          SCOPE  
          CLR       @CSR               :FORCE TO BE CSR #1  
          MOV       #-1,@BAR         :ALL ONES TO BAR  
          JSR       PC,CKSWR  
          BIS       #BIT12,@CSR       :DEVICE  
          BIC       #BIT12,@CSR       :INIT  
          TST       @BAR             :IS Z BIT SET?  
          BEQ       .+4               : DID BAR GET CLEARED  
          HLT
```


1003
1004
1005
1006
1007 002532 000004
1008 002534 005077 176330
1009 002540 012777 010000 176322
1010 002546 012737 000010 014122
1011 002554 012777 177777 176310
1012 002562 004737 014144
1013 002566 042777 010000 176274
1014 002574 012777 010000 176266
1015 002602 005777 176264
1016 002606 001401
1017 002610 104000
1018
1019

```
.SBTTL TEST 5 THAT DEVICE INIT CLEARS BDR  
:*****  
:TEST 5 THAT DEVICE INIT CLEARS BDR  
:*****  
SCOPE  
CLP @CSR ;FORCE TO BE CSR #1  
MOV #BIT12,@CSR ;MAINT MODE  
MOV #10,ICOUNT  
MOV #-1,@BDR ;ALL ONES TO BDR  
JSR PC,CKSWR  
BIC #BIT12,@CSR ;DEVICE INIT  
MOV #BIT12,@CSR  
TST @BDR ;CHECK FOR ZERO  
BEQ .+4  
HLT
```

```

1021                                     .SBTTL TEST 6 THAT DEVICE INIT CLEARS ALL R/W BITS IN THE CSR
1022                                     :*****
1023                                     :TEST 6 THAT DEVICE INIT CLEARS ALL R/W BITS IN THE CSR
1024                                     :*****
1025 002612 000004                                     SCOPE
1026 002614 012777 010000 176246                     MOV    #BIT12,@CSR      ;MAINT MODE
1027 002622 012737 000010 014122                     MOV    #10,ICOUNT
1028 002630 052777 010576 176232                     BIS    #10576,@CSR      ;SET MAINT(12),CYCLE(08),IE(06)XBA17(05)
1029                                             ;XBA16(04),FNCT03(03),FNCT02(02),FNCT1(01)
1030 002636 004737 014144                                     JSR    PC,CKSWR
1031 002642 042777 010000 176220                     BIC    #BIT12,@CSR      ;DEVICE INIT
1032 002650 012777 010000 176212                     MOV    #BIT12,@CSR      ;MAINT MODE
1033 002656 022777 010200 176204                     CMP    #10200,@CSR      ;TEST FOR ZERO'S,AND READY BIT SET
1034 002664 001401                                     BEQ    .+4               ;WERE ALL R/W BITS CLEARED
1035 002666 104000                                     HLT
    
```

1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053

```
.SBTTL TEST 7 DOES RESET CLEAR WCR?  
:*****  
: TEST 7 DOES RESET CLEAR WCR?  
:*****
```

```
1042 002670 000004  
1043 002672 005077 176172  
1044 002676 012737 000010 014122  
1045 002704 012777 177777 176152  
1046 002712 004737 014144  
1047 002716 000005  
1048 002720 005777 176140  
1049 002724 001401  
1050 002726 104000
```

```
SCOPE  
CLR @CSR ;FORCE TO BE CSR #1  
MOV #10,ICOUNT  
MOV #-1,@WCR ;ALL ONES TO WCR  
JSR PC,CKSWR  
RESET ;INIT  
TST @WCR ;LOOKING FOR Z-BIT TO SET  
BEQ .+4 ;DID WCR GET CLEARED?  
HLT ;WCR NOT CLEAR
```

1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066

002730 000004
002732 005077 176132
002736 012777 177777 176120
002744 022777 177777 176112
002752 001401
002754 104000

.SBTTL TEST 10 CAN ALL WCR BITS BE SET?
:*****
: TEST 10 CAN ALL WCR BITS BE SET?
:*****

SCOPE
CLR @CSR ;FORCE TO BE CSR #1
MOV #-1,@WCR ;SET ALL BITS IN WCR
CMP #-1,@WCR ;LOOKING FOR Z-BIT TO SET
BEQ .+4 ;SEE IF ALL BITS GOT SET
HLT ;ALL BITS AREN'T SET

1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085

002756 000004
002760 005077 176104
002764 012777 177777 176074
002772 004737 014144
002776 000005
003000 005777 176062
003004 001401
003006 104000

.SBTTL TEST 11 DOES RESET CLEAR BAR?
:*****
: TEST 11 DOES RESET CLEAR BAR?
:*****

SCOPE
CLR @CSR ;FORCE TO BE CSR #1
MOV #-1,@BAR ;ALL ONES TO BAR
JSR PC,CKSWR
RESET ;INIT
TST @BAR ;LOOKING FOR Z-BIT TO SET
BEQ .+4 ;DID BAR GET CLEARED?
HLT ;BAR NOT CLEAR

1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099

.SBTTL TEST 12 CAN BITS 15-01 IN BAR BE SET?
:*****
: TEST 12 CAN BITS 15-01 IN BAR BE SET?
:*****

003010 000004
003012 012777 177776 176046
003020 022777 177776 176040
003026 001401
003030 104000

SCOPE
MOV #-2,@BAR ;SET BITS 15-01 IN BAR
CMP #-2,@BAR ;LOOKING FOR Z-BIT TO SET
BEQ .+4 ;SEE IF BITS 15-01 GOT SET

HLT

1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121

.SBTTL TEST 13 TEST THAT FNCT1 CAN BE SET AND CLEARED
:*****
:TEST 13 TEST THAT FNCT1 CAN BE SET AND CLEARED
:*****

1107	003032	000004			SCOPE		
1108	003034	005077	176030		CLR	@CSR	;FORCE TO BE CSR #1
1109	003040	012777	010000	176022	MOV	#BIT12,@CSR	;SET MAINT MODE
1110	003046	012737	002000	014122	MOV	#2000,ICOUNT	
1111	003054	052777	000002	176006	BIS	#BIT1,@CSR	;SET FNCT1
1112	003062	032777	000002	176000	BIT	#BIT1,@CSR	;TEST FNCT1
1113	003070	001001			BNE	+.4	;IS IT SET?
1114	003072	104000			HLT		;FNCT1 IS CLEAR
1115	003074	042777	000002	175766	BIC	#BIT1,@CSR	;CLEAR FNCT1
1116	003102	032777	000002	175760	BIT	#BIT1,@CSR	;TEST FNCT1
1117	003110	001401			SEQ	+.4	;WAS IT CLEAR
1118	003112	104000			HLT		;FNCT1 WAS SET

1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142

.SBTTL TEST 14 TEST THAT FNCT2 CAN BE SET AND CLEARED
:*****
:TEST 14 TEST THAT FNCT2 CAN BE SET AND CLEARED
:*****

003114	000004			SCOPE		
003116	005077	175746		CLR	@CSR	:FORCE TO BE CSR #1
003122	012777	010000	175740	MOV	#BIT12,@CSR	:SET MAINT MODE
003130	052777	000004	175732	BIS	#BIT2,@CSR	:SET FNCT2
003136	032777	000004	175724	BIT	#BIT2,@CSR	:TEST FNCT2
003144	001001			BNE	+.4	:IS IT SET?
003146	104000			HLT		:FNCT2 IS CLEAR
003150	042777	000004	175712	BIC	#BIT2,@CSR	:CLEAR FNCT2
003156	032777	000004	175704	BIT	#BIT2,@CSR	:TEST FNCT2
003164	001401			BEQ	+.4	:WAS IT CLEAR?
003166	104000			HLT		:FNCT2 WAS SET

1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161

.SBTTL TEST 15 TEST THAT FNCT3 CAN BE SET AND CLEARED
:*****
: TEST 15 TEST THAT FNCT3 CAN BE SET AND CLEARED
:*****

003170	000004			SCOPE		
003172	005077	175672		CLR	@CSR	:FORCE TO BE CSR #1
003176	012777	010000	175664	MOV	#BIT12,@CSR	:SET MAINT MODE
003204	052777	000010	175656	BIS	#BIT3,@CSR	:SET FNCT3
003212	032777	000010	175650	BIT	#BIT3,@CSR	:TEST FNCT3
003220	001001			BNE	+.4	:IS IT SET?
003222	104000			HLT		:FNCT3 IS CLEAR
003224	042777	000010	175636	BIC	#BIT3,@CSR	:CLEAR FNCT3
003232	032777	000010	175630	BIT	#BIT3,@CSR	:TEST FNCT3
003240	001401			BEQ	+.4	:WAS IT CLEAR?
003242	104000			HLT		:FNCT3 WAS SET

1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178

.SBTTL TEST 16 TEST THAT XBA16 CAN BE SET AND CLEARED

: TEST 16 TEST THAT XBA16 CAN BE SET AND CLEARED

003244	000004			SCOPE		
003246	005077	175616		CLR	@CSR	:FORCE TO BE CSR #1
003252	012777	010000	175610	MOV	#BIT12,@CSR	:SET MAINT MODE
003260	052777	000020	175602	BIS	#BIT4,@CSR	:SET XBA16
003266	032777	000020	175574	BIT	#BIT4,@CSR	:TEST XBA16
003274	001001			BNE	+.4	:IS IT SET?
003276	104000			HLT		:XBA16 IS CLEAR
003300	042777	000020	175562	BIC	#BIT4,@CSR	:CLEAR XBA16
003306	032777	000020	175554	BIT	#BIT4,@CSR	:TEST XBA16
003314	001401			BEQ	+.4	:IS IT CLEAR
003316	104000			HLT		:XBA16 WAS SET

1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195

003320 000004
003322 005077 175542
003326 012777 010000 175534
003334 052777 000040 175526
003342 032777 000040 175520
003350 001001
003352 104000
003354 042777 000040 175506
003362 032777 000040 175500
003370 001401
003372 104000

```
.SBTTL TEST 17 TEST THAT XBA17 CAN BE SET AND CLEARED  
:*****  
:TEST 17 TEST THAT XBA17 CAN BE SET AND CLEARED  
:*****  
SCOPE  
CLR @CSR :FORCE TO BE CSR #1  
MOV #BIT12,@CSR :SET MAINT MODE  
BIS #BIT5,@CSR :SET XBA17  
BIT #BIT5,@CSR :TEST XBA17  
BNE .+4 :IS IT SET?  
HLT :XBA17 IS CLEAR  
BIC #BIT5,@CSR :CLEAR XBA17  
BIT #BIT5,@CSR :TEST XBA17  
BEQ .+4 :IS IT CLEAR?  
HLT :XBA17 WAS SET
```

1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212

.SBTTL TEST 20 TEST THAT IE CAN BE SET AND CLEARED

.....
:.....
TEST 20 TEST THAT IE CAN BE SET AND CLEARED
:.....
:.....

SCOPE
CLR @CSR :FORCE TO BE CSR #1
MOV #BIT12,@CSR :SET MAINT MODE
BIS #BIT6,@CSR :SET IE
BIT #BIT6,@CSR :TEST IE
BNE .+4 :IS IT SET?
HLT :IE IS CLEAR
BIC #BIT6,@CSR :CLEAR IE
BIT #BIT6,@CSR :TEST IE
BEQ .+4 :IS IT CLEAR?
HLT :IE WAS SET

```
1214 .SBTTL TEST 21 TEST THAT CYCLE CAN BE SET AND CLEARED
1215 :*****
1216 :TEST 21 TEST THAT CYCLE CAN BE SET AND CLEARED
1217 :*****
1218 003450 000004 SCOPE
1219 003452 005077 175412 CLR @CSR :FORCE TO BE CSR #1
1220 003456 012777 010000 175404 MOV #BIT12,@CSR :SET MAINT MODE
1221 003464 052777 000400 175376 BIS #BIT8,@CSR :SET CYCLE
1222 003472 032777 000400 175370 BIT #BIT8,@CSR :TEST CYCLE
1223 003500 001001 BNE .+4 :IS IT SET?
1224 003502 104000 HLT :CYCLE WAS CLEAR
1225 003504 042777 000400 175356 BIC #BIT8,@CSR :CLEAR CYCLE
1226 003512 032777 000400 175350 BIT #BIT8,@CSR :TEST CYCLE
1227 003520 001401 BEQ .+4 :IS IT CLEAR?
1228 003522 104000 HLT :CYCLE WAS SET
1229
```

1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248

003524 000004
003526 005077 175336
003532 012777 010000 175330
003540 032777 010000 175322
003546 001001
003550 104000
003552 042777 010000 175310
003560 032777 010000 175302
003566 001401
003570 104000
003572 001401
003574 104000

```
.SBTTL TEST 22 TEST THAT MAINT CAN BE SET AND CLEARED  
:*****  
: TEST 22 TEST THAT MAINT CAN BE SET AND CLEARED  
:*****  
SCOPE  
CLR @CSR :FORCE TO BE CSR #1  
MOV #BIT12,@CSR :SET MAINT  
BIT #BIT12,@CSR :TEST MAINT  
BNE .+4 :IS IT SET?  
HLT :MAINT WAS CLEAR  
BIC #BIT12,@CSR :CLEAR MAINT  
BIT #BIT12,@CSR :TEST MAINT  
BEQ .+4 :IS MAINT CLEAR?  
HLT :MAINT WAS SET  
  
BEQ .+4 :IS IT CLEAR?  
HLT :CYCLE WAS SET
```

```

1250 .SBTTL TEST 23 THAT ALL CSR R/W BITS CAN BE SET AND CLEARED
1251 *****
1252 TEST 23 TEST THAT ALL CSR R/W BITS CAN BE SET AND CLEARED
1253 MAINT MODE (INTERNAL WRAP-AROUND)
1254 *****
1255 003576 000004 SCOPE
1256 003600 005077 175264 CLR @CSR ;FORCE TO BE CSR #1
1257 003604 012777 010000 175256 MOV #BIT12,@CSR ;MAINT MODE
1258 003612 052777 020576 175250 BIS #20576,@CSR ;SET FOLLOWING:MAINT(12),CYCLE(08),IE(06),XBA17(05),
; XBA16(04),FNCT3(03),FUNCT2(02),FNCT1(01)
1259
1260 003620 032777 000002 175242 BIT #BIT1,@CSR ;TEST FNCT1
1261 003626 001001 BNE .+4 ;IS IT SET?
1262 003630 104000 HLT ;FNCT1 IS CLEAR
1263 003632 032777 000004 175230 BIT #BIT2,@CSR ;TEST FNCT2
1264 003640 001001 BNE .+4 ;IS IT SET?
1265 003642 104000 HLT ;FNCT2 IS CLEAR
1266 003644 032777 000010 175216 BIT #BIT3,@CSR ;TEST FNCT3
1267 003652 001001 BNE .+4 ;IS IT SET?
1268 003654 104000 HLT ;FNCT3 IS CLEAR
1269 003656 032777 000020 175204 BIT #BIT4,@CSR ;TEST XBA16
1270 003664 001001 BNE .+4 ;IS IT SET?
1271 003666 104000 HLT ;XBA16 IS CLEAR
1272 003670 032777 000040 175172 BIT #BIT5,@CSR ;TEST XBA17
1273 003676 001001 BNE .+4 ;IS IT SET?
1274 003700 104000 HLT ;XBA17 IS CLEAR
1275 003702 032777 000100 175160 BIT #BIT6,@CSR ;TEST IE
1276 003710 001001 BNE .+4 ;IS IT SET?
1277 003712 104000 HLT ;IE IS CLEAR
1278 003714 032777 000400 175146 BIT #BIT8,@CSR ;TEST CYCLE
1279 003722 001001 BNE .+4 ;IS CYCLE SET?
1280 003724 104000 HLT ;CYCLE IS CLEAR
1281 003726 032777 010000 175134 BIT #BIT12,@CSR ;TEST MAINT
1282 003734 001001 BNE .+4 ;IS MAINT SET?
1283 003736 104000 HLT ;MAINT IS CLEAR
    
```

```

1285 003740 042777 030576 175122      BIC    #30576,@CSR      ;CLEAR ALL R/W BITS IN CSR
1286 003746 032777 000002 175114      BIT    #BIT1,@CSR      ;TEST FNCT1
1287 003754 001401                BEQ    .+4              ;IS FNCT1 CLEAR?
1288 003756 104000                HLT                    ;FNCT1 IS SET
1289 003760 032777 000004 175102      BIT    #BIT2,@CSR      ;TEST FNCT2
1290 003766 001401                BEQ    .+4              ;IS FNCT2 CLEAR?
1291 003770 104000                HLT                    ;FNCT2 IS SET
1292 003772 032777 000010 175070      BIT    #BIT3,@CSR      ;TEST FNCT3
1293 004000 001401                BEQ    .+4              ;IS FNCT3 CLEAR?
1294 004002 104000                HLT                    ;FNCT3 IS SET
1295 004004 032777 000020 175056      BIT    #BIT4,@CSR      ;TEST XBA16
1296 004012 001401                BEQ    .+4              ;IS XBA16 CLEAR
1297 004014 104000                HLT                    ;XBA16 IS SET
1298 004016 032777 000040 175044      BIT    #BIT5,@CSR      ;TEST XBA17
1299 004024 001401                BEQ    .+4              ;IS XBA17 CLEAR?
1300 004026 104000                HLT                    ;XBA17 IS SET
1301 004030 032777 000100 175032      BIT    #BIT6,@CSR      ;TEST IE
1302 004036 001401                BEQ    .+4              ;IS IE CLEAR?
1303 004040 104000                HLT                    ;IE IS SET
1304 004042 032777 000400 175020      BIT    #BIT8,@CSR      ;TEST CYCLE
1305 004050 001401                BEQ    .+4              ;IS CYCLE CLEAR?
1306 004052 104000                HLT                    ;CYCLE IS SET
1307 004054 032777 010000 175006      BIT    #BIT12,@CSR     ;TEST MAINT
1308 004062 001401                BEQ    .+4              ;IS MAINT CLEAR?
1309 004064 104000                HLT                    ;MAINT IS SET
1310
  
```



```

1312
1313
1314
1315
1316
1317
1318 004066 000004
1319 004070 005077 174774
1320 004074 012777 010000 174766
1321 004102 012737 000010 014122
1322 004110 052777 010576 174752
1323
1324 004116 017701 174746
1325 004122 052701 167201
1326 004126 005201
1327 004130 001401
1328 004132 104000
1329 004134 004737 014144
1330 004140 000005
1331 004142 017701 174722
1332 004146 042701 127000
1333 004152 022701 000200
1334 004156 001401
1335 004160 104000
1336

.SBTTL TEST 24 ALL R/W BITS IN CSR CAN BE SET AND RESET TO ZERO
*****
TEST 24 ALL R/W BITS IN CSR CAN BE SET AND RESET TO ZERO, THAT READY
IS SET, NEX IS CLEAR, GO IS READ AS A 0, AND ERROR IS CLEAR.
MAINT MODE (INTERNAL WRAP-AROUND)
*****
SCOPE
CLR @CSR ;FORCE TO BE CSR #1
MOV #BIT12,@CSR ;MAINT MODE
MOV #10,ICOUNT
BIS #10576,@CSR ;SET FOLLOWING: MAINT(12),CYCLE(08),IE(06),XBA17(05),
; XBA16(04),FNCT3(03),FNCT2(02),FNCT1(01)
MOV @CSR,R1 ;MOVE (CSR) TO R1
BIS #167201,R1 ;SETS BITS IN R1 THAT WERE NOT SET IN CSR
INC R1 ;R1 SHOULD GO FROM -1 TO ZERO
BEQ .+4 ;WERE ALL CSR R/W BITS SET?
HLT ;NOT ALL BITS WERE SET
JSR PC,CKSWR
RESET ;CLEAR ALL CSR R/W BITS
MOV @CSR,R1 ;MOVE (CSR) TO R1
BIC #127000,R1 ;CLEAR ERROR,ATTN,DSTATA,DSTATB,DSTATC
CMP #200,R1 ;TEST FOR READY
BEQ .+4
HLT ;RESET DIDN'T LEAVE CSR AS IT SHOULD HAVE
    
```

```

1338
1339
1340
1341
1342
1343
1344 004162 000004
1345 004164 032737 000001 001144
1346 004172 001426
1347 004174 005077 174670
1348 004200 032777 000001 174662
1349 004206 001401
1350 004210 104000
1351 004212 012777 100000 174650
1352 004220 032777 000001 174642
1353 004226 001001
1354 004230 104000
1355 004232 005077 174632
1356 004236 032777 000001 174624
1357 004244 001401
1358 004246 104000
1359

```

```

.SBTTL TEST 25 CSR AND EIR BITO TEST
:*****
:TEST 25 CSR AND EIR BITO TEST
:TEST THAT BITO OF CSR READS AS A ZERO
:TEST THAT BITO OF EIR READS AS A ONE
:*****
SCOPE
BIT    #BIT0,BORW
BEQ    T26
CLR    @CSR          ;FORCE TO BE CSR
BIT    #BIT0,@CSR    ;IS BITO A ZERO?
BEQ    .+4           ;YES IT IS A ZERO
HLT    ;BITO READ AS A ONE
MOV    #BIT15,@CSR   ;SET BIT15 TO GO TO EIR
BIT    #BIT0,@CSR    ;IS BITO A ONE?
BNE    .+4           ;YES IT IS A ONE
HLT    ;BITO READS AS A ZERO
CLR    @CSR          ;RETURN TO CSR
BIT    #BIT0,@CSR    ;DID WE GET BACK
BEQ    .+4           ;YES
HLT    ;NO WE ARE STUCK IN EIR

```

```

1361
1362
1363
1364
1365
1366
1367 004250 000004
1368 004252 032737 000001 001144
1369 004260 001444
1370 004262 012737 002000 014122
1371 004270 005077 174574
1372 004274 012777 010000 174566
1373 004302 022777 010200 174560
1374 004310 001401
1375 004312 104000
1376 004314 112777 000004 174546
1377 004322 032777 020000 174540
1378 004330 001001
1379 004332 104000
1380 004334 042777 020004 174526
1381 004342 032777 020000 174520
1382 004350 001401
1383 004352 104000
1384 004354 017705 174510
1385 004360 005705
1386 004362 100001
1387 004364 104000
1388 004366 005077 174476
  
```

```

      .SBTTL TEST 26 THAT ATTN CAN BE SET VIA F2 & ERROR SETS
      :*****
      :TEST 26 TEST THAT ATTN CAN BE SET VIA F2 & ERROR SETS;
      :TEST THAT ATTN,AND THEREFORE ERROR WILL CLEAR
      :MAINTENANCE MODE (INTERNAL WRAP-AROUND)
      :*****
T26:  SCOPE
      BIT      #BIT0,BORW
      BEQ      T27          ;IF A 'B' SKIP TEST
      MOV      #2000,ICOUNT
      CLR      @CSR        ;FORCE TO BE CSR #1
      MOV      #BIT12,@CSR ;MAINT
      CMP      #10200,@CSR ;ONLY READY AND MAINT
      BEQ      .+4
      HLT
      :SOMETHING OTHER THAN READY&MAINT
      MOVB     #BIT2,@CSR  ;SET F2
      BIT      #BIT13,@CSR ;TEST ATTN
      BNE      .+4        ;IS IT SET?
      HLT
      :ATTN WAS CLEAR
      BIC      #20004,@CSR ;CLEAR ATTN & F2
      BIT      #BIT13,@CSR ;TEST ATTN
      BEQ      .+4        ;IS ATTN CLEAR?
      HLT
      :ATTN WAS SET
      MOV      @CSR,R5    ;SAVE CSR #1
      TST      R5        ;IS ERROR CLEAR?
      BPL      .+4
      HLT
      :ERROR IS STILL SET
      CLR      @CSR      ;RETURN TO CSR #1
  
```

1392
1393
1394
1395
1396 004372 000004
1397 004374 005077 174470
1398 004400 012737 002000 014122
1399 004406 012777 052525 174450
1400 004414 022777 052525 174442
1401 004422 001401
1402 004424 104000
1403 004426 012777 125252 174430
1404 004434 022777 125252 174422
1405 004442 001401
1406 004444 104000
1407

```
.SBTTL TEST 27 CAN WCR HOLD ALTERNATE ONES AND ZEROS
:*****
:TEST 27 CAN WCR HOLD ALTERNATE ONE'S AND ZERO'S
:*****
T27: SCOPE
      CLR @CSR ;FORCE TO BE CSR #1
      MOV #2000,ICOUNT
      MOV #052525,@WCR ;ALT 0'S AND 1'S TO WCR
      CMP #052525,@WCR ;LOOKING FOR Z-BIT TO SET
      BEQ .+4 ;DOES WCR HAVE THE CORRECT PATTERN?
      HLT ;WCR DOESN'T HAVE THE CORRECT PATTERN
      MOV #125252,@WCR ;ALT 1'S AND 0'S TO WCR
      CMP #125252,@WCR ;LOOKING FOR Z-BIT TO SET
      BEQ .+4 ;DOES WCR HAVE THE CORRECT PATTERN?
      HLT ;WCR DOESN'T HAVE THE CORRECT PATTERN
```


1451
 1452
 1453
 1454
 1455 004602 000004
 1456 004604 005077 174260
 1457 004610 005037 014122
 1458 004614 005001
 1459 004616 005077 174242
 1460 004622 020177 174236
 1461 004626 001401
 1462 004630 104000
 1463 004632 005277 174226
 1464 004636 005201
 1465 004640 001370
 1466

```

.SBTTL TEST 32 INCREMENTING PATTERN TO WRAP-AROUND IN WCR
:*****
:TEST 32 INCREMENTING PATTERN TO WRAP-AROUND IN WCR
:*****
SCOPE
CLR @CSR          .FORCE TO BE CSR #1
CLR ICOUNT
CLR R1            ;SET-UP
CLR @WCR         ;SET-UP
INCWC: CMP R1,@WCR ;SEE IF THEY ARE EQUAL
      BEQ .+4      ;ARE THEY EQUAL?
      HLT          ;THEY'RE NOT EQUAL
      INC @WCR     ;GET NEXT NUMBER
      INC R1       ;GET NEXT NUMBER
      BNE INCWC    ;DONE WITH TEST? IF NOT CONTINUE
    
```

```

1468
1469
1470
1471
1472
1473 004642 000004
1474 004644 005001
1475 004646 005077 174214
1476 004652 020177 174210
1477 004656 001401
1478 004660 104000
1479 004662 062777 000002 174176
1480 004670 062701 000002
1481 004674 001366
  
```

```

      .SBTTL TEST 33 INCREMENTING PATTERN TO WRAP-AROUND IN BAR
      :*****
      :TEST 33 INCREMENTING PATTERN TO WRAP-AROUND IN BAR
      :MAINT MODE(INTERN. WRAP-AROUND)
      :*****
      SCOPE
      CLR      R1           ;SET-UP
      CLR      @BAR        ;SET-UP
      INCBA:  CMP      R1,@BAR ;SEE IF THEY ARE EQUAL
      BEQ      .+4         ;ARE THEY EQUAL?
      HLT                     ;THEY'RE NOT EQUAL
      ADD      #2,@BAR     ;GET NEXT NUMBER
      ADD      #2,R1       ;GET NEXT NUMBER
      BNE      INCBA      ;DONE WITH TEST? IF NOT CONTINUE
  
```


1483
 1484
 1485
 1486
 1487
 1488
 1489
 1490
 1491
 1492
 1493
 1494
 1495
 1496
 1497
 1498
 1499
 1500
 1501
 1502
 1503
 1504
 1505
 1506
 1507
 1508
 1509
 1510
 1511
 1512
 1513
 1514
 1515
 1516

004676 000004
 004700 005077 174164
 004704 012777 010000 174156
 004712 032777 00C016 174150
 004720 001401
 004722 104000
 004724 052777 000002 174136
 004732 032777 001000 174130
 004740 001001
 004742 104000
 004744 032777 006000 174116
 004752 001401
 004754 104000
 004756 012777 010000 174104
 004764 052777 000004 174076
 004772 032777 002000 174070
 005000 001001
 005002 104000
 005004 032777 005000 174056
 005012 001401
 005014 104000
 005016 012777 010000 174044
 005024 052777 000010 174036
 005032 032777 004000 174030
 005040 001001
 005042 104000
 005044 032777 003000 174016
 005052 001401
 005054 104000

```

.SBTTL TEST 34 TEST THAT FNCT BITS CONTROL DSTAT BITS
:*****
:TEST 34 TEST THAT FNCT BITS CONTROL DSTAT BITS
:MAINTENANCE MODE (INTERNAL WRAP-AROUND)
:*****
x5: SCOPE
    CLR @CSR ;CLR FUNCT BITS AND FORCE TO BE CSR #1
    MOV #10000,@CSR ;MAINT MODE
    BIT #16,@CSR ;CHECK FUNCTION BITS
    BEQ .+4 ;FUNCTION BITS CLEAR?
    HLT ;FUNCTION BITS NOT CLEAR
    BIS #BIT1,@CSR ;SET FNCT1
    BIT #BIT9,@CSR ;CHECK DSTAT C
    BNE .+4 ;IS IT SET?
    HLT ;DSTAT C IS CLEAR
    BIT #6000,@CSR ;CHECK THAT DSTAT A AND DSTAT B ARE CLEAR
    BEQ .+4 ;ARE THEY CLEAR?
    HLT ;DSTAT A AND/OR DSTAT B IS SET
    MOV #10000,@CSR ;MAINT MODE
    BIS #BIT2,@CSR ;SET FNCT2
    BIT #BIT10,@CSR ;CHECK DSTAT B
    BNE .+4 ;IS IT SET?
    HLT ;DSTAT B IS CLEAR
    BIT #5000,@CSR ;CHECK THAT DSTAT A AND DSTAT C ARE CLEAR
    BEQ .+4 ;ARE THEY CLEAR?
    HLT ;DSTAT A AND/OR DSTAT B IS SET
    MOV #10000,@CSR ;MAINT MODE
    BIS #BIT3,@CSR ;SET FNCT3
    BIT #BIT11,@CSR ;CHECK DSTAT A
    BNE .+4 ;IS IT SET?
    HLT ;DSTAT A IS CLEAR
    BIT #3000,@CSR ;CHECK THAT DSTAT B AND DSTAT C ARE CLEAR
    BEQ .+4 ;ARE THEY CLEAR?
    HLT ;DSTAT B AND/OR DSTAT C IS SET
    
```

```

1518
1519
1520
1521
1522
1523 005056 000004
1524 005060 005077 174004
1525 005064 012737 000010 014122
1526 005072 012777 177777 173772
1527 005100 004737 014144
1528 005104 000005
1529 005106 052777 010000 173754
1530 005114 005777 173752
1531 005120 001401
1532 005122 104000
    
```

```

.SBTTL TEST 35 TEST THAT RESET CLEARS BDR
*****
TEST 35 TEST THAT RESET CLEARS BDR
MAINTENANCE MODE (INTERNAL WRAP-AROUND)
*****
SCOPE
CLR @CSR ;FORCE TO BE CSR #1
MOV #10,ICOUNT
MOV #-1,@BDR ;ALL ONES TO BDR
JSR PC,CKSWR
RESET ;INIT
BIS #10000,@CSR ;MAINT MODE
TST @BDR ;LOOKING FOR Z-BIT TO SET
BEQ .+4 ;DID BDR GET CLEARED?
HLT ;BDR NOT CLEAR
    
```

1534
1535
1536
1537
1538
1539 005124 000004
1540 005126 005077 173736
1541 005132 012737 002000 014122
1542 005140 012777 010000 173722
1543 005146 012777 177777 173716
1544 005154 022777 177777 173710
1545 005162 001401
1546 005164 104000
1547

```
.SBTTL TEST 36 TEST THAT ALL BDR BITS CAN BE SET  
:*****  
:TEST 36 TEST THAT ALL BDR BITS CAN BE SET  
:MAINTENANCE MODE (INTERNAL WRAP-AROUND)  
:*****  
SCOPE  
CLR @CSR ;FORCE TO BE CSR #1  
MOV #2000,I COUNT  
MOV #10000,@CSR ;MAINT MODE  
MOV #-1,@BDR ;SET ALL BITS IN BDR  
CMP #-1,@BDR ;LOOKING FOR Z-BIT TO SET  
BEQ .+4 ;SEE IF ALL BITS GOT SET  
HLT ;ALL BDR BITS AREN'T SET
```

1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565

005166 000004
005170 005077 173674
005174 012777 010000 173666
005202 012777 052525 173662
005210 022777 052525 173654
005216 001401
005220 104000
005222 012777 125252 173642
005230 022777 125252 173634
005236 001401
005240 104000

```
.SBTTL TEST 37 TEST THAT BDR CAN HOLD ALTERNATE ONES AND ZEROS
*****
TEST 37 TEST THAT BDR CAN HOLD ALTERNATE ONE'S AND ZERO'S
MAINTENANCE MODE (INTERNAL WRAP-AROUND)
*****
SCOPE
CLR @CSR ;FORCE TO BE CSR #1
MOV #10000,@CSR ;MAINT MODE
MOV #052525,@BDR ;ALT 0'S AND 1'S TO BDR
CMP #052525,@BDR ;LOOKING FOR Z-BIT TO SET
BEQ .+4 ;DOES BDR HAVE THE CORRECT PATTERN?
HLT ;BDR IS WRONG
MOV #125252,@BDR ;ALT 1'S AND 0'S TO BDR
CMP #125252,@BDR ;LOOKING FOR Z-BIT TO SET
BEQ .+4 ;DOES BDR HAVE THE CORRECT PATTERN?
HLT ;BDR IS WRONG
```

```

1567 .SBTTL TEST 40 THAT GOING TO EIR BLOCKS DATA XFERS FROM ODR TO IDR
1568 :*****
1569 :TEST 40 TEST THAT GOING TO EIR BLOCKS DATA TRANSFER FROM
1570 :ODR TO IDR
1571 :*****
1572 005242 000004 SCOPE
1573 005244 032737 000001 001144 BIT #BIT0,BORW
1574 005252 001442 BEQ T41
1575 005254 005077 173610 CLR @CSR ;FORCE TO BE CSR
1576 005260 012737 002000 014122 MOV #2000,ICOUNT ;
1577 005266 012777 010000 173574 MOV #BIT12,@CSR ;SET MAINT MODE
1578 005274 012777 052525 173570 MOV #52525,@BDR ;SET ALT 0'S AND 1'S TO BDR
1579 005302 022777 052525 173562 CMP #52525,@BDR ;LOOK FOR CORRECT PATTERN
1580 005310 001401 BEQ .+4 ;BDR IS GOOD
1581 005312 104000 HLT ;BDR IS WRONG
1582 005314 012777 110000 173546 MOV #110000,@CSR ;GO TO EIR
1583 005322 012777 125252 173542 MOV #125252,@BDR ;SET ALT 1'S AND 0'S TO BDR
1584 005330 022777 125252 173534 CMP #125252,@BDR ;CHECK PATTERN
1585 005336 001001 BNF .+4 ;SHOULD NOT CHECK OUT - OK
1586 005340 104000 HLT ;DATA SHOULD NOT HAVE GOTTEN THROUGH
1587 005342 022777 052525 173522 CMP #52525,@BDR ;TEST FOR OLD PATTERN
1588 005350 001401 BEQ .+4 ;ALL IS FINE GO TO NEXT TEST
1589 005352 104000 HLT ;BDR IS WRONG
1590 005354 005077 173510 CLR @CSR ;GO BACK TO CSR
1591
    
```

1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612

005360 000004
005362 005077 173502
005366 012777 010000 173474
005374 005037 014122
005400 005001
005402 005077 173464
005406 020177 173460
005412 001401
005414 104000
005416 005277 173450
005422 005201
005424 001370

```
.SBTTL TEST 41 INCREMENTING PATTERN TO WRAP-AROUND IN BDR
:*****
:TEST 41 INCREMENTING PATTERN TO WRAP-AROUND IN BDR
:MAINTENANCE MODE (INTERNAL WRAP-AROUND)
:*****
T41: SCOPE
    CLR @CSR ;FORCE TO BE CSR #1
    MOV #10000,@CSR ;MAINT MODE
    CLR !COUNT
    CLR R1 ;SET-UP
    CLR @BDR ;SET-UP
INCD8: CMP R1,@BDR ;SEE IF THEY ARE EQUAL
    BEQ .+4 ;ARE THEY EQUAL?
    HLT ;THEY'RE NOT EQUAL
    INC @BDR ;GET NEXT NUMBER
    INC R1 ;GET NEXT NUMBER
    BNE INCD8 ;DONE WITH TEST? IF NOT CONTINUE
```

```

1614
1615
1616
1617
1618
1619 005426 000004
1620 005430 005077 173434
1621 005434 012737 002000 014122
1622 005442 012777 010000 173420
1623 005450 022777 010200 173412
1624 005456 001401
1625 005460 104000
1626 005462 012777 177600 173374
1627 005470 013777 001152 173370
1628 005476 052777 000010 173364
1629 005504 052777 000001 173356
1630 005512 105777 173352
1631 005516 100001
1632 005520 104000
1633 005522 052777 000400 173340
1634 005530 005037 001166
1635 005534 105777 173330
1636 005540 100406
1637 005542 062737 000004 001166
1638 005550 100401
1639 005552 000770
1640 005554 104000
1641 005556 000240
    
```

```

.SBTTL TEST 42 TEST THAT GO CLEARS READY AND SETS AGAIN
*****
TEST 42 TEST THAT GO CLEARS READY AND SETS AGAIN
MAINTENANCE MODE (INTERNAL WRAP-AROUND)
*****
SCOPE
CLR @CSR ;FORCE TO BE CSR #1
MOV #2000,I COUNT
MOV #10000,@CSR ;MAINT MODE
CMP #10200,@CSR ;ONLY READY
SEC .+4
HLT
MOV #-200,@WCR ;SET-UP WCR
MOV INBUF,@BAR ;SET-UP BAR
BIS #10,@CSR ;FNCT3 (NON BURST)
BIS #1,@CSR ;GO CLEARS READY
TSTB @CSR ;CHECK READY
BPL .+4 ;IS READY CLEAR?
HLT ;READY IS STILL SET
BIS #400,@CSR ;SET CYCLE AND START XFERS
CLR RDYCHK ;CLEAR READY CHECK
TSTRDY: TSTB @CSR ;CHECK READY
BMI DONE ;IF SET GO TO DONE
ADD #4,RDYCHK ;CHECKING TIME FOR READY TO BE SET
BMI .+4 ;IF RDYCHK GETS NEGATIVE IT TOOK TOO LONG
BR TSTRDY ;CHECK AGAIN
HLT ;READY GOT CLEARED BUT NEVER SET AGAIN
DONE: NOP ;GO TO NEXT TEST
    
```

```

1644 .SBTTL TEST 43 TEST THAT DR11W DOES INTERRUPT WITH PROC AT LEVEL 3
1645 *****
1646 TEST 43 TEST THAT DR11-W DOES INTERRUPT WITH PROC AT LEVEL 3
1647 MAINTENANCE MODE (INTERNAL WRAP-AROUND)
1648 INTERRUPT VIA ATTN
1649 *****
1650 005560 000004          X4: SCOPE
1651 005562 005077 173302 CLR @CSR ;FORCE TO BE CSR #1
1652 005566 012777 000140 173206 MOV #140,@PSW ;STATUS AT LEVEL 3
1653 005574 032777 000200 173266 BIT #BIT7,@CSR ;CHECK READY BIT
1654 005602 001010 BNE P3INV ;IS IT SET?
1655 005604 004737 014144 JSR PC,CKSWR
1656 005610 000005 RESET ;INIT TO SET READY
1657 005612 032777 000200 173250 BIT #BIT7,@CSR ;SEE IF READY IS SET NOW
1658 005620 001001 BNE .+4 ;IS IT SET
1659 005622 104000 HLT ;READY CAN'T BE SET BY INIT
1660 005624 012777 005720 173304 P3INV: MOV #P3INT,@DRINV ;SET UP INT VECTOR
1661 005632 012777 000340 173300 MOV #340,@DRVS
1662 005640 012777 010000 173222 MOV #10000,@CSR ;MAINT MODE
1663 005646 052777 000101 173214 BIS #101,@CSR ;IE,GO
1664 005654 052777 020004 173206 BIS #20004,@CSR ;SET ATTN;ERROR SETS,SETS READY,AND INTERRUPT OCCURS
1665 005662 012737 001000 001312 MOV #1000,TIME ;DELAY
1666 005670 005337 001312 22$: DEC TIME
1667 005674 001375 BNE 22$
1668 005676 013777 001140 173232 MOV DRVS,@DRINV ;RESTORE INTERRUPT VECTOR
1669 005704 005077 173230 CLR @DRVS ;RESTORE TRAP CATCHER
1670 005710 104000 HLT ;DR11-W DIDN'T INTERRUPT
1671 005712 000005 RESET ;IF NO INTERRUPT,CLEAR ANY OUTSTANDING
1672 ;INTERRUPT REQUEST IN BOARD HDWR.
1673 005714 000137 005740 JMP X7
1674 005720 005077 173144 P3INT: CLR @CSR ;CLEAR IE
1675 005724 022626 CMP (SP)+,(SP)+ ;REPOSITION THE STACK AFTER AN INTERRUPT
1676 005726 013777 001140 173202 MOV DRVS,@DRINV ;RESTORE INTERRUPT VECTOR
1677 005734 005077 173200 CLR @DRVS ;RESTORE TRAP CATCHER
1678
1679
    
```



```

1682 .SBTTL TEST 44 THAT DR11W DOES NOT INTERRUPT WITH PROC AT LEVEL 7
1683 :*****
1684 :TEST 44 TEST THAT DR11-W DOES NOT INTERRUPT WITH PROC AT LEVEL 7
1685 :MAINTENANCE MODE (INTERNAL WRAP-AROUND)
1686 :INTERRUPT VIA ATTN
1687 :*****
1688 005740 000004 X7: SCOPE
1689 005742 005077 173122 CLR @CSR ;FORCE TO BE CSR #1
1690 005746 005037 014122 CLR ICOUNT
1691 005752 012777 000340 173022 MOV #340,@PSW ;STATUS AT LEVEL 7
1692 005760 032777 000200 173102 BIT #BIT7,@CSR ;CHECK READY BIT
1693 005766 001010 BNE P7INV ;IS IT SET
1694 005770 004737 014144 JSR PC,CKSWR
1695 005774 000005 RESET ;INIT TO SET READY
1696 005776 032777 000200 173064 BIT #BIT7,@CSR ;SEE IF READY IS SET NOW
1697 006004 001001 BNE .+4 ;IS READY SET?
1698 006006 104000 HLT ;READY CAN'T BE SET BY INIT
1699 006010 012777 006100 173120 P7INV: MOV #P7ERR,@DRINV ;SET UP INT VECTOR
1700 006016 012777 000340 173114 MOV #340,@DRVS
1701 006024 012777 010000 173036 MOV #10000,@CSR ;MAINT MODE
1702 006032 052777 000101 173030 BIS #101,@CSR ;IE,GO
1703 006040 052777 020004 173022 BIS #20004,@CSR ;SET ATTN;ERROR SETS,SETS READY,AND INTERRUPT OCCURS
1704 006046 012737 001000 001312 MOV #1000,TIME ;DELAY
1705 006054 005337 001312 22$: DEC TIME
1706 006060 001375 BNE 22$
1707 006062 000005 RESET ;CLR IE,OUTSTANDING INTERR RFQUEST
1708 006064 013777 001140 173044 MOV DRVS,@DRINV
1709 006072 005077 173042 CLR @DRVS
1710 006076 000411 BR X1
1711 006100 022626 P7ERR: CMP (SP)+,(SP)+ ;RESTORE STACK
1712 006102 005077 172762 CLR @CSR ;CLEAR IE
1713 006106 013777 001140 173022 MOV DRVS,@DRINV
1714 006114 005077 173020 CLR @DRVS
1715 006120 104000 HLT ;DR11-W INTERRUPTED, BUT IT SHOULDN'T HAVE
1716
    
```

```

1718 .SBTTL TEST 45 THAT DR11W DOES INTERRUPT AND INDICATE BR LEVEL
1719 :*****
1720 :TEST 45 THAT DR11W DOES INTERRUPT AND INDICATE BR LEVEL
1721 :MAINTENANCE MODE (INTERNAL WRAP-AROUND)
1722 :INTERRUPT VIA ATTN
1723 :*****
1724 006122 000004 X1: SCOPE
1725 006124 012777 000340 172650 MOV #340,@PSW ;STATUS AT LEVEL 7
1726 006132 005077 172732 NSTAT: CLR @CSR ;FORCE TO BE CSR #1
1727 006136 032777 000200 172724 BIT #BIT7,@CSR ;CHECK READY BIT
1728 006144 001010 BNE PINV ;IS IT SET
1729 006146 004737 014144 JSR PC,CKSWR
1730 006152 000005 RESET ;INIT TO SET READY
1731 006154 032777 000200 172706 BIT #BIT7,@CSR ;SEE IF READY IS SET NOW
1732 006162 001001 BNE .+4 ;IS READY SET?
1733 006164 104000 HLT ;READY CAN'T BE SET BY INIT
1734 006166 012777 006272 172742 PINV: MOV #BRLEV,@DRINV ;SET UP INT VECTOR
1735 006174 012777 000340 172736 MOV #340,@DRVS
1736 006202 012777 010000 172660 MOV #10000,@CSR ;MAINT MODE
1737 006210 052777 000101 172652 BIS #101,@CSR ;IE,GO
1738 006216 052777 020004 172644 BIS #20004,@CSR ;SET ATTN;ERROR SETS,SETS READY,AND INTERRUPT OCCURS
1739 006224 012737 001000 001312 MOV #1000,TIME ;DELAY
1740 006232 005337 001312 22$: DEC TIME
1741 006236 001375 BNE 22$
1742 006240 000005 RESET ;CLR IE,OUTSTANDING INTERR REQUEST
1743 006242 013777 001140 172666 MOV DRVS,@DRINV
1744 006250 005077 172664 CLR @DRVS
1745 006254 017737 172522 023412 MOV @PSW,LEVEL ;SAVE OLD STATUS LEVEL
1746 006262 162777 000040 172512 SUB #40,@PSW ;NEW PSW
1747 006270 000720 BR NSTAT ;TRY AGAIN
1748 006272 022626 BRLEV: CMP (SP)+,(SP)+ ;RESTORE STACK
1749 006274 005077 172570 CLR @CSR ;CLEAR IE
1750 006300 013777 001140 172630 MOV DRVS,@DRINV
1751 006306 005077 172626 CLR @DRVS
1752 006312 042737 177437 023412 BIC #177437,LEVEL ;GET
1753 006320 013701 023412 MOV LEVEL,R1 ;BR
1754 006324 006201 ASR R1 ;INTERRUPT
1755 006326 006201 ASR R1 ;LEVEL
1756 006330 006201 ASR R1
1757 006332 006201 ASR R1
1758 006334 006201 ASR R1
1759 006336 010137 023412 MOV R1,LEVEL
1760 006342 104401 001351 TYPE .$CRLF
1761 006346 104401 001351 TYPE .$CRLF
1762 006352 104401 006360 TYPE ,65$ ;:TYPE ASCIZ STRING
006356 000421 BR 64$ ;:GET OVER THE ASCIZ
;:65$: .ASCIZ /TEST 40: DR11W INTERRUPT LEVEL =/
64$: MOV LEVEL,-(SP) ;:SAVE LEVEL FOR TYPEOUT
006422 013746 023412 TYPDS ;:GO TYPE--DECIMAL ASCII WITH SIGN
006426 104405
1764 006430 104401 001351 TYPE .$CRLF
1765 006434 104401 001351 TYPE .$CRLF
1766
    
```

```

1769 .SBTTL TEST 46 THAT GIVING A GO WITHOUT CLEARING ERROR CAUSES INTRPT
1770 :*****
1771 :TEST 46 TEST THAT GIVING A GO WITHOUT CLEARING A PREVIOUS
1772 :ERROR CAUSES ANOTHER INTERRUPT
1773 :FOR THIS TEST, THE NORMAL INTERRUPT MECHANISM SHOULD BE WORKING
1774 :MAINTENANCE MODE (INTERNAL WRAP-AROUND)
1775 :*****
1776 006440 000004          T55: SCOPE
1777 006442 005077 172422   CLR @CSR ;FORCE TO BE CSR #1
1778 006446 012777 006534 172462   MOV #ERRDO,@DRINV ;INTERRUPT VECTOR TO ERRDO
1779 006454 012777 000140 172456   MOV #140,@DRVS ;INTERRUPT STATUS TO LEVEL 4
1780 006462 005077 172314   CLR @PSW ;LET THE DR11-W INTERRUPT
1781 006466 012777 010000 172374   MOV #10000,@CSR ;MAINT MODE
1782 006474 052777 000101 172366   BIS #101,@CSR ;IE GO
1783 006502 052777 020004 172360   BIS #20004,@CSR ;SET ATTN AND INTERRUPT
1784 006510 012737 001000 001312   MOV #1000,TIME ;DELAY
1785 006516 005337 001312   22$: DEC TIME
1786 006522 001375          BNE 22$
1787 006524 104000          HLT ;NO DR11-W INTERRUPT
1788 006526 000005          RESET ;CLEAR OUTSTANDING INTERR. REQST.
1789 006530 000137 006632   JMP T60
1790 006534 022626          ERRDO: CMP (SP)+,(SP)+ ;READJUST STACK
1791 006536 005777 172326   TST @CSR ;TEST CSR
1792 006542 100401          BMI .+4 ;ERROR SET?
1793 006544 104000          HLT ;ERROR IS CLEAR - SHOULD HAVE ERROR
1794 006546 012777 006614 172362   MOV #ERRDO1,@DRINV ;INTERRUPT VECTOR TO ERRDO1
1795 006554 005077 172306   CLR @BAR ;PREVENT CAUSING ANOTHER ERROR
1796 006560 012777 177777 172276   MOV #-1,@WCR ;SET-UP WCR
1797 006566 005277 172276   INC @CSR ;GO TO CSR
1798 006572 005037 006600   CLR 1$+2
1799 006576 005227 000001   1$: INC #1 ;DELAY;WAIT FOR INTERRUPT
1800 006602 001375          BNE 1$
1801 006604 104000          HLT ;NO DR11-W INTERRUPT
1802 006606 000005          RESET ;CLEAR INTERRUPT REQUEST
1803 006610 000137 006632   JMP T60
1804 006614 022626          ERRDO1: CMP (SP)+,(SP)+
1805 006616 005777 172246   TST @CSR ;CHECK ERROR
1806 006622 100001          BPL .+4 ;ERROR SET?
1807 006624 104000          HLT ;ERROR IS SET - SHOULD BE CLEAR
1808 ;PREVIOUS ERROR WAS CLEAR
1809 006626 004737 010156   JSR PC,NORMAL
1810
1811
    
```

```

1813 .SBTTL TEST 47 THAT FUNCTION BITS INCREMENT WITH MAINT MODE TRANSFERS
1814 :*****
1815 : TEST 47 TEST THAT FUNCTION BITS INCREMENT WITH MAINT MODE TRANSFERS
1816 :*****
1817 006632 000004 T60: SCOPE
1818 006634 005077 172230 CLR @CSR ;FORCE TO BE CSR
1819 006640 012737 002000 014122 MOV #2000,ICOUNT ;SET-UP BAR
1820 006646 013777 001152 172212 MOV INBUF,@BAR ;SET-UP BAR
1821 006654 012737 000001 001202 MOV #1,FNCNT ;GET READY FOR CHECKING
1822 006662 012777 006762 172246 MFLOOP: MOV #T60CHK,@DRINV ;INTERRUPT VECTOR
1823 006670 013777 001074 172242 MOV DRINL,@DRVS ;INTERRUPT VECTOR PRIORITY TO DRINL
1824 006676 013737 001202 001306 MOV FNCNT,TEMP ;NUMBER OF TRANSFERS
1825 006704 005437 001306 NEG TEMP ;SET UP FOR WCR
1826 006710 013777 001306 172146 MOV TEMP,@WCR ;SET-UP FOR 1 TRANSFER
1827 006716 005077 172060 CLR @PSW ;LET THE DR11-W INTERRUPT
1828 006722 012777 010000 172140 MOV #10000,@CSR ;MAINT MODE
1829 006730 052777 000501 172132 BIS #501,@CSR ;IE,CYCLE & GO
1830 006736 012737 001000 00 312 MOV #1000,TIME ;WAIT FOR INTERRUPT
1831 006744 005337 001312 T6: DEC TIME
1832 006750 001375 BNE T6
1833 006752 104000 HLT ;NO INTERRUPT OCCURRED
1834 006754 000005 RESET ;IF NO INTERRUPT,CLEAR ANY OUTSTANDING
1835 ;INTERRUPT REQUEST IN BOARD HDWR.
1836 006756 000137 007024 JMP T61 ;GO TO NEXT TEST
1837 006762 022626 T60CHK: CMP (SP)+,(SP)+
1838 006764 117701 172100 MOVB @CSR,R1 ;LOWER BYTE OF CSR TO R1
1839 006770 042701 000700 BIC #700,R1 ;GET RID OF READY,CYCLE,IF BECAUSE OF MAINT MODE
1840 006774 006201 ASR R1 ;MOVE IT RIGHT ONE PLACE
1841 006776 123701 001202 CMPB FNCNT,R1 ;CHECK AGAINST FNCNT
1842 007002 001401 BEQ .+4 ;SHOULD BE EQUAL
1843 007004 104000 HLT ;FUNCTION BITS DIDN'T INCREMENT IN MAINT MODE
1844 007006 005237 001202 INC FNCNT ;GET READY FOR NEXT PASS
1845 007012 022737 000010 001202 CMP #10,FNCNT ;ONLY 10 BECAUSE FNCT3-1 GO TO ZERO
1846 007020 001401 BEQ T61 ;IF ITS EQUAL GO CHECK DATA
1847 007022 000717 BR MFLOOP ;DO IT AGAIN
    
```

```

1849
1850
1851
1852
1853 007024 000004
1854 007026 005077 172036
1855 007032 012737 000010 001156
1856 007040 013737 001156 001164
1857 007046 005437 001164
1858 007052 004737 007714
1859 007056 004737 007752
1860 007062 013777 001164 171774
1861 007070 013777 001152 171770
1862 007076 012777 177777 171766
1863 007104 012777 007174 172024
1864 007112 013777 001074 172020
1865 007120 005077 171656
1866 007124 012777 010000 171736
1867 007132 052777 000501 171730
1868 007140 012737 001000 001312
1869 007146 005337 001312
1870 007152 001375
1871 007154 104000
1872 007156 000005
1873
1874 007160 004737 010016
1875 007164 004737 010116
1876 007170 000137 007206
1877 007174 022626
1878 007176 004737 010016
1879 007202 004737 010116
1880

      .SBTTL TEST 50 TEST FOR 10 MAINT MODE TRANSFERS
      :*****
      :TEST 50 TEST FOR 10 MAINT MODE TRANSFERS
      :*****
T61:  SCOPE
      CLR      @CSR      ;FORCE TO BE CSR #1
      MOV      #10,BUFLEN ;BUFLEN=10
      MOV      BUFLN,WCLN ;PREPARE NUMBER FOR WCR
      NEG      WCLN      ;2'S COMPLEMENT OF BUFLN
      JSR      PC,LODBUF  ;LOAD IN BUFFER WITH INCREMENTING PATTERN
      JSR      PC,CHKBFF  ;LOAD CHECK BUFFER WITH MODIFIED INCREMENTING PATTERN
      MOV      WCLN,@WCR  ;SET UP WCR
      MOV      INBUF,@BAR ;SET UP BAR
      MOV      #-1,@BDR   ;MAINT AIDE
      MOV      #T61CHK,@DRINV ;INTERRUPT VECTOR
      MOV      DRINL,@DRVS ;INTERRUPT STATUS AT PRIORITY DRINL
      CLR      @PSW      ;LET DR11-W INTERRUPT
      MOV      #10000,@CSR ;MAINT MODE
      BIS      #501,@CSR  ;IE,CYCLE & GO
      MOV      #1000,TIME ;WAIT FOR INTERRUPT
T61:  DEC      TIME
      BNE     1$
      HLT
      RESET ;NO INTERRUPT OCCURED
      ;IF NO INTERRUPT,CLEAR ANY OUTSTANDING
      ;INTERRUPT REQUEST IN BOARD HDWR.
      JSR      PC,INTA
      JSR      PC,DATCHK
      JMP      T61.5
T61CHK: CMP      (SP)+,(SP)+
      JSR      PC,INTA ;CLEAR IE,CHECK ERROR,READY,WCR 0,& BAR
      JSR      PC,DATCHK
  
```

```

1882
1883
1884
1885
1886
1887
1888 007206 000004
1889 007210 005077 171654
1890 007214 012737 000010 014122
1891 007222 012777 000340 171552
1892 007230 012737 001000 001312
1893 007236 012737 000010 001156
1894 007244 013737 001156 001164
1895 007252 005437 001164
1896 007256 004737 007714
1897 007262 004737 007752
1898 007266 013777 001164 171570
1899 007274 013777 001152 171564
1900 007302 012777 177777 171562
1901 007310 012777 007360 171620
1902 007316 013777 001074 171614
1903 007324 012777 010000 171536
1904 007332 052777 000501 171530
1905 007340 005337 001312
1906 007344 001375
1907 007346 004737 010016
1908 007352 004737 010116
1909 007356 000404
1910 007360 022626
1911 007362 104000
1912 007364 000137 007514
1913 007370 012777 010000 171472
1914 007376 012737 001000 001312
1915 007404 013777 001164 171452
1916 007412 013777 001152 171446
1917 007420 012777 007510 171510
1918 007426 012777 010000 171434
1919 007434 052777 000501 171426
1920 007442 005337 001312
1921 007446 001375
1922 007450 022777 010700 171412
1923 007456 001401
1924 007460 104000
1925 007462 022777 177770 171374
1926 007470 001004
1927 007472 023777 001152 171366
1928 007500 001401
1929 007502 104000
1930 007504 000005
1931 007506 000402
1932 007510 022626
1933 007512 104000
1934 007514 004737 010156
1935
  
```

```

.SBTTL TEST 51 10 MAINTENANCE MODE XFERS
:*****
TEST 51 TEST THAT DOING 10 MAINTENANCE MODE XFERS
WITHOUT EXPERIENCING AN INTERRUPT, WILL INHIBIT
'GO' FROM INITIATING FURTHER XFERS
:*****
T61.5: SCOPE
      CLR @CSR ;FORCE TO BE CSR #1
      MOV #10,ICOUNT
      MOV #340,@PSW ;NO CPU INTERRUPT
      MOV #1000,TIME ;SET DELAY
      MOV #10,BUFLEN ;BUFLen=10
      MOV BUFLen,WCLen ;PREPARE NUMBER FOR WCR
      NEG WCLen ;2'S COMPLEMENT OF BUFLen
      JSR PC,LODBUF ;LOAD IN BUFFER WITH INCREMENTING PATTERN
      JSR PC,CHKBFF ;LOAD CHECK BUFFER WITH MODIFIED INCREMENTING PATTERN
      MOV WCLen,@WCR ;SET UP WCR
      MOV INBUF,@BAR ;SET UP BAR
      MOV #-1,@BDR ;MAINT AIDE
      MOV #WRONG1,@DRINV ;INTERRUPT VECTOR
      MOV DRINL,@DRVS ;INTERRUPT STATUS AT PRIORITY DRINL
      MOV #10000,@CSR ;MAINT MODE
      BIS #501,@CSR ;IE,CYCLE & GO
1$: DEC TIME ;WAIT FOR XFERS COMPLETE
      BNE 1$
      JSR PC,INTA ;CLR IE,CHECK ERROR,READY,WCR=0,BAR
      JSR PC,DATCHK ;CHECK PROPER DATA
      BR OK
WRONG1: CMP (SP)+,(SP)+ ;SHOULD NOT INTERRUPT
      HLT ;NO INTERRUPT 1ST TIME
      JMP NRM
OK: MOV #10000,@CSR
      MOV #1000,TIME
      MOV WCLen,@WCR
      MOV INBUF,@BAR
      MOV #WRONG2,@DRINV
      MOV #10000,@CSR ;MAINT MODE
      BIS #501,@CSR ;IE,CYCLE,GO
3$: DEC TIME
      BNE 3$
      CMP #10700,@CSR ;ONLY READY,MAIN,IE,& CYCLE
      BEQ .+4
      HLT ;CSR IN ERROR
      CMP #-10,@WCR ;CHECK THAT NO XFERS WERE MADE
      BNE 5$
      CMP INBUF,@BAR ;
      BEQ RSET
5$: HLT ;XFERS SHOULD HAVE BEEN INHIBITED
RSET: RESET ;WILL ELIMINATE ANY OUTSTANDING INTERRUPT REQUESTS
      BR NRM
WRONG2: CMP (SP)+,(SP)+ ;READJUST STACK
      HLT ;SHOULD NOT INTERRUPT 2ND TIME
NRM: JSR PC,NORMAL
  
```

```

1937
1938
1939
1940
1941 007520 000004
1942 007522 005077 171342
1943 007526 012737 002000 014122
1944 007534 012737 000200 001156
1945 007542 013737 001156 001164
1946 007550 005437 001164
1947 007554 004737 007714
1948 007560 004737 007752
1949 007564 013777 001164 171272
1950 007572 013777 001152 171266
1951 007600 012777 000001 171264
1952 007606 012777 007676 171322
1953 007614 013777 001074 171316
1954 007622 005077 171154
1955 007626 012777 010000 171234
1956 007634 052777 000501 171226
1957 007642 012737 001000 001312
1958 007650 005337 001312
1959 007654 001375
1960 007656 104000
1961 007660 000005
1962
1963 007662 004737 010016
1964 007666 004737 010116
1965 007672 000137 010276
1966 007676 022626
1967 007700 004737 010016
1968 007704 004737 010116
1969 007710 000137 010276

.SBTTL TEST 52 TEST FOR 200 NPR TRANSFERS IN MAINT MODE
:*****
:TEST 52 TEST FOR 200 NPR TRANSFERS IN MAINT MODE
:*****
SCOPE
CLR @CSR ;FORCE TO BE CSR #1
MOV #200,ICOUNT
MOV #200,BUFLEN ;LENGTH OF BUFFER = 200
MOV BUFLN,WCLN ;PREPARE NUMBER FOR WCR
NEG WCLN ;2'S COMPLEMENT OF BUFLN
JSR PC,LODBUF ;LOAD INBUF WITH INCREMENTING PATTERN
JSR PC,CHKBUF ;LOAD CHKBUFF WITH MODIFIED INCREMENTED PATTERN
MOV WCLN,@WCR ;SET UP WCR
MOV INBUF,@BAR ;SET UP BAR
MOV #1,@BDR ;MAINT AIDE
MOV #T62CHK,@DRINV ;INT VECTOR
MOV DRINL,@DRVS ;INT VECTOR AT PRIORITY DRINL
CLR @PSW ;LET THE DR11-W INTERRUPT
MOV #10000,@CSR ;MAINT MODE
BIS #501,@CSR ;IE,CYCLE & GO
MOV #1000,TIME
1$: DEC TIME
BNE 1$
HLT ;NO INTERRUPT OCCURED
RESET ;IF NO INTERRUPT,CLEAR ANY OUTSTANDING
;INTERRUPT REQUEST IN BOARD HDWR.

JSR PC,INTA
JSR PC,DATCHK
JMP T42
T62CHK: CMP (SP)+,(SP)+
JSR PC,INTA ;CLR IF,CHECK ERROR,RESDY,WC 0,& BAR
JSR PC,DATCHK ;CHECK THAT CORRECT DATA WAS TRANSFERRED
JMP T42
    
```

```

1971 007714 013702 001152          LODBUF: MOV     INBUF,R2          ;MOVE STARTING ADDRESS OF INBUF TO R2
1972 007720 005037 001160          CLR     LENCHK          ;CLEAR LENGTH CHECK
1973 007724 005022                    CLR     (R2)+          ;CLEAR STARTING ADDRESS OF INBUFF AND INC BY 2
1974 007726 005237 001160          LOADA: INC    LENCHK          ;INC LENGTH CHECK BY 1
1975 007732 023737 001160 001156  CMP     LENCHK,BUFLEN    ;CHECK FOR DONE
1976 007740 001403                    BEQ     LDEXIT          ;IS INBUF FILLED?
1977 007742 013722 001160          MOV     LENCHK,(R2)+    ;LOAD NEXT BUFFER WORD
1978 007746 000767                    BR     LOADA           ;CONTINUE CHECKING
1979 007750 000207          LDEXIT: RTS     PC           ;EXIT
1980 007752 013702 001154          CHKBFF: MOV    CHKBUF,R2  ;STARTING ADDRESS OF CHECK-BUFFER TO R2
1981 007756 005037 001160          CLR     LENCHK          ;CLEAR LENGTH CHECK
1982 007762 005003                    CLR     R3             ;CLEAR R3
1983 007764 010322          CHKA:  MOV     R3,(R2)+   ;MOVE R3 TO CHKBUF ADDRESS AND INC BY 2
1984 007766 010322          MOV     R3,(R2)+   ;MOVE R3 TO NEXT CHKBUF ADDRESS AND INC BY 2
1985 007770 062737 000002 001160  ADD     #2,LENCHK       ;ADD 2 TO LENGTH CHECK
1986 007776 023737 001160 001156  CMP     LENCHK,BUFLEN    ;CHECK FOR DONE
1987 010004 100003                    BPL     .+10           ;IS CHECK-BUFFER FILLED?
1988 010006 062703 000002          ADD     #2,R3         ;NEXT NUMBER FOR BUFFER
1989 010012 000764                    BR     CHKA           ;CONTINUE FILLING
1990 010014 000207          RTS     PC           ;EXIT
1991 010016 042777 000100 171044  INTA:  BIC     #BIT6,@CSR   ;CLEAR IE
1992 010024 005777 171040          TST     @CSR          ;CHECKING FOR ERROR
1993 010030 100001                    BPL     .+4           ;ERROR SET?
1994 010032 104000                    HLT     ;ERROR BIT IS SET
1995 010034 105777 171030          TSTB   @CSR          ;CHECKING READY BIT
1996 010040 100401                    BMI     .+4           ;IS READY SET
1997 010042 104000                    HLT     ;FALSE INTERRUPT - ERROR AND READY ARE CLEAR
1998 010044 005777 171014          TST     @WCR          ;TEST1 FOR WCR=0
1999 010050 001401                    BEQ     .+4           ;WAS IT EQUAL?
2000 010052 104000                    HLT     ;WC NOT = 0
2001 010054 013702 001156          MOV     BUFLN,R2       ;BUFFER LENGTH TO R2
2002 010060 063702 001156          ADD     BUFLN,R2       ;NUMBER OF XFERS TIMES 2
2003 010064 063702 001152          ADD     INBUF,R2       ;CORRECT BAR
2004 010070 005737 001314          TST     CABLE          ;IS THIS CABLE TESTING?
2005 010074 001401                    BEQ     1$           ;NO
2006 010076 005202          INC     R2             ;CABLE MODE TESTING LEAVES BIT 0
2007                    ;OF BAR SET.THEREFORE MUST CHECK
2008                    ;FOR ODD ADDRESS.
2009 010100 027702 170762          1$:   CMP     @BAR,R2       ;CHECKING BAR
2010 010104 001401                    BEQ     .+4           ;IS BAR CORRECT
2011 010106 104000                    HLT     ;NO
2012 010110 004737 010156          JSR     PC,NORMAL      ;EXIT
2013 010114 000207          RTS     PC
2014
2015 010116 013702 001154          DATCHK: MOV    CHKBUF,%2  ;STARTING ADDRESS OF CHECK BUFFER TO R2
2016 010122 013703 001152          MOV    INBUF,%3       ;STARTING ADDRESS OF IN BUFFER TO R3
2017 010126 005037 001160          CLR    LENCHK         ;CLEAR LENGTH CHECK
2018 010132 005237 001160          COMPAR: INC   LENCHK         ;MAKE A COMPARISON
2019 010136 022223          CMP    (%2)+,(%3)+    ;IS THE DATA CORRECT?
2020 010140 001401                    BEQ    .+4           ;BRANCH IF OK
2021 010142 104000                    HLT    ;BAD DATA
2022 010144 023737 001160 001156  CMP    LENCHK,BUFLEN  ;SEE IF THE BUFFER HAS BEEN CHECKED
2023 010152 001367          BNE    COMPAR         ;BUFFER CHECKED?
2024 010154 000207          RTS     %7
2025 010156 013777 001140 170752  NORMAL: MOV   DRVS,@DRINV ;RESTORE DR11-B INTERRUPT VECTOR
2026 010164 005077 170750          CLR    @DRVS         ;RESTORE DR11-B INTERRUPT STATUS
2027 010170 012777 000340 170604  MOV    #340,@PSW      ;RESTORE PROC TO PRIORITY LEVEL ?
    
```



```

2028 010176 000207          RTS      %7          ;EXIT
2029 010200 012702 052525  DATOCK: MOV    #52525,%2    ;DATO NUMBER TO R2
2030 010204 013703 001152          MOV    INBUF,%3    ;STARTING ADDRESS OF IN BUFFER TO R3
2031 010210 005037 001160          CLR    LENCHK      ;CLEAR LENGTH CHECK
2032 010214 005237 001160  COMPRR: INC    LENCHK      ;MAKE A COMPARISON
2033 010220 020223          CMP    %2,(%3)+    ;IS THE DATA CORRECT?
2034 010222 001401          BEQ    .+4         ;BRANCH IF OK
2035 010224 104000          HLT                    ;BAD DATA
2036 010226 023737 001160 001156  CMP    LENCHK,BUFLEN ;SEE IF THE BUFFER HAS BEEN CHECKED
2037 010234 001367          BNE    COMPRR      ;BUFFER CHECKED?
2038 010236 020223          CMP    %2,(%3)+    ;CHECK END OF BUFFER + 1
2039 010240 001001          BNE    .+4         ;SEE IF TOO MANY WORDS WERE TRANSFERRED
2040 010242 104000          HLT                    ;TOO MANY
2041 010244 000207          RTS      %7          ;EXIT
2042 010246 042777 000100 170614  ERRCHK: BIC    #BIT6,@CSR    ;CLEAR IE
2043 010254 005777 170610          TST    @CSR        ;CHECKING FOR ERROR
2044 010260 100001          BPL    .+4         ;ERROR SET?
2045 010262 104000          HLT                    ;ERROR BIT IS SET
2046 010264 105777 170600          TSTB   @CSR        ;CHECKING READY BIT
2047 010270 100401          BMI    .+4         ;IS RDY SET
2048 010272 104000          HLT                    ;FALSE ENTRY - ERROR AND READY ARE CLEAR
2049 010274 000207          RTS      %7          ;EXIT
2050
2051
    
```


2092
2093
2094
2095
2096
2097 010452 000004
2098 010454 005077 170410
2099 010460 012737 000010 014122
2100 010466 012777 177760 170370
2101 010474 012777 177776 170364
2102 010502 012777 010546 170426
2103 010510 013777 001074 170422
2104 010516 012737 001000 001312
2105 010524 005077 170252
2106 010530 012777 000563 170332
2107 010536 005337 001312
2108 010542 001375
2109 010544 104000
2110 010546 017705 170316
2111 010552 022626
2112 010554 005705
2113 010556 100401
2114 010560 104000
2115 010562 105705
2116 010564 100401
2117 010566 104000
2118 010570 032705 040000
2119 010574 001001
2120 010576 104000
2121 010600 005077 170262
2122 010604 005077 170260
2123 010610 004737 014144
2124 010614 000005
2125 010616 004737 010156
2126
2127
2128
2129
2130

```

.SBTTL TEST 54 CROSSING A 32K BOUNDRY DOESNT CAUSE A BAOF OR FORCE ERROR
:*****
TEST 54 TEST THAT CROSSING A 32K BOUNDRY DOES NOT CAUSE
A BAOF AND DOES NOT FORCE AN ERROR
:*****
SCOPE
CLR @CSR ;FORCE TO BE CSR #1
MOV #10,ICOUNT
MOV #-20,@WCR ;SET UP WCR
MOV #-2,@BAR ;SET UP BAR FOR PROC STATUS ADDRESS
MOV #BAOFCK,@DRINV ;INTERRUPT VECTOR TO BAOFCK
MOV DRINL,@DRVS ;INTERRUPT STATUS TO LEVEL DRINL
MOV #1000,TIMF
CLR @PSW ;LET THE DR11-W INTERRUPT
MOV #563,@CSR ;CYCLE,IE, FNCT1, XBA17, XBA16, AND GO TO CSR
1$: DEC TIME
BNE 1$ ;WAIT FOR INTERRUPT
HLT ;NO INTERRUPT
BAOFCK: MOV @CSR,R5 ;SAVE CSR
CMP (SP)+,(SP)+ ;RESTORE THE STACK
TST R5 ;TEST CSR
BMI .+4 ;ERROR SET?
HLT ;ERROR NOT SET
TSTB R5 ;TEST FOR READY
BMI .+4 ;READY SET?
HLT ;READY ISN'T SET
BIT #40000,R5 ;CHECK NEX
BNE .+4 ;ARE THEY CLEAR?
HLT ;NEX IS CLEAR
CLR @BAR ;CLEAR BUS ADDRESS REGISTER
CLR @CSR ;RETURN TO CSR #1
JSR PC,CKSWR
RESET ;INIT
JSR PC,NORMAL

;THIS ENDS MAINTENANCE MODE TESTING

```

```

2132 .SBTTL
2133 .SBTTL      CABLE MODE TESTING
2134 .SBTTL
2135           ;CABLE MODE TESTING(WRAP-AROUND
2136           ;CABLE IN USER SLOTS)
2137
2138           ;TESTS 55 THRU 62 ARE PERFORMED IF:
2139           : 1.SWITCH REGISTER 10 IS SET,OR
2140           : 2.SWITCH REG. 10 & 9 ARE DOWN(0),AND THE CABLE IS
2141           : IN PLACE:THE PROGRAM WILL TEST FOR CABLE.
2142
2143           ;THE FOLLOWING ROUTINE DETERMINES IF CABLE IS IN
2144           ;OR SWITCH REGISTER OPTION IS SELECTED
2145 010622 005077 170242 CBL: CLR @CSR ;FORCE TO BE CSR #1
2146 010626 004737 014144 JSR PC,CKSWR
2147 010632 032777 002000 170466 BIT #2000,@SWR ;MAINTENANCE MODE TESTS ONLY?
2148 010640 001402 BEQ 1$ ;NO
2149 010642 000137 013132 JMP END ;YES
2150 010646 032777 001000 170452 1$: BIT #1000,@SWR ;PERFORM CABLE TESTS UNCONDITIONALLY?
2151 010654 001402 BEQ 2$ ;NO;DETERMINE CABLE STATUS BY PROGRAM DETECTION
2152 010656 000137 010722 JMP YESCBL
2153 010662 000005 2$: RESET
2154 010664 005077 170200 CLR @CSR ;FORCE CSR #1
2155 010670 017701 170174 MOV @CSR,R1 ;CHECK CSR
2156 010674 032701 127000 BIT #127000,R1 ;TEST FOR ERROR,ATTN,DSTATB,DCTATC
2157 010700 001015 BNE NOCBL
2158 010702 112777 000004 170160 MOVB #4,@CSR ;IF ATTN IS SET,CABLE IS IN
2159 010710 017701 170154 MOV @CSR,R1
2160 010714 032701 020000 BIT #20000,R1
2161 010720 001405 BEQ NOCBL
2162 010722 012737 000001 001314 YESCBL: MOV #1,CABLE
2163 010730 000137 010744 JMP T33
2164 010734 005037 001314 NOCBL: CLR CABLE ;DON'T DO CABLE TESTS
2165 010740 000137 013132 JMP END

```

```

2167 .SBTTL TEST 55 TEST FOR 1 DATI NON BURST MODE TRANSFER
2168 :*****
2169 :TEST 55 TEST FOR 1 DATI NON BURST MODE TRANSFER
2170 :CABLE MODE(WITH WRAP AROUND CABLE IN USER SLOTS)
2171 :*****
2172 010744 005077 170120 T33: CLR @CSR ;FORCE TO BE CSR #1
2173 010750 012737 010764 014126 MOV #T33STR,RETURN ;REVISE SCOPE RETURN
2174 010756 012737 000001 001314 MOV #1,CABLE ;PRINT CABLE MESGE WITH ERROR PRINT
2175 010764 T33STR:
2176 010764 005077 170100 CLR @CSR
2177 010770 005777 170074 TNPR1: TST @CSR ;CHECK ERROR BIT
2178 010774 100001 BPL .+4 ;IS IT CLEAR?
2179 010776 104000 HLT ;ERROR SHOULD NOT BE SET
2180 011000 022777 000200 170062 CMP #200,@CSR ;CHECK READY
2181 011006 001401 BEQ .+4 ;
2182 011010 104000 HLT ;SOMETHING OTHER THAN READY IS SET
2183 011012 012777 177777 170044 NPRRDY: MOV #-1,@WCR ;SET UP FOR 1 TRANSFER
2184 011020 012777 001146 170040 MOV #NPR1,@BAR ;TRANSFER FROM BUS ADDRESS IN NPR1
2185 011026 005077 170040 CLR @BDR ;GET READY TO RECEIVE DATA
2186 011032 012737 052525 001146 MOV #52525,NPR1 ;SET UP TRANSFER DATA
2187 011040 012777 011114 170070 MOV #INTB,@DRINV ;INTERRUPT VECTOR TO INTB
2188 011046 012777 000005 170064 MOV #5,@DRVS ;INTERRUPT PRIORITY TO LEVEL 5
2189 011054 005077 167722 CLR @PSW ;LET THE DR11-W INTERRUPT
2190 011060 012777 000110 170002 MOV #110,@CSR ;NON BURST(FNCT3),IE
2191 011066 052777 000401 167774 BIS #401,@CSR ;CYCLE,GO
2192 011074 005037 011102 CLR 1$+2 ;WAIT FOR NPR AND INTERRUPT
2193 011100 005227 000001 1$: INC #1
2194 011104 001375 BNE 1$
2195 011106 104000 HLT ;NO DR11-W INTERRUPT
2196 011110 000005 RESET ;IF NO INTERRUPT,CLEAR ANY OUTSTANDING
2197 ;INTERRUPT REQUEST IN BOARD HDWR.
2198 011112 000424 BR T33CLR ;CLEAR IE
2199 011114 004737 010246 INTB: JSR PC,ERPCHK ;CLEAR IE,CHECK FOR ERROR
2200 011120 005777 167740 TST @WCR ;TEST WCR
2201 011124 001401 BEQ .+4 ;IS WCR EQUAL TO ZERO?
2202 011126 104000 HLT ;WCR NOT EQUAL TO ZERO
2203 011130 022777 001151 167730 CMP #NPR1+3,@BAR ;COMPARE CORRECT BAR WITH BAR
2204 ;CABLE MODE TESTING LEAVES BIT 0
2205 ;OF BAR SET.THEREFORE MUST
2206 ;CHECK FOR ODD ADDRESS
2207 011136 001401 BEQ .+4 ;IS THE BAR CORRECT?
2208 011140 104000 HLT ;BAR IS WRONG
2209 011142 022777 052525 167722 CMP #52525,@BDR ;CHECK FOR CORRECT DATA
2210 011150 001401 BEQ .+4 ;DATA GET TRANSFERRED?
2211 011152 104000 HLT ;BAD DATA IN BDR
2212 011154 004737 010156 JSR PC,NORMAL
2213 011160 022626 CMP (SP)+,(SP)+ ;RESTORE STACK
2214 011162 000404 BR TNPRO ;GO TO NEXT TEST
2215 011164 005077 167700 T33CLR: CLR @CSR ;CLEAR IE
2216 011170 004737 010156 JSR PC,NORMAL
2217
    
```

```

2219 .SBTTL TEST 56 TEST FOR 1 DATO NON BURST MODE TRANSFER
2220 :*****
2221 :TEST 56 TEST FOR 1 DATO NON BURST MODE TRANSFER
2222 :CABLE MODE(WITH WRAP AROUND CABLE IN USER SLOTS)
2223 :*****
2224 011174 000004 TNPRO: SCOPE
2225 011176 005077 167666 CLR @CSR ;FORCE TO BE CSR #1
2226 011202 012777 177777 167654 MOV #-1,@WCR ;SET UP FOR 1 TRANSFER
2227 011210 012777 001146 167650 MOV #NPR1,@BAR ;TRANSFER TO BUS ADDRESS IN NPR1
2228 011216 005037 001146 CLR NPR1 ;GET READY TO RECEIVE DATA
2229 011222 012777 052525 167642 MOV #52525,@BDR ;SET UP TO TRANSFER DATA
2230 011230 012777 011304 167700 MOV #INTC,@DRINV ;INTERRUPT VECTOR TO INTC
2231 011236 013777 001074 167674 MOV DRINL,@DRVS ;INTERRUPT STATUS TO LEVEL DRINL
2232 011244 005077 167532 CLR @PSW ;PROC STATUS TO ZERO
2233 011250 012777 000112 167612 MOV #112,@CSR ;DATO(FNCT1),FNCT3,IE
2234 011256 052777 000401 167604 BIS #401,@CSR ;CYCLE,GO
2235 011264 005037 011272 CLR 1$+2 ;WAIT FOR NPR AND INTER
2236 011270 005227 000001 1$: INC #1
2237 011274 001375 BNE 1$
2238 011276 104000 HLT ;NO DR11-W INTERRUPT
2239 011300 000005 RESET ;IF NO INTERRUPT,CLEAR ANY OUTSTANDING
2240 ;INTERRUPT REQUEST IN BOARD HDWR.
2241 011302 000424 BR T34CLR ;CLEAR IE
2242 011304 004737 010246 INTC: JSR PC,ERRCHK ;CLEAR IE,CHECK FOR ERROR
2243 011310 005777 167550 TST @WCR ;TEST WCR
2244 011314 001401 BEQ .+4 ;IS WCR EQUAL TO ZERO?
2245 011316 104000 HLT ;WCR EQUAL TO ZERO
2246 011320 022777 001151 167540 CMP #NPR1+3,@BAR ;COMPARE CORRECT BAR WITH BAR
2247 ;CABLE MODE TESTING LEAVES BIT 0
2248 ;OF BAR SET.THEREFORE MUST
2249 ;CHECK FOR ODD ADDRESS
2250 011326 001401 BEQ .+4 ;IS THE BAR CORRECT?
2251 011330 104000 HLT ;BAR IS WRONG
2252 011332 023727 001146 052525 CMP NPR1,#52525 ;CHECK FOR CORRECT DATA
2253 011340 001401 BFQ .+4 ;CORRECT DATA TRANSFERRED?
2254 011342 104000 HLT ;BAD DATA
2255 011344 004737 010156 JSR PC,NORMAL
2256 011350 022626 CMP (SP)+,(SP)+ ;RESTORE STACK
2257 011352 000404 BR T37 ;GO TO NEXT TEST
2258 011354 005077 167510 T34CLR: CLR @CSR ;CLEAR IE
2259 011360 004737 010156 JSR PC,NORMAL
2260
2261
    
```

```

2263 .SBTTL TEST 57 STRING OF 200 DATIS BURST MODE XFERS
2264 :*****
2265 :TEST 57 STRING OF 200 DATI'S BURST MODE XFERS
2266 :CABLE MODE(WITH WRAP AROUND CABLE IN USER SLOTS)
2267 :*****
2268 011364 000004 T37: SCOPE
2269 011366 005077 167476 CLR @CSR ;FORCE TO BE CSR #1
2270 011372 012737 000200 001156 MOV #200,BUFLEN ;LENGTH OF BUFFER=200
2271 011400 004737 007714 JSR PC,LODBUF ;LOAD THE BUFFER WITH INCREMENTING PATTERN
2272 011404 013737 001156 001164 MOV BUFLEN,WCLEN ;PREPARE NUMBER FOR WCR
2273 011412 005437 001164 NEG WCLEN ;2'S COMPLEMENT OF BUFLEN
2274 011416 013777 001164 167440 MOV WCLEN,@WCR ;SET UP WCR
2275 011424 013777 001152 167434 MOV INBUF,@BAR ;SET UP BAR
2276 011432 012777 177777 167432 MOV #-1,@BDR ;MAINT AIDE
2277 011440 012777 011524 167470 MOV #T37CHK,@DRINV ;INT VECTOR
2278 011446 013777 001074 167464 MOV DRINL,@DRVS ;INT VECTOR TO PRIORITY DRINL
2279 011454 005077 167322 CLR @PSW ;LET THE DR11-W INTERRUPT
2280 011460 012777 000100 167402 MOV #100,@CSR ;IE
2281 011466 052777 000401 167374 BIS #401,@CSR ;CYCLE,GO
2282 011474 012737 001000 001312 MOV #1000,TIME ;WAIT FOR INTERRUPT
2283 011502 005337 001312 1$: DEC TIME
2284 011506 001375 BNE 1$
2285 011510 104000 HLT
2286 011512 000005 RESET ;IF NO INTERRUPT,CLEAR ANY OUTSTANDING
2287 ;INTERRUPT REQUEST IN BOARD HDWR.
2288 011514 004737 010156 JSR PC,NORMAL
2289 011520 000137 011544 JMP T40
2290 011524 022626 T37CHK: CMP (SP)+,(SP)+
2291 011526 004737 010016 JSR PC,INTA ;CLR IE,CHECK ERROR,READY,WC=0,@BAR
2292 011532 022777 000177 167332 CMP #177,@BDR ;CHECK THAT WORD #200 OF INBUF IS IN BAR
2293 011540 001401 BEQ .+4 ;IS IT?
2294 011542 104000 HLT ;BAD DATA IN BDR
    
```

```

2296 .SBTTL TEST 60 STRING OR 200 DATOS BURST MODE XFERS
2297 :*****
2298 :TEST 60 STRING OR 200 DATO'S BURST MODE XFERS
2299 :CABLE MODE(WITH WRAP AROUND CABLE IN USER SLOTS)
2300 :*****
2301 011544 000004 T40: SCOPE
2302 011546 005077 167316 CLR @CSR ;FORCE TO BE CSR #1
2303 011552 012737 000200 001156 MOV #200,BUFLEN ;LENGTH OF BUFFER=200
2304 011560 004737 007714 JSR PC,LODBUF ;LOAD THE BUFFER WITH INCREMENTING PATTERN
2305 011564 013737 001156 001164 MOV BUFLEN,WCLEN ;PREPARE NUMBER FOR WCR
2306 011572 005437 001164 NEG WCLEN ;2'S COMPLEMENT OF BUFLEN
2307 011576 013777 001164 167260 MOV WCLEN,@WCR ;SET UP WCR
2308 011604 013777 001152 167254 MOV INBUF,@BAR ;SET UP BAR
2309 011612 012777 052525 167252 MOV #52525,@BDR ;SET UP BDR
2310 011620 012777 011704 167310 MOV #T40CHK,@DRINV ;INTERRUPT VECTOR
2311 011626 013777 001074 167304 MOV DRINL,@DRVS ;INTERRUPT VECTOR TO PRIORITY DRINL
2312 011634 005077 167142 CLR @PSW ;LET THE DR11-W INTERRUPT
2313 011640 012777 000102 167222 MOV #102,@CSR ;IE,FNCT1
2314 011646 052777 000401 167214 BIS #401,@CSR ;CYCLE,GO
2315 011654 012737 001000 001312 MOV #1000,TIME ;WAIT FOR INTERRUPT
2316 011662 005337 001312 1$: DEC TIME
2317 011666 001375 BNE 1$
2318 011670 104000 HLT
2319 011672 000005 RESET ;IF NO INTERRUPT,CLEAR ANY OUTSTANDING
2320 ;INTERRUPT REQUEST IN BOARD HDWR.
2321 011674 004737 010156 JSR PC,NORMAL
2322 011700 000137 011716 JMP T56
2323 011704 022626 T40CHK: CMP (SP)+,(SP)+
2324 011706 004737 010016 JSR PC,INTA ;CLEAR IE,CHECK ERROR,READY,W(0)& BAR
2325 011712 004737 010200 JSR PC,DATOCK ;CHECK INBUF
    
```



```

2327 .SBTTL TEST 61 STRING OF 200 DATIS NON-BURST MODE
2328 :*****
2329 : TEST 61 STRING OF 200 DATI'S NON-BURST MODE
2330 : CABLE MODE(WITH WRAP AROUND CABLE IN USER SLOTS)
2331 :*****
2332 011716 000004 T56: SCOPE
2333 011720 005077 167144 CLR @CSR ;FORCE TO BE CSR #1
2334 011724 012737 000200 001156 MOV #200,BUFLEN ;LENGTH OF BUFFER=200
2335 011732 004737 007714 JSR PC,LODBUF ;LOAD THE BUFFER WITH INCREMENTING PATTERN
2336 011736 013737 001156 001164 MOV BUFLEN,WCLEN ;PREPARE NUMBER FOR WCR
2337 011744 005437 001164 NEG WCLEN ;2'S COMPLEMENT OF BUFLEN
2338 011750 013777 001164 167106 MOV WCLEN,@WCR ;SET-UP WCR
2339 011756 013777 001152 167102 MOV INBUF,@BAR ;SET-UP BAR
2340 011764 012777 177777 167100 MOV #-1,@BDR ;MAINT AIDE
2341 011772 012777 012056 167136 MOV #T56CHK,@DRINV ;INT VECTOR
2342 012000 013777 001074 167132 MOV DRINL,@DRVS ;INT VECTOR TO PRIORITY DRINL
2343 012006 005077 166770 CLR @PSW ;LET THE DR11-W INTERRUPT
2344 012012 012777 000110 167050 MOV #110,@CSR ;FNCT3,IE
2345 012020 052777 000401 167042 BIS #401,@CSR ;CYCLE,GO
2346 012026 012737 001000 001312 MOV #1000,TIME ;WAIT FOR INTERRUPT
2347 012034 005337 001312 T5: DFC TIME
2348 012040 001375 BNE T5
2349 012042 104000 HLT
2350 012044 000005 RESET ;IF NO INTERRUPT,CLEAR ANY OUTSTANDING
2351 ; INTERRUPT REQUEST IN BOARD HDWR.
2352 012046 004737 010156 JSR PC,NORMAL
2353 012052 000137 012100 JMP T57
2354 012056 022626 T56CHK: CMP (SP)+,(SP)+
2355 012060 004737 010016 JSR PC,INTA ;CLEAR IE,CHECK ERROR,READY,WC 0,& BAR
2356 012064 000240 NOP
2357 012066 022777 000177 166776 CMP #177,@BDR ;CHECK THAT WORD #200 OF INBUF IS IN BAR
2358 012074 001401 BEQ .+4 ;IS IT?
2359 012076 104000 HLT ;BAD DATA IN BDR
2360

```

```

2362 .SBTTL TEST 62 STRING OF 200 DATOS NON-BURST MODE
2363 :*****
2364 :TEST 62 STRING OF 200 DATO'S NON-BURST MODE
2365 :CABLE MODE(WITH WRAP AROUND CABLE IN USER SLOTS)
2366 :*****
2367 012100 000004 57: SCOPE
2368 012102 005077 166762 CLR @CSR ;FORCE TO BE CSR #1
2369 012106 012737 000200 001156 MOV #200,BUFLEN ;LENGTH OF BUFFER-200
2370 012114 004737 007714 JSR PC,LODBUF ;LOAD THE BUFFER WITH INCREMENTING PATTERN
2371 012120 013737 001156 001164 MOV BUFLEN,WCLEN ;PREPARE NUMBER FOR WCR
2372 012126 005437 001164 NEG WCLEN ;2'S COMPLEMENT OF BUFLEN
2373 012132 013777 001164 166724 MOV WCLEN,@WCR ;SET UP WCR
2374 012140 013777 001152 166720 MOV INBUF,@BAR ;SET UP BAR
2375 012146 012777 052525 166716 MOV #52525,@BDR ;SET UP BDR
2376 012154 012777 012234 166754 MOV #T57CHK,@DRINV ;INTERRUPT VECTOR
2377 012162 013777 001074 166750 MOV DRINL,@DRVS ;INTERRUPT VECTOR TO PRIORITY DRINI
2378 012170 005077 166606 CLR @PSW ;LET THE DR11-W INTERRUPT
2379 012174 012777 000102 166666 MOV #102,@CSR ;FNCT1,IE
2380 012202 052777 000401 166660 BIS #401,@CSR ;CYCLE,GO
2381 012210 012737 001000 001312 MOV #1000,TIME ;WAIT FOR INTERRUPT
2382 012216 005337 001312 1$: DEC TIME
2383 012222 001375 BNE 1$
2384 012224 104000 HLT
2385 012226 000005 RESET ;IF NO INTERRUPT,CLEAR ANY OUTSTANDING
2386 ;INTERRUPT REQUEST IN BOARD HDWR.
2387 012230 004737 010156 JSR PC,NORMAL
2388 012234 022626 T57CHK: CMP (SP)+,(SP)+
2389 012236 004737 010016 JSR PC,INTA ;CLEAR IE,CHECK ERROR,READY,WCR O.B BAR
2390 012242 000240 NOP
2391 012244 004737 010200 JSR PC,DATOCK ;CHECK INBUF
2392
    
```

```

2394 .SBTTL TEST 63 DETERMINE N-CYCLE SWITCH SETTING
2395 :*****
2396 :TEST 63 DETERMINE N-CYCLE SWITCH SETTING
2397 :HAVE SWITCH FLIPPED AND WAIT UNTIL IT IS DONE
2398 :THIS IS DONE FIRST PASS ONLY
2399 :*****
2400 012250 032737 000001 001144 BIT #BIT0,BORW
2401 012256 001550 BEQ SWT1
2402 012260 005737 001200 TST FLAG ;WERE WE HERE BEFORE
2403 012264 001402 BEQ 10$ ;NO
2404 012266 000137 013132 JMP END ;YES DONE THIS PASS
2405 012272 005237 001200 10$: INC FLAG ;SET FLAG FOR NEXT PASS
2406 012276 012777 100000 166564 MOV #BIT15,@CSR ;GO TO EIR
2407 012304 032777 000400 166556 BIT #BIT8,@CSR ;IS IT SET
2408 012312 001055 BNE FLOP ;YES
2409 012314 012737 077777 001312 FLIP: MOV #77777,TIME ;SET TIMER
2410 012322 012705 000100 MOV #100,R5 ;SET PRINT DELAY
2411 012326 104401 012334 TYPE ,65$ ;TYPE ASCIZ STRING
    012332 000425 BR 64$ ;GET OVER THE ASCIZ
    ;:65$: .ASCIZ <CRLF>/CHANGE POSITION OF N-CYCLE BURST SWITCH/
    64$:
2412 012406 032777 000400 166454 20$: BIT #BIT8,@CSR ;CHECK FOR CHANGE
2413 012414 001402 BEQ 30$ ;NOT YET
2414 012416 000137 012600 JMP SWT1 ;DONE CONTINUE
2415 012422 005337 001312 30$: DEC TIME ;DEC TIMER
2416 012426 001367 BNE 20$ ;CHECK AGAIN
2417 012430 012737 077777 001312 MOV #77777,TIME ;RESET TIMER
2418 012436 005305 DEC R5 ;DEC PRINT DELAY
2419 012440 001362 BNE 20$ ;CHECK AGAIN
2420 012442 000137 012314 JMP FLIP ;TELL AGAIN TO FLIP SWITCH
2421 012446 012737 077777 001312 FLOP: MOV #77777,TIME ;SET TIMER
2422 012454 012705 000100 MOV #100,R5 ;SET PRINT DELAY
2423 012460 104401 012466 TYPE ,65$ ;TYPE ASCIZ STRING
    012464 000425 BR 64$ ;GET OVER THE ASCIZ
    ;:65$: .ASCIZ <CRLF>/CHANGE POSITION OF N-CYCLE BURST SWITCH/
    64$:
2424 012540 032777 000400 166322 40$: BIT #BIT8,@CSR ;CHECK FOR CHANGE
2425 012546 001002 BNE 50$ ;NOT YET
2426 012550 000137 012600 JMP SWT1 ;DONE CONTINUE
2427 012554 005337 001312 50$: DEC TIME ;DEC TIMER
2428 012560 001767 BEQ 40$ ;CHECK AGAIN
2429 012562 012737 077777 001312 MOV #77777,TIME ;RESET TIMER
2430 012570 005305 DEC R5 ;DEC PRINT DELAY
2431 012572 001362 BNE 40$ ;CHECK AGAIN
2432 012574 000137 012446 JMP FLOP ;TELL AGAIN TO FLIP SWITCH
    
```

```

2434 .SBTTL TEST 64 STRING OF 200 DATIS BURST MODE XFERS (SWITCH FLIPPED)
2435 :*****
2436 :TEST 64 STRING OF 200 DATI'S BURST MODE XFERS (SWITCH FLIPPED)
2437 :CABLE MODE(WITH WRAP AROUND CABLE IN USER SLOTS)
2438 :*****
2439 012600 000004 SWT1: SCOPE
2440 012602 005077 166262 CLR @CSR ;FORCE TO BE CSR #1
2441 012606 012737 000200 001156 MOV #200,BUFLEN ;LENGTH OF BUFFER=200
2442 012614 004737 007714 JSR PC,LODBUF ;LOAD THE BUFFER WITH INCREMENTING PATTERN
2443 012620 013737 001156 001164 MOV BUFLEN,WCLEN ;PREPARE NUMBER FOR WCR
2444 012626 005437 001164 NEG WCLEN ;2'S COMPLEMENT OF BUFLEN
2445 012632 013777 001164 166224 MOV WCLEN,@WCR ;SET UP WCR
2446 012640 013777 001152 166220 MOV INBUF,@BAR ;SET UP BAR
2447 012646 012777 177777 166216 MOV #-1,@BDR ;MAINT AIDE
2448 012654 012777 012740 166254 MOV #SWINT1,@DRINV ;INT VECTOR
2449 012662 013777 001074 166250 MOV DRINL,@DRVS ;INT VECTOR TO PRIORITY DRINL
2450 012670 005077 166106 CLR @PSW ;LET THE DR11-W INTERRUPT
2451 012674 012777 000100 166166 MOV #100,@CSR ;IE
2452 012702 052777 000401 166160 BIS #401,@CSR ;CYCLE,GO
2453 012710 012737 001000 001312 MOV #1000,TIME ;WAIT FOR INTERRUPT
2454 012716 005337 001312 1$: DEC TIME
2455 012722 001375 BNE 1$
2456 012724 104000 HLT
2457 012726 000005 RESET ;IF NO INTERRUPT,CLEAR ANY OUTSTANDING
2458 ;INTERRUPT REQUEST IN BOARD HDWR.
2459 012730 004737 010156 JSR PC,NORMAL
2460 012734 000137 012760 JMP SWT2
2461 012740 022626 SWINT1: CMP (SP)+,(SP)+
2462 012742 004737 010016 JSR PC,INTA ;CLR IE,CHECK ERROR,READY,WC=0,&BAR
2463 012746 022777 000177 166116 CMP #177,@BDR ;CHECK THAT WORD #200 OF INBUF IS IN BAR
2464 012754 001401 BEQ .+4 ;IS IT?
2465 012756 104000 HLT ;BAD DATA IN BDR
    
```

```

2467 .SBTTL TEST 65 STRING OR 200 DATOS BURST MODE XFERS (SWITCH FLIPPED)
2468 :*****
2469 :TEST 65 STRING OR 200 DATO'S BURST MODE XFERS (SWITCH FLIPPED)
2470 :CABLE MODE(WITH WRAP AROUND CABLE IN USER SLOTS)
2471 :*****
2472 012760 000004 SW2: SCOPE
2473 012762 005077 166102 CLR @CSR ;FORCE TO BE CSR #1
2474 012766 012737 000200 001156 MOV #200,BUFLEN ;LENGTH OF BUFFER=200
2475 012774 004737 007714 JSR PC,LODBUF ;LOAD THE BUFFER WITH INCREMENTING PATTERN
2476 013000 013737 001156 001164 MOV BUFLEN,WCLEN ;PREPARE NUMBER FOR WCR
2477 013006 005437 001164 NEG WCLEN ;2'S COMPLEMENT OF BUFLEN
2478 013012 013777 001164 166044 MOV WCLEN,@WCR ;SET UP WCR
2479 013020 013777 001152 166040 MOV INBUF,@BAR ;SET UP BAR
2480 013026 012777 052525 166036 MOV #52525,@BDR ;SET UP BDR
2481 013034 012777 013120 166074 MOV #SWINT2,@DRINV ;INTERRUPT VECTOR
2482 013042 013777 001074 166070 MOV DRINL,@DRVS ;INTERRUPT VECTOR TO PRIORITY DRINL
2483 013050 005077 165726 CLR @PSW ;LET THE DR11-W INTERRUPT
2484 013054 012777 000102 166006 MOV #102,@CSR ;IE, FNCT1
2485 013062 052777 000401 166000 BIS #401,@CSR ;CYCLE,GO
2486 013070 012737 001000 001312 MOV #1000,TIME ;WAIT FOR INTERRUPT
2487 013076 005337 001312 1$: DEC TIME
2488 013102 001375 BNE 1$
2489 013104 104000 HLT
2490 013106 000005 RESET ;IF NO INTERRUPT,CLEAR ANY OUTSTANDING
2491 ;INTERRUPT REQUEST IN BOARD HDWR.
2492 013110 004737 010156 JSR PC,NORMAL
2493 013114 000137 013132 JMP END
2494 013120 022626 SWINT2: CMP (SP)+,(SP)+
2495 013122 004737 010016 JSR PC,INTA ;CLEAR IE,CHECK ERROR,READY,WC=0,& BAR
2496 013126 004737 010200 JSR PC,DATOCK ;CHECK INBUF
2497
2498 ;THIS ENDS CABLE MODE TESTING
2499
    
```

```

2501
2502
2503
2504 013132 012737 000207 177566 END: MOV #207,@#177566 ;RING BELL
2505 013140 105737 177564 TSTB @#177564
2506 013144 100375 BPL .-4
2507 013146 005277 165642 INC @PNTPAS ;INCREMENT PASS COUNT FOR THIS BOARD
2508 013152 005737 001316 TST HLTDET ;ERROR OCCUR THIS PASS?
2509 013156 001402 BEQ 2$ ;NO
2510 013160 005277 165632 INC @PNTERR ;YES INCR. ERROR COUNT FOR THIS BRD.
2511 013164 032737 000001 001314 2$: BIT #1,CABLE ;HAVE CABLE TESTS BEEN DONE?
2512 013172 001405 BEQ 1$ ;NO
2513 013174 012702 014755 MOV #MANCBL,R2 ;YES
2514 013200 004737 015372 JSR PC,TTOUT
2515 013204 000404 BR END1
2516 013206 012702 014723 1$: MOV #MANT,R2
2517 013212 004737 015372 JSR PC,TTOUT
2518 013216 END1:
2519 013216 012702 014622 MOV #SENPAS,R2 ;PRINT 'END PASS'
2520 013222 004737 015372 JSR PC,TTOUT
2521 013226 017746 165562 MOV @PNTPAS,-(SP) ;;SAVE @PNTPAS FOR TYPEOUT
2522 013232 104405 TYPDS ;;GO TYPE--DECIMAL ASCII WITH SIGN
2523 013234 104401 001354 TYPE ,SENULL
2524 013240 013702 000042 MOV @#42,R2
2525 013246 000005 BEQ NXTBRD
2526 013250 004712 RESET
2527 013252 000240 SENDAD: JSR PC,(R2)
2528 013254 000240 NOP
2529 013256 000240 NOP
2530 013260 062737 000002 001010 NXTBRD: ADD #2,PNTRAD ;POINT TO NEXT BOARD FOR NEXT PASS
2531 013266 062737 000002 001012 ADD #2,PNTVAD
2532 013274 062737 000002 001014 ADD #2,PNTPAS
2533 013302 062737 000002 001016 ADD #2,PNTERR
2534 013310 000137 001604 JMP BEGIN
2535
2536
2537
2538
2539
2540
2541
    
```

```

2543
2544
2545
2546
2547 013314 032777 000001 165546 PRINT: BIT #BIT0,@CSR ;CHECK WHICH CSR
2548 013322 001422 BEQ 10$ ;WE ARE IN CSR #1
2549 013324 017737 165540 001142 MOV @CSR,EIR ;SAVE CSR #2
2550 013332 013701 001070 MOV CSR,R1 ;ADDRESS OF CSR INTO R1
2551 013336 062701 000001 ADD #1,R1 ;R1 NOW HAS ADD OF UPPER BYTE
2552 013342 142701 000200 BICB #200,R1 ;RETURN TO CSR #1
2553 013346 017705 165516 MOV @CSR,R5 ;SAVE CSR IN R5
2554 013352 105737 001143 TSTB EIR+1 ;TEST FOR ANY ERROR
2555 013356 001406 BEQ 20$ ;NO ERROR BIT(S) SET
2556 013360 052705 100000 BIS #BIT15,R5 ;ERROR WAS SET - RESTORE IT
2557 013364 000137 013374 JMP 20$ ;CONTINUE
2558 013370 017705 165474 10$: MOV @CSR,R5 ;SAVE CSR IN R5
2559 013374 012737 000001 001316 20$: MOV #1,HLTDET ;SAYS ERROR OCCURRED THIS PASS
2560 013402 004737 014144 JSR PC,CKSWR ;
2561 013406 037727 165714 020000 BIT @SWR,#20000 ;TEST FOR INHIBIT PRINT OUT
2562 013414 001142 BNE 1$ ;IF SO, BRANCH OVER PRINT
2563 013416 012637 013770 MOV (SP)+,SAVPC ;PC OF FAILING ROUTINE
2564 013422 012637 013772 MOV (SP)+,SAVCC ;CC OF ERROR CONDITION
2565 013426 024646 CMP -(SP),-(SP) ;REPOSITION THE STACK
2566 013430 012777 000215 165540 MOV #215,@TPB ;CR
2567 013436 105777 165532 TSTB @TPS
2568 013442 100375 BPL -4
2569 013444 012777 000212 165524 MOV #212,@TPB ;LINE FEED
2570 013452 105777 165516 TSTB @TPS
2571 013456 100375 BPL -4
2572 013460 010237 013762 MOV R2,SAVR2 ;SAVE R2
2573 013464 010337 013764 MOV R3,SAVR3 ;SAVE R3
2574 013470 010437 013766 MOV R4,SAVR4 ;SAVE R4
2575 013474 104401 001351 TYPE ,SCLF
2576 013500 032737 000001 001144 BIT #BIT0,BORW
2577 013506 001403 BEQ 15$
2578 013510 104401 015014 TYPE ,MSG4
2579 013514 000402 BR 25$
2580 013516 104401 015163 15$: TYPE ,MSG7
2581 013522 104401 001351 25$: TYPE ,SCLF
2582 013526 013746 001320 MOV TESTNO,-(SP) ;;SAVE TESTNO FOR TYPEOUT
013532 104402 TYPOC ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
2583 013534 104401 001354 TYPE ,SENULL
2584 013540 012777 000240 165430 MOV #240,@TPB
2585 013546 013746 013770 MOV SAVPC,-(SP) ;;SAVE SAVPC FOR TYPEOUT
013552 104402 TYPOC ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
2586 013554 104401 001354 TYPE ,SENULL
2587 013560 012777 000240 165410 MOV #240,@TPB
2588 013566 105777 165402 TSTB @TPS ;SPACE BETWEEN WORDS
2589 013572 100375 BPL -4
2590 013574 013746 013772 MOV SAVCC,-(SP) ;;SAVE SAVCC FOR TYPEOUT
013600 104402 TYPOC ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
2591 013602 104401 001354 TYPE ,SENULL
2592 013606 012777 000240 165362 MOV #240,@TPB ;PRINT SPACE
2593 013614 105777 165354 TSTB @TPS ;PRINTER DONE
2594 013620 100375 BPL -4 ;BRANCH WHEN NOT DONE
2595 013622 012737 013650 000004 MOV #3$,BUSFRR ;
2596 013630 010546 MOV R5,-(SP) ;;SAVE R5 FOR TYPEOUT
  
```

```

2597 013632 104402          TYP0C          ;:GO TYPE--OCTAL ASCII(ALL DIGITS)
2598 013634 104401 001354  TYPE          , $ENULL
2599 013640 012737 000006 000004  MOV          #6,BUSERR
2600 013646 000405          BR           4$
2601 013650 022626          CMP          (SP)+,(SP)+
2602 013652 012737 000006 000004  MOV          #6,BUSERR
2603 013660 000401          BR           5$
2604 013662 000240          4$: NOP
2605 013664 005737 001314          5$: TST      CABLE          ;PRINT CABLE MSSGE?
2606 013670 001404          BEQ          2$          ;NO
2607 013672 104401 015057          TYPE          ,MSG5
2608 013676 104401 015131          TYPE          ,MSG6
2609 013702 013702 013762          2$: MOV      SAVR2,R2
2610 013706 013703 013764          MOV      SAVR3,R3
2611 013712 013704 013766          MOV      SAVR4,R4
2612 013716 004737 014144          JSR      PC,CKSWR
2613 013722 005777 165400          1$: TST      @SWR          ;CHECK SR FOR HALT SWITCH
2614 013726 100001          BPL          .+4
2615 013730 000000          HALT
2616 013732 023737 000042 000046          CMP      @#42,@#46          ;HALT ON ERROR UP IN SWR
2617 013740 001001          BNE          .+4          ;ARE WE IN ACT11 AUTO MODE?
2618 013742 000000          HALT          ;BRANCH ON NO
2619 013744 037727 165356 000400          BIT      @SWR,#400          ;HALT ON ERROR IF IN ACT11 AUTO MODE
2620 013754 000137 013132          BEQ      CONTIN          ;GO TO NEXT BOARD?
2621          JMP      END          ;IF NO,CONTINUE THIS BOARD
2622 013760 000002          CONTIN: RTI
2623 013762 000000          SAVR2: 0
2624 013764 000000          SAVR3: 0
2625 013766 000000          SAVR4: 0
2626 013770 000000          SAVPC: 0
2627 013772 000000          SAVCC: 0
2628
    
```


2630
2631
2632
2633
2634
2635
2636
2637
2638
2639
2640
2641
2642

013774 004737 014144
014000 032777 040000 165320
014006 001003
014010 011637 014126
014014 000002
014016 022606
014020 012677 164756
014024 000177 000076

```
.....  
: SCOPE LOOP ROUTINE ENTERED BY USER TRAP  
:.....  
SCOPEA: JSR PC,CKSWR  
          BIT #40000,@SWR  
          BNE SCOPEB ;SCOPE, BIT IS A ONE  
          MOV @SP,RETURN ;NO - SAVE PC FOR NEXT TIME  
          RTI ;RETURN IN SEQUENCE  
SCOPEB: CMP (SP)+,SP ;REPOSITION THE STACK  
          MOV (SP)+,@PSW  
          JMP @RETURN ;SCOPE RETURN
```

```

2644
2645
2646
2647
2648
2649
2650 014030 004737 014144 SCOPEC: JSR PC,CKSWR
2651 014034 032777 040000 165264 BIT #40000,@SWR ;TEST SR FOR SCOPE
2652 014042 001365 BNE SCOPEB ;YES SCOPE
2653 014044 005777 164744 TST @PNTPAS ;FIRST @PASS (@PASCNT-0) ?
2654 014050 001415 BEQ SCOPEG ;BR IF YES, INHIBIT ITERATIONS
2655 014052 004737 014144 JSR PC,CKSWR
2656 014056 032777 004000 165242 BIT #4000,@SWR ;TEST FOR ITERATION
2657 014064 001007 BNE SCOPEG ;INHIBIT ITERATION
2658 014066 023737 014124 014122 CMP SCOPEF,ICOUNT
2659 014074 001403 BEQ SCOPEG ;EXIT - DONE
2660 014076 005237 014124 INC SCOPEF ;INCREMENT COUNT
2661 014102 000745 BR SCOPEB ;LOOP SOME MORE
2662 014104 005037 014124 SCOPEG: CLR SCOPEF ;CLEAR COUNT
2663 014110 011637 014126 MOV @SP,RETURN ;SAVE SCOPE RETURN POINTER
2664 014114 005237 001320 INC TESTNO
2665 014120 000002 RTI ;RETURN INLINE-NEXT TEST
2666 014122 004000 ICOUNT: 4000
2667 014124 000000 SCOPEF: 0 ;COUNT LOCATION FOR ITERATION LOOP
2668 014126 001604 RETURN: BEGIN ;ADDRESS OF LAST TEST
2669 .FVFN
2670 014130 000137 000200 JMP 200
2671
    
```

```

2673
2674
2675
2676
2677
2678 014134 000000
2679 014136 000000
2680 014140 000000
2681 014142 000000
2682
2683 014144 105777 165020
2684 014150 100152
2685 014152 017737 165014 014142
2686 014160 042737 177600 014142
2687 014166 005737 000042
2688 014172 001012
2689 014174 022737 000023 014142
2690 014202 001006
2691 014204 012702 014560
2692 014210 004737 015372
2693 014214 000137 021230
2694 014220 022737 000176 001326 1$:
2695 014226 001123
2696
2697 014230 022737 000007 014142
2698 014236 001117
2699 014240 012702 014552
2700 014244 004737 015372
2701 014250 012702 014573
2702 014254 004737 015372
2703 014260 017746 165042
2704 014264 104402
2705 014266 104401 001354
2706 014272 012702 014603
2707 014276 004737 015372
2708 014302 005037 014134
2709 014306 005037 014134
2710 014312 012737 000007 014136
2711 014320 004737 014500 1$:
2712 014324 042737 177600 014142
2713 014332 122737 000025 014142
2714 014340 001001
2715 014342 000742 3$:
2716 014344 122737 000015 014142 2$:
2717 014352 001011
2718 014354 012702 014567
2719 014360 004737 015372
2720 014364 022737 000007 014136
2721 014372 001036
2722 014374 000440 8$:
2723 014376 122737 000060 014142 4$:
2724 014404 003004
2725 014406 122737 000067 014142
2726 014414 002005
2727 014416 012702 014614 5$:
2728 014422 004737 015372
2729 014426 000745

*****
CHECK SWITCH REGISTER ROUTINE. CHECKS FOR ^G TO ALLOW CHANGING
OF LOC. 176.
*****
TEMPST: .WORD 0
COUNT: .WORD 0
RDSW: .WORD 0
T . . . .WORD 0

CKSWR: TSTB @TKS ;YES, WAIT FOR
BPL OUT ;READY, GET CHARACTER
MOV @TKB, TIB ;AND STRIP OFF
BIC #177600, TIB ;THE GARBAGE
TST @42 ;IF AUTO MODE, DON'T CHECK
BNE 1$ ;FOR CNTRL S
CMP #23, TIB
BNE 1$ ;NOT CNTRL S
MOV #SCNTS, R2
JSR PC, TTOUT
JMP RUNSUM ;LEAVE PROGRAM AND PRINT RUN SUMMARY
CMP #SWREG, SWR ;SOFTW. SWITCH REG PRESENT?
BNE OUT

CMP #7, TIB ;IS IT A <^G>
BNE OUT
MOV #SCNTG, R2
JSR PC, TTOUT
CNTLU: MOV #OUTSWR, R2
JSR PC, TTOUT
MOV @SWR, -(SP) ;;SAVE @SWR FOR TYPEOUT
;;GO TYPE--OCTAL ASCII(ALL DIGITS)
TYPE , $ENULL
MOV #INNEW, R2
JSR PC, TTOUT
CLR @TEMPST
$READ: CLR TEMPST
MOV #7, COUNT
1$: JSR PC, TTIN ;GO READ A CHARACTER
BIC #177600, TIB ;STRIP OFF GARBAGE
CMPB #25, TIB ;IS IT A ^U?
BNE 2$ ;BRANCH IF NOT
BR CNTLU ;START OVER
2$: CMPB #15, TIB ;IS IT A <CR>?
BNE 4$ ;BRANCH IF NOT
MOV #CARLF, R2
JSR PC, TTOUT
CMP #7, COUNT ;WAS IT FIRST CHARACTER
BNE 7$ ;CHANGE SWR IF NOT FIRST ONE
BR OUT ;GET OUT
7$:
8$: BR OUT
4$: CMPB #60, TIB
BGT 5$
CMPB #67, TIB
BGE 6$
5$: MOV #SQUEST, R2
JSR PC, TTOUT
BR 3$ ;START OVER IF NOT LEGAL CHARACTER
3$:

```

2729	014430	006337	014134		6\$:	ASL	TEMPST		
2730	014434	006337	014134			ASL	TEMPST		
2731	014440	006337	014134			ASL	TEMPST		
2732	014444	142737	000060	014142		BICB	#60,TIB	:	GET NITTY-GRITTY
2733	014452	153737	014142	014134		BISB	TIB,TEMPST		
2734	014460	005337	014136			DEC	COUNT	:	ONLY WANT 6 DIGITS
2735	014464	001754				BEQ	5\$		
2736	014466	000714				BR	1\$		
2737	014470	013777	014134	164630	7\$:	MOV	TEMPST,@SWR	:	CHANGE SWITCH REGISTER CONTENTS
2738	014476	000207			OUT:	RTS	PC	:	RETURN TO PROGRAM

```

2740
2741
2742
2743
2744
2745
2746 014500 005077 164464      TTIN:  CLR      @TKS
2747 014504 005077 164462      CLR      @TKB
2748 014510 005037 014142      CLR      TIB
2749 014514 005277 164450      INC      @TKS
2750 014520 105777 164444      TTIN1:  TSTB    @TKS
2751 014524 100375              BPL      TTIN1
2752 014526 017737 164440 014142  MOV      @TKB,TIB
2753 014534 105777 164434      TTIN2:  TSTB    @TPS
2754 014540 100375              BPL      TTIN2
2755 014542 113777 014142 164426  MOVB     TIB,@TPB
2756
2757 014550 000207              RTS      PC
2758 014552      137      136      107  $CNTG.  .ASCIZ  '_^G &'
      014555      040      046      000
2759 014560      040      137      136  $CNTS:  .ASCIZ  /_ ^S &/
      014563      123      040      046
      014566      000
2760 014567      137      040      046  CARLF:  .ASCIZ  '_ &'
      014572      000
2761 014573      137      123      127  OUTSWR: .ASCIZ  '_SWR= &'
      014576      122      075      040
      014601      046      000
2762 014603      040      040      116  INNEW:  .ASCIZ  ' NEW= &'
      014606      105      127      075
      014611      040      046      000
2763 014614      137      077      040  $QUEST: .ASCIZ  ' _? _&'
      014617      137      046      000
2764 014622      040      137      040  $ENPAS: .ASCIZ  ' _ END PASS # &'
      014625      105      116      104
      014630      040      120      101
      014633      123      123      040
      014636      040      040      043
      014641      040      040      040
      014644      046      000
2765 014646      137      040      103  $TITLE: .ASCIZ  '_ CZDRL-DR11W GEN NPR INTFC LOGIC TEST  __&'
      014651      132      104      122
      014654      114      055      104
      014657      122      061      061
      014662      127      040      107
      014665      105      116      040
      014670      116      120      122
      014673      040      111      116
      014676      124      106      103
      014701      040      114      117
      014704      107      111      103
      014707      040      124      105
      014712      123      124      040
      014715      040      040      137
      014720      137      046      000
2766 014723      137      124      105  MANT:   .ASCIZ  /_TEST MODE: MAINT ONLY &/
      014726      123      124      040
    
```

	014731	115	117	104	
	014734	105	072	040	
	014737	115	101	11*	
	014742	116	124	040	
	014745	117	116	114	
	014750	131	040	040	
	014753	046	000		
2767	014755	015	012	124	MANCBL: .ASCIZ<15><12>/TEST MODE: MAINT AND CABLF/<15><12>
	014760	105	123	124	
	014763	040	115	117	
	014766	104	105	072	
	014771	040	115	101	
	014774	111	116	124	
	014777	040	101	116	
	015002	104	040	103	
	015005	101	102	114	
	015010	105	015	012	
	015013	000			
2768	015014	124	105	123	MSG4: .ASCIZ /TESTNO ERRFC PSW DR11W STATUS /
	015017	124	116	117	
	015022	040	040	105	
	015025	122	122	120	
	015030	103	040	040	
	015033	120	123	127	
	015036	040	040	040	
	015041	104	122	061	
	015044	061	127	040	
	015047	123	124	101	
	015052	124	125	123	
	015055	040	000		
2769	015057	015	012	120	MSG5: .ASCIZ <15><12>/PROGRAM IS PERFORMING CABLE MODE TEST/<15><12>
	015062	122	117	107	
	015065	122	101	115	
	015070	040	111	123	
	015073	040	120	105	
	015076	122	106	117	
	015101	122	115	111	
	015104	116	107	040	
	015107	103	101	102	
	015112	114	105	040	
	015115	115	117	104	
	015120	105	040	124	
	015123	105	123	124	
	015126	015	012	000	
2770	015131	125	123	105	MSG6: .ASCIZ /USER CABLE SHOULD BE IN/<15><12>
	015134	122	040	103	
	015137	101	102	114	
	015142	105	040	123	
	015145	110	117	125	
	015150	114	104	040	
	015153	102	105	040	
	015156	111	116	015	
	015161	012	000		
2771	015163	015	012	124	MSG7: .ASCIZ <15><12>/TESTNO ERRPC PSW DR11B STATUS/<15><12>
	015166	105	123	124	
	015171	116	117	040	
	015174	040	105	122	

	015177	122	120	103	
	015202	040	040	120	
	015205	123	127	040	
	015210	040	104	122	
	015213	061	061	102	
	015216	040	123	124	
	015221	101	124	125	
	015224	123	015	012	
	015227	000			
2772	015230	015	012	104	MSG8: .ASCIZ <15><12>/DIAGNOSTIC TESTING OF DR11B NOW IN PROGRESS/<15><12>
	015233	111	101	107	
	015236	116	117	123	
	015241	124	111	103	
	015244	040	124	105	
	015247	123	124	111	
	015252	116	107	040	
	015255	117	106	040	
	015260	104	122	061	
	015263	061	102	040	
	015266	116	117	127	
	015271	040	111	116	
	015274	040	120	122	
	015277	117	107	122	
	015302	105	123	123	
	015305	015	012	000	
2773	015310	015	012	104	MSG9: .ASCIZ <15><12>/DIAGNOSTIC TESTING OF DR11W NOW IN PROGRESS/<15><12>
	015313	111	101	107	
	015316	116	117	123	
	015321	124	111	103	
	015324	040	124	105	
	015327	123	124	111	
	015332	116	107	040	
	015335	117	106	040	
	015340	104	122	061	
	015343	061	127	040	
	015346	116	117	127	
	015351	040	111	116	
	015354	040	120	122	
	015357	117	107	122	
	015362	105	123	123	
	015365	015	012	000	
2774					.EVEN
2775					
2776					
2777	015370	000000			OFL: 0 ;FIRST CHAR FLAG
2778					
2779					

```

2781
2782
2783
2784
2785
2786 015372 105712
2787 015374 001403
2788 015376 122712 000046
2789 015402 001005
2790 015404 042777 000100 163562 1$:
2791 015412 005002
2792 015414 000207
2793 015416 122712 000137 .EMPTY:
2794 015422 001411
2795 015424 122712 000041
2796 015430 001414
2797 015432 105777 163536 1$:
2798 015436 100375
2799 015440 112277 163532
2800 015444 000752
2801 015446 005202 .RET:
2802 015450 010237 021402
2803 015454 012702 015470
2804 015460 000767
2805 015462 013702 021402 .REST:
2806 015466 000741
2807
2808 015470 015 012 041 .RETR:
2809
    
```

```

*****
TTY ASCII OUTPUT ROUTINE
*****
TTOUT: TSTB (R2) ;CHECK FOR NULL CHARACTER
        BEQ 1$ ;IF NOT, TYPE THE CHARACTER
        CMPB #'$',(R2) ;CHECK FOR TERMINATOR
        BNE .EMPTY
        BIC #100,@TPS
        CLR R2 ;CLEAR POINTER TO CHARACTER
        RTS PC ;RETURN
        .EMPTY: CMPB #'$',(R2) ;CRLF CHAR?
        BEQ .RET
        CMPB #'$',(R2) ;CHECK FOR RETURN TERMINATOR
        BEQ .REST
        TSTB @TPS
        BPL 1$
        MOVB (R2)+,@TPB ;TYPE CHARACTER
        BR TTOUT
        .RET: INC R2
        MOV R2,@SAV ;SET UP NEW POINTER
        MOV #.RETR,R2
        BR .RET-6
        .REST: MOV @SAV,R2
        BR TTOUT
        .RETR: .BYTE 15,12,'
        .EVEN
    
```


2811
2812
2813

```

.SBTTL
.SBTTL SYSMAC ROUTINES
.SBTTL TYPE ROUTINE
:*****
:*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
:*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
:*NOTE1: $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
:*NOTE2: $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
:*NOTE3: $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
:*
:*CALL:
:*1) USING A TRAP INSTRUCTION
:* TYPE ,MESADR ;:MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
:*OR
:* TYPE
:* MESADR
:*
015474 105737 001343 $TYPE: TSTB $TPFLG ;:IS THERE A TERMINAL?
015500 100002 BPI 1$ ;:BR IF YES
015502 000000 HALT ;:HALT HERE IF NO TERMINAL
015504 000407 BR 3$ ;:LEAVE
015506 010046 1$: MOV R0,-(SP) ;:SAVE R0
015510 017600 000002 2$: MOV @2(SP),R0 ;:GET ADDRESS OF ASCIZ STRING
015514 112046 2$: MOVB (R0)+,-(SP) ;:PUSH CHARACTER TO BE TYPED ONTO STACK
015516 001005 BNE 4$ ;:BR IF IT ISN'T THE TERMINATOR
015520 005726 TST (SP)+ ;:IF TERMINATOR POP IT OFF THE STACK
015522 012600 60$: MOV (SP)+,R0 ;:RESTORE R0
015524 062716 3$: ADD #2,(SP) ;:ADJUST RETURN PC
015530 000002 RTI ;:RETURN
015532 122716 000011 4$: CMPB #HT,(SP) ;:BRANCH IF <HT>
015536 001430 BEQ 8$
015540 122716 000200 CMPB #CRLF,(SP) ;:BRANCH IF NOT <CRLF>
015544 001006 BNE 5$
015546 005726 TST (SP)+ ;:POP <CR><LF> EQUIV
015550 104401 TYPE ;:TYPE A CR AND LF
015552 001351 $CRLF
015554 105037 015710 CLRB $CHARCNT ;:CLEAR CHARACTER COUNT
015560 000755 BR 2$ ;:GET NEXT CHARACTER
015562 004737 015644 5$: JSR PC,$TYPEC ;:GO TYPE THIS CHARACTER
015566 123726 001342 6$: CMPB $FILLC,(SP)+ ;:IS IT TIME FOR FILLER CHARS.?
015572 001350 BNE 2$ ;:IF NO GO GET NEXT CHAR.
015574 013746 001340 MOV $NULL,-(SP) ;:GET # OF FILLER CHARS. NEEDED
;:AND THE NULL CHAR.
015600 105366 000001 7$: DECB 1(SP) ;:DOES A NULL NEED TO BE TYPED?
015604 002770 BLT 6$ ;:BR IF NO--GO POP THE NULL OFF OF STACK
015606 004737 015644 JSR PC,$TYPEC ;:GO TYPE A NULL
015612 105337 015710 DECB $CHARCNT ;:DO NOT COUNT AS A COUNT
015616 000770 BR 7$ ;:LOOP
;HORIZONTAL TAB PROCESSOR
015620 112716 000040 8$: MOVB #' ,(SP) ;:REPLACE TAB WITH SPACE
015624 004737 015644 9$: JSR PC,$TYPEC ;:TYPE A SPACE
015630 132737 000007 015710 BITB #7,$CHARCNT ;:BRANCH IF NOT AT
015636 001372 BNE 9$ ;:TAB STOP
015640 005726 TST (SP)+ ;:POP SPACE OFF STACK
015642 000724 BR 2$ ;:GET NEXT CHARACTER
015644 105777 163464 $TYPEC: TSTB @5$PS ;:WAIT UNTIL PRINTER IS READY
015650 100375 BPL $TYPEC

```

```
015652 116677 000002 163456      MOVB 2(SP),@STPB      ;;LOAD CHAR TO BE TYPED INTO DATA REG.
015660 122766 000015 000002      CMPB #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
015666 001003                      BNE 1$               ;;BRANCH IF NO
015670 105037 015710                      CLRB $CHARCNT       ;;YES--CLEAR CHARACTER COUNT
015674 000406                      BR $TYPEX           ;;EXIT
015676 122766 000012 000002 1$:  CMPB #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
015704 001402                      BEQ $TYPEX         ;;BRANCH IF YES
015706 105227                      INCB (PC)+         ;;COUNT THE CHARACTER
015710 000000                      $CHARCNT: .WORD 0  ;;CHARACTER COUNT STORAGE
015712 000207                      $TYPEX: RTS       PC
```

2815

```

.SBTTL BINARY TO OCTAL (ASCII) AND TYPE
*****
*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
*OCTAL (ASCII) NUMBER AND TYPE IT.
*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
*CALL:
*   MOV     NUM,-(SP)      ;;NUMBER TO BE TYPED
*   TYPOS   ;;CALL FOR TYPEOUT
*   .BYTE  N              ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
*   .BYTE  M              ;;M=1 OR 0
*                           ;;1-TYPE LEADING ZEROS
*                           ;;0=SUPPRESS LEADING ZEROS
*$TYPON---ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
*$TYPOS OR $TYPOC
*CALL:
*   MOV     NUM,-(SP)      ;;NUMBER TO BE TYPED
*   TYPON   ;;CALL FOR TYPEOUT
*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
*CALL:
*   MOV     NUM,-(SP)      ;;NUMBER TO BE TYPED
*   TYPOC   ;;CALL FOR TYPEOUT
015714 017646 000000 $TYPOS: MOV @ (SP),-(SP) ;;PICKUP THE MODE
015720 116637 000001 016137 MOVB 1(SP),%0FILL ;;LOAD ZERO FILL SWITCH
015726 112637 016141 MOVB (SP)+,%0MODE+1 ;;NUMBER OF DIGITS TO TYPE
015732 062716 000002 ADD #2,(SP) ;;ADJUST RETURN ADDRESS
015736 000406 BR $TYPON
015740 112737 000001 016137 $TYPOC: MOVB #1,%0FILL ;;SET THE ZERO FILL SWITCH
015746 112737 000006 016141 MOVB #6,%0MODE+1 ;;SET FOR SIX(6) DIGITS
015754 112737 000005 016136 $TYPON: MOVB #5,%0CNT ;;SET THE ITERATION COUNT
015762 010346 MOV R3,-(SP) ;;SAVE R3
015764 010446 MOV R4,-(SP) ;;SAVE R4
015766 010546 MOV R5,-(SP) ;;SAVE R5
015770 113704 016141 MOVB %0MODE+1,R4 ;;GET THE NUMBER OF DIGITS TO TYPE
015774 005404 NEG R4
015776 062704 000006 ADD #6,R4 ;;SUBTRACT IT FOR MAX. ALLOWED
016002 110437 016140 MOVB R4,%0MODE ;;SAVE IT FOR USE
016006 113704 016137 MOVB %0FILL,R4 ;;GET THE ZERO FILL SWITCH
016012 016605 000012 MOV #2(SP),R5 ;;PICKUP THE INPUT NUMBER
016016 005003 CLR R3 ;;CLEAR THE OUTPUT WORD
016020 006105 1$: ROL R5 ;;ROTATE MSB INTO 'C'
016022 000404 BR 3$ ;;GO DO MSB
016024 006105 2$: ROL R5 ;;FORM THIS DIGIT
016026 006105 ROL R5
016030 006105 ROL R5
016032 010503 MOV R5,R3
016034 006103 3$: ROL R3 ;;GET LSB OF THIS DIGIT
016036 105337 016140 DECB %0MODE ;;TYPE THIS DIGIT?
016042 100016 BPL 7$ ;;BR IF NO
016044 042703 177770 BIC #177770,R3 ;;GET RID OF JUNK
016050 001002 BNE 4$ ;;TEST FOR 0
016052 005704 TST R4 ;;SUPPRESS THIS 0?
016054 001403 BEQ 5$ ;;BR IF YES
016056 005204 4$: INC R4 ;;DON'T SUPPRESS ANYMORE 0'S
016060 052703 000060 BIS #'0,R3 ;;MAKE THIS DIGIT ASCII
016064 052703 000040 5$: BIS #' ,R3 ;;MAKE ASCII IF NOT ALREADY

```

```

016070 110337 016134      MOVB   R3,8$      ;;SAVE FOR TYPING
016074 104401 016134      TYPE   ,8$       ;;GO TYPE THIS DIGIT
016100 105337 016136      7$:   DECB   $OCNT  ;;COUNT BY 1
016104 003347          BGT    2$        ;;BR IF MORE TO DO
016106 002402          BLT    6$        ;;BR IF DONE
016110 005204          INC    R4        ;;INSURE LAST DIGIT ISN'T A BLANK
016112 000744          BR     2$        ;;GO DO THE LAST DIGIT
016114 012605      6$:   MOV   (SP)+,R5  ;;RESTORE R5
016116 012604          MOV   (SP)+,R4  ;;RESTORE R4
016120 012603          MOV   (SP)+,R3  ;;RESTORE R3
016122 016666 000002 000004  MOV   2(SP),4(SP) ;;SET THE STACK FOR RETURNING
016130 012616          MOV   (SP)+,(SP)
016132 000002          RTI                    ;;RETURN
016134 000          8$:   .BYTE 0      ;;STORAGE FOR ASCII DIGIT
016135 000          .BYTE 0      ;;TERMINATOR FOR TYPE ROUTINE
016136 000      $OCNT: .BYTE 0  ;;OCTAL DIGIT COUNTER
016137 000      $OFILL: .BYTE 0 ;;ZERO FILL SWITCH
016140 000000      $OMODE: .WORD 0  ;;NUMBER OF DIGITS TO TYPE

```

2817

```

.SBTTL CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
*****
*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
*REPLACED WITH SPACES.
*CALL:
*
*      MOV      NUM,-(SP)      ;;PUT THE BINARY NUMBER ON THE STACK
*      TYPDS    ;;GO TO THE ROUTINE
$TYPDS:
016142      MOV      R0,-(SP)      ;;PUSH R0 ON STACK
016142      MOV      R1,-(SP)      ;;PUSH R1 ON STACK
016144      MOV      R2,-(SP)      ;;PUSH R2 ON STACK
016146      MOV      R3,-(SP)      ;;PUSH R3 ON STACK
016150      MOV      R5,-(SP)      ;;PUSH R5 ON STACK
016152      MOV      #20200,-(SP)  ;;SET BLANK SWITCH AND SIGN
016154      MOV      20(SP),R5     ;;GET THE INPUT NUMBER
016160      BPL      1$           ;;BR IF INPUT IS POS.
016164      NEG      R5           ;;MAKE THE BINARY NUMBER POS.
016166      MOVVB   #'-,1(SP)     ;;MAKE THE ASCII NUMBER NEG.
016170      CLR      R0           ;;ZERO THE CONSTANTS INDEX
016176      MOV      #SDBLK,R3     ;;SETUP THE OUTPUT POINTER
016200      MOVVB   #' ,(R3)+     ;;SET THE FIRST CHARACTER TO A BLANK
016204      CLR      R2           ;;CLEAR THE BCD NUMBER
016210      MOV      $DTBL(R0),R1  ;;GET THE CONSTANT
016212      SUB      R1,R5        ;;FORM THIS BCD DIGIT
016216      BLT     4$           ;;BR IF DONE
016220      INC     R2           ;;INCREASE THE BCD DIGIT BY 1
016222      BR      3$
016224      ADD     R1,R5        ;;ADD BACK THE CONSTANT
016226      TST     R2           ;;CHECK IF BCD DIGIT=0
016230      BNE     5$           ;;FALL THROUGH IF 0
016232      TSTB   (SP)         ;;STILL DOING LEADING 0'S?
016234      BMI     7$           ;;BR IF YES
016236      ASLB   (SP)         ;;MSD?
016240      BCC     6$           ;;BR IF NO
016242      MOVVB   1(SP),-1(R3)  ;;YES--SET THE SIGN
016244      BIS     #'0,R2        ;;MAKE THE BCD DIGIT ASCII
016252      BIS     #' ,R2        ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
016256      MOVVB   R2,(R3)+     ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
016262      TST     (R0)+        ;;JUST INCREMENTING
016264      CMP     R0,#10       ;;CHECK THE TABLE INDEX
016266      BLT     2$           ;;GO DO THE NEXT DIGIT
016272      BGT     2$           ;;GO TO EXIT
016274      MOV     R5,R2        ;;GET THE LSD
016276      BR      6$           ;;GO CHANGE TO ASCII
016300      TSTB   (SP)+        ;;WAS THE LSD THE FIRST NON-ZERO?
016302      BPL     9$           ;;BR IF NO
016304      MOVVB   -1(SP),-2(R3) ;;YES--SET THE SIGN FOR TYPING
016306      CLRB   (R3)         ;;SET THE TERMINATOR
016314      MOV     (SP)+,R5     ;;POP STACK INTO R5
016316      MOV     (SP)+,R3     ;;POP STACK INTO R3
016320      MOV     (SP)+,R2     ;;POP STACK INTO R2
016322      MOV     (SP)+,R1     ;;POP STACK INTO R1
016324      MOV     (SP)+,R0     ;;POP STACK INTO R0
016326      TYPE   ,SDBLK       ;;NOW TYPE THE NUMBER
016330

```

```

016334 016666 000002 000004      MOV      2(SP),4(SP)      ;;ADJUST THE STACK
016342 012616                    MOV      (SP)+,(SP)
016344 000002                    RTI          ;;RETURN TO USER
016346 023420                    $DTBL: 10000.
016350 001750                    1000.
016352 000144                    100.
016354 000012                    10.
016356                    $DBLK: .BLKW 4

```

2819

```
.SBTTL TTY INPUT ROUTINE
:*****
:ENABL LSB
:DSABL LSB
:*****
:*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
:*CALL:
:*      RDCHR          ;;INPUT A SINGLE CHARACTER FROM THE TTY
:*      RETURN HERE   ;;CHARACTER IS ON THE STACK
:*                      ;;WITH PARITY BIT STRIPPED OFF
:
016366 011646          $RDCHR: MOV      (SP),-(SP)      ;;PUSH DOWN THE PC
016370 016666 000004 000002 MOV      4(SP),2(SP)    ;;SAVE THE PS
016376 105777 162726 1$:   TSTB    @STKS        ;;WAIT FOR
016402 100375          BPL      1$           ;;A CHARACTER
016404 117766 162722 000004 MOVB    @STKB,4(SP)    ;;READ THE TTY
016412 042766 177600 000004 BIC     #^C<177>,4(SP) ;;GET RID OF JUNK IF ANY
016420 026627 000004 000023 CMP     -(SP),#23     ;;IS IT A CONTROL-S?
016426 001013          BNE     3$           ;;BRANCH IF NO
016430 105777 162674 2$:   TSTB    @STKS        ;;WAIT FOR A CHARACTER
016434 100375          BPL      2$           ;;LOOP UNTIL ITS THERE
016436 117746 162670  MOVB    @STKB,-(SP)   ;;GET CHARACTER
016442 042716 177600  BIC     #^C177,(SP)  ;;MAKE IT 7-BIT ASCII
016446 022627 000021  CMP     (SP)+,#21    ;;IS IT A CONTROL-Q?
016452 001366          BNE     2$           ;;IF NOT DISCARD IT
016454 000750          BR      1$           ;;YES, RESUME
016456 026627 000004 000140 3$:   CMP     4(SP),#140   ;;IS IT UPPER CASE?
016464 002407          BLT      4$           ;;BRANCH IF YES
016466 026627 000004 000175  CMP     4(SP),#175   ;;IS IT A SPECIAL CHAR?
016474 003003          BGT      4$           ;;BRANCH IF YES
016476 042766 000040 000004  BIC     #40,4(SP)   ;;MAKE IT UPPER CASE
016504 000002 4$:   RTI          ;;GO BACK TO USER
:*****
:*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
:*CALL:
:*      RDLIN         ;;INPUT A STRING FROM THE TTY
:*      RETURN HERE  ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
:*                      ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
:
016506 010346          $RDLIN: MOV      R3,-(SP)      ;;SAVE R3
016510 012703 016614 1$:   MOV     #STYIN,R3    ;;GET ADDRESS
016514 022703 016623 2$:   CMP     #STYIN+7.,R3 ;;BUFFER FULL?
016520 101405          BLOS    4$           ;;BR IF YES
016522 104406          RDCHR   ;;GO READ ONE CHARACTER FROM THE TTY
016524 112613          MOVB    (SP)+,(R3)   ;;GET CHARACTER
016526 122713 000177 10$:  CMPB   #177,(R3)    ;;IS IT A RUBOUT
016532 001003          BNE     3$           ;;SKIP IF NOT
016534 104401 001350 4$:   TYPE   ,SQUES     ;;TYPE A '?'
016540 000763          BR      1$           ;;CLEAR THE BUFFER AND LOOP
016542 111337 016612 3$:   MOVB    (R3),9$     ;;ECHO THE CHARACTER
016546 104401 016612  TYPE   ,9$
016552 122723 000015  CMPB   #15,(R3)+    ;;CHECK FOR RETURN
016556 001356          BNE     2$           ;;LOOP IF NOT RETURN
016560 105063 177777  CLRB   -1(R3)      ;;CLEAR RETURN (THE 15)
016564 104401 001352  TYPE   ,LF         ;;TYPE A LINE FEED
016570 012603          MOV     (SP)+,R3    ;;RESTORE R3
016572 011646          MOV     (SP),-(SP)  ;;ADJUST THE STACK AND PUT ADDRESS OF THE
016574 016666 000004 000002 MOV     4(SP),2(SP)  ;;FIRST ASCII CHARACTER ON IT
```

```
016602 012766 016614 000004      MOV    #STTYIN,4(SP)
016610 000002      RTI          ;;RETURN
016612      000      9S:    .BYTE 0      ;;STORAGE FOR ASCII CHAR. TO TYPE
016613      000      .BYTE 0      ;;TERMINATOR
016614      STTYIN: .BLKB 7.      ;;RESERVE 7. BYTES FOR TTY INPUT
016623      136      125      015  SCNTLU: .ASCIZ /^U/<15><12>      ;;CONTROL 'U'
016626      017      000
016630      136      107      015  SCNTIG: .ASCIZ /^G/<15><12>      ;;CONTROL 'G'
016633      012      000
016635      015      012      123  SMSWR: .ASCIZ <15><12>/SWR = /
016640      127      122      040
016643      075      040      000
016646      040      040      116  SMNEW: .ASCIZ / NEW - /
016651      105      127      040
016654      075      040      000

      .EVEN
```


2821

```
.SBTTL READ A DECIMAL NUMBER FROM THE TTY
:*****
:*THIS ROUTINE WILL READ A DECIMAL (ASCII) NUMBER FROM THE TTY AND
:*CHANGE IT TO BINARY. IF TOO MANY CHARACTERS OR ANY ILLEGAL CHARACTERS
:*ARE READ A '?' FOLLOWED BY A CARRIAGE RETURN-LINE FEED WILL BE TYPED.
:*THE COMPLETE NUMBER MUST BE RETYPED. THE INPUT IS TERMINATED BY THE
:*USER TYPING A CARRIAGE RETURN. THE RANGE OF THE INPUT NUMBER IS
:*POSITIVE 32767 TO NEGATIVE 32768.
:*CALL:
```

```

*      RDDEC          ;;READ A DECIMAL NUMBER
*      RETURN HERE   ;;NUMBER IS ON TOP OF THE STACK
:
$RDDEC: MOV      (SP),-(SP)    ;;PROVIDE SPACE FOR
MOV      4(SP),2(SP)        ;;THE INPUT NUMBER
MOV      R0,-(SP)           ;;PUSH R0 ON STACK
MOV      R1,-(SP)           ;;PUSH R1 ON STACK
MOV      R2,-(SP)           ;;PUSH R2 ON STACK
1$:  RDLIN          ;;READ AN ASCII LINE
MOV      (SP)+,R0           ;;ADDRESS OF 1ST CHAR.
MOV      R0,6$             ;;SAVE INCASE OF BAD INPUT
CLR      -(SP)              ;;CLEAR DATA WORD
CLR      R2                 ;;SIGN SET POSITIVE
CMPB    #'-,(R0)           ;;SEE IF A MINUS SIGN WAS TYPED
BNE     2$                 ;;BR IF NO MINUS SIGN
MOVB    (R0)+,R2           ;;SAVE FOR LATER USE
2$:  MOVB    (R0)+,R1       ;;PICKUP THIS CHARACTER
BEQ     3$                 ;;GET OUT IF ZERO
CMPB    #'0,R1             ;;MAKE SURE THIS CHARACTER
BGT     5$                 ;;IS A DIGIT BETWEEN 0 & 9
CMPB    #'9,R1
BLT     5$
BIT     #'(7777),(SP)      ;;DON'T LET NUMBER GET TO BIG
BNE     5$                 ;;BR IF NUMBER WOULD OVERFLOW
ASL     (SP)                ;;*2
MOV     (SP),-(SP)         ;;SAVE FOR LATER
ASL     (SP)                ;;*4
ASL     (SP)                ;;*8
ADD     (SP)+,(SP)         ;;*10
BVS     5$                 ;;OVERFLOW ISN'T ALLOWED
SUB     #'0,R1             ;;STRIP AWAY THE ASCII JUNK
ADD     R1,(SP)            ;;ADD IN THIS DIGIT
BVS     5$                 ;;OVERFLOW ISN'T ALLOWED
BR      2$                 ;;LOOP
3$:  TST      R2            ;;CHECK IF NUMBER IS NEG
BEQ     4$                 ;;BR IF NO
NEG     (SP)                ;;YES--NEGATE THE NUMBER
4$:  MOV     (SP)+,12(SP)   ;;SAVE THE RESULT
MOV     (SP)+,R2           ;;POP STACK INTO R2
MOV     (SP)+,R1           ;;POP STACK INTO R1
MOV     (SP)+,R0           ;;POP STACK INTO R0
RTI                    ;;RETURN
5$:  TST      (SP)+        ;;CLEAN PARTIAL NUMBER FROM STACK
CLRB   (R0)                ;;SET A TERMINATOR
TYPE   TYPE                ;;TYPE THE INPUT UP TO BAD CHAR.
6$:  .WORD   0              ;;POINTER GOES HERE
TYPE   $QUES               ;;'?' 'CR' & 'LF'
BR     1$                 ;;TRY AGAIN

```

```

016660 011646
016662 016666 000004 000002
016670 010046
016672 010146
016674 010246
016676 104407
016700 012600
016702 010037 017026
016706 005046
016710 005002
016712 122710 000055
016716 001001
016720 112002
016722 112001
016724 001424
016726 122701 000060
016732 003032
016734 122701 000071
016740 002427
016742 032716 170000
016746 001024
016750 006316
016752 011646
016754 006316
016756 006316
016760 062616
016762 102416
016764 162701 000060
016770 060116
016772 102412
016774 000752
016776 005702
017000 001401
017002 005416
017004 012666 000012
017010 012602
017012 012601
017014 012600
017016 000002
017020 005726
017022 105010
017024 104401
017026 000000
017030 104401 001350
017034 000720

```

2823

```

.SBTTL READ AN OCTAL NUMBER FROM THE TTY
:*****
:*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
:*CHANGE IT TO BINARY.
:*THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL
:*OCTAL DIGITS. IF AN ILLEGAL CHARACTER IS READ A '?' WILL BE TYPED
:*FOLLOWED BY A CARRIAGE RETURN-LINE FEED. THE COMPLETE NUMBER MUST
:*THEN BE RETYPED. THE INPUT IS TERMINATED BY TYPING A CARRIAGE RETURN.
:*CALL:
:*
RDOCT                                ;;READ AN OCTAL NUMBER
RETURN HERE                          ;;LOW ORDER BITS ARE ON TOP OF THE STACK
                                        ;;HIGH ORDER BITS ARE IN $HI OCT
$RDOCT: MOV      (SP),-(SP)           ;;PROVIDE SPACE FOR THE
MOV      4(SP),2(SP)                ;;INPUT NUMBER
MOV      R0,-(SP)                    ;;PUSH R0 ON STACK
MOV      R1,-(SP)                    ;;PUSH R1 ON STACK
MOV      R2,-(SP)                    ;;PUSH R2 ON STACK
1$: RDLIN                               ;;READ AN ASCII LINE
MOV      (SP)+,R0                    ;;GET ADDRESS OF 1ST CHARACTER
MOV      R0,$$                       ;;AND SAVE IT
CLR      R1                          ;;CLEAR DATA WORD
CLR      R2
2$: MOVB      (R0)+,-(SP)              ;;PICKUP THIS CHARACTER
BEQ      3$                          ;;IF ZERO GET OUT
CMPB     #'0,(SP)                    ;;MAKE SURE THIS CHARACTER
BGT      4$                          ;;IS AN OCTAL DIGIT
CMPB     #'7,(SP)
BLT      4$
ASL      R1                          ;;*2
ROL      R2
ASL      R1                          ;;*4
ROL      R2
ASL      R1                          ;;*8
ROL      R2
BIC      #'C7,(SP)                   ;;STRIP THE ASCII JUNK
ADD      (SP)+,R1                    ;;ADD IN THIS DIGIT
BR       2$                          ;;LOOP
3$: TST      (SP)+                    ;;CLEAN TERMINATOR FROM STACK
MOV      R1,12(SP)                   ;;SAVE THE RESULT
MOV      R2,$HI OCT
MOV      (SP)+,R2                    ;;POP STACK INTO R2
MOV      (SP)+,R1                    ;;POP STACK INTO R1
MOV      (SP)+,R0                    ;;POP STACK INTO R0
RTI                                     ;;RETURN
4$: TST      (SP)+                    ;;CLEAN PARTIAL FROM STACK
CLRB     (R0)                        ;;SET A TERMINATOR
TYPE                                          ;;TYPE UP THRU THE BAD CHAR.
5$: .WORD    0
TYPE     $QUES                        ;; '?' 'CR' & 'LF'
BR       1$                          ;;TRY AGAIN
$HI OCT: .WORD 0                      ;;HIGH ORDER BITS GO HERE

```

```

017036 011646
017040 016666 000004 000002
017046 010046
017050 010146
017052 010246
017054 104407
017056 012600
017060 010037 017164
017064 005001
017066 005002
017070 112046
017072 001420
017074 122716 000060
017100 003026
017102 122716 000067
017106 002423
017110 006301
017112 006102
017114 006301
017116 006102
017120 006301
017122 006102
017124 042716 177770
017130 062601
017132 000756
017134 005726
017136 010166 000012
017142 010237 017174
017146 012602
017150 012601
017152 012600
017154 000002
017156 005726
017160 105010
017162 104401
017164 000000
017166 104401 001350
017172 000730
017174 000000

```

2825

.SBTTL TRAP DECODER

*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
*GO TO THAT ROUTINE.

017176 010046
017200 016600 000002
017204 005740
017206 111000
017210 006300
017212 016000 017232
017216 000200

\$TRAP: MOV R0,-(SP) ;;SAVE R0
MOV 2(SP),R0 ;;GET TRAP ADDRESS
TST -(R0) ;;BACKUP BY 2
MOVB (R0),R0 ;;GET RIGHT BYTE OF TRAP
ASL R0 ;;POSITION FOR INDEXING
MOV \$TRPAD(R0),R0 ;;INDEX TO TABLE
RTS R0 ;;GO TO ROUTINE

;;THIS IS USE TO HANDLE THE "GETPRI" MACRO

017220 011646
017222 016666 000004 000002
017230 000002

\$TRAP2: MOV (SP),-(SP) ;;MOVE THE PC DOWN
MOV 4(SP),2(SP) ;;MOVE THE PSW DOWN
RTI ;;RESTORE THE PSW

.SBTTL TRAP TABLE

*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
*BY THE "TRAP" INSTRUCTION.

ROUTINE

017232 017220
017234 015474
017236 015740
017240 015714
017242 015754
017244 016142
017246 016366
017250 016506
017252 017036
017254 016660

\$TRPAD: .WORD \$TRAP2
\$TYPE ;;CALL=TYPE TRAP+1(104401) TTY TYPEOUT ROUTINE
\$TYPOC ;;CALL=TYPOC TRAP+2(104402) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
\$TYPOS ;;CALL=TYPOS TRAP+3(104403) TYPE OCTAL NUMBER (NO LEADING ZEROS)
\$TYPON ;;CALL=TYPON TRAP+4(104404) TYPE OCTAL NUMBER (AS PER LAST CALL)
\$TYPDS ;;CALL=TYPDS TRAP+5(104405) TYPE DECIMAL NUMBER (WITH SIGN)
\$RDCHR ;;CALL=RDCHR TRAP+6(104406) TTY TYPEIN CHARACTER ROUTINE
\$RDLIN ;;CALL=RDLIN TRAP+7(104407) TTY TYPEIN STRING ROUTINE
\$RDOCT ;;CALL=RDOCT TRAP+10(104410) READ AN OCTAL NUMBER FROM TTY
\$RDDEC ;;CALL=RDDEC TRAP+11(104411) READ A DECIMAL NUMBER FROM TTY

2826

```

2828 .SBTTL MULTIPLE BOARD DIALOGUE
2829
2830
2831 ; MBD MONITOR-COMMAND DECODER
2832
2833 MBD:
017256 104401 017264 TYPE ,65$ ;;TYPE ASCIZ STRING
017256 000424 BR ,64$ ;;GET OVER THE ASCIZ
;;65$: .ASCIZ <CRLF><CRLF>/DO YOU WISH MULTIPLE BOARD DIALOGUE?/
64$:
2834 017334 104401 017342 TYPE ,67$ ;;TYPE ASCIZ STRING
017334 000412 BR ,66$ ;;GET OVER THE ASCIZ
;;67$: .ASCIZ <CRLF>/((Y=YES, N=NO): /
66$:
2835 017366 104406 RDCHR
2836 017370 012637 001360 MOV (SP)+,ANSWER
2837 017374 023727 001360 000131 CMP ANSWER,#131 ;IS ANSWER YES?
2838 017402 001423 BEQ YESMBD ;IF YES,CONTINUE
2839 017404 023727 001360 000116 CMP ANSWER,#116 ;IS ANSWER NO?
2840 017412 001006 BNE ERRMSG ;IF NEITHER YES OR NO ERROR IN REPLY
2841 017414 104401 017422 TYPE ,69$ ;;TYPE ASCIZ STRING
017420 000402 BR ,68$ ;;GET OVER THE ASCIZ
;;69$: .ASCIZ /N/<CRLF>
68$:
2842 017426
2843 017426 000207 ERRMSG: RTS PC ; NO,SKIP DIALOGUE
2844 017430 104401 017436 TYPE ,65$ ;;TYPE ASCIZ STRING
017430 000405 BR ,64$ ;;GET OVER THE ASCIZ
;;65$: .ASCIZ / ??? /
64$:
2845 017450 000702 BR MBD
2846
2847 017452 104401 017460 YESMBD: TYPE ,65$ ;;TYPE ASCIZ STRING
017452 000402 BR ,64$ ;;GET OVER THE ASCIZ
;;65$: .ASCIZ /Y/<CRLF><CRLF>
64$:
2848 017464 104401 017472 PROMPT: TYPE ,65$ ;;TYPE ASCIZ STRING
017464 000403 BR ,64$ ;;GET OVER THE ASCIZ
;;65$: .ASCIZ <CRLF>/= /
64$:
2849 017500 RDCHR ;
2850 017502 012637 001360 MOV (SP)+,ANSWER
2851 017506 023727 001360 000114 CMP ANSWER,#114 ;LIST PRESENT TABLE?
2852 017514 001420 BEQ LSTTBL ;YES
2853 017516 023727 001360 000122 CMP ANSWER,#122 ;RUN PROGRAM?
2854 017524 001423 BEQ RUNPRG ;YES
2855 017526 023727 001360 000105 CMP ANSWER,#105 ;EDIT TABLE?
2856 017534 001425 BEQ EDIT
2857
2858 017536 104401 017544 TYPE ,67$ ;;TYPE ASCIZ STRING
017542 000404 BR ,66$ ;;GET OVER THE ASCIZ
;;67$: .ASCIZ / ??? /
66$:
2859 017554 000743 BR PROMPT

```

```

2860
2861 017556          LSTBL:
      017556 104401 017564      TYPE      .65$      ;;TYPE ASCIZ STRING
      017562 000401          BR          64$      ;;GET OVER THE ASCIZ
      ;;65$: .ASCIZ  /L/
      64$:
2862 017566          JSR      PC,PRTBL
      017566 004737 017626      BR          PROMPT
2863 017572 000734
2864 017574          RUNPRG:
      017574 104401 017602      TYPE      .65$      ;;TYPE ASCIZ STRING
      017600 000402          BR          64$      ;;GET OVER THE ASCIZ
      ;;65$: .ASCIZ  /R/<CRLF>
      64$:
2865 017606          RTS      PC
2866 017610          EDIT:
      017610 104401 017616      TYPE      .65$      ;;TYPE ASCIZ STRING
      017614 000401          BR          64$      ;;GET OVER THE ASCIZ
      ;;65$: .ASCIZ  /E/
      64$:
2867 017620          JSR      PC,TBLEDT
2868 017624 000717      BR          PROMPT
2869
2870 017626 005037 001322      PRTBL: CLR      LOOP
2871 017632 012701 001024      MOV      #REGADR,R1
2872 017636 012702 001076      MOV      #VECADR,R2
2873 017642 104401 017650      TYPE      .65$      ;;TYPE ASCIZ STRING
      017646 000424          BR          64$      ;;GET OVER THE ASCIZ
      ;;65$: .ASCIZ  <CRLF><CRLF><CRLF>/NO. BOARDS REQUESTED FOR TESTING: /
      64$:
2874 017720          MOV      QTYBRD,-(SP)  ;;SAVE QTYBRD FOR TYPEOUT
      017720 013746 001022      TYPDS          ;;GO TYPE--DECIMAL ASCII WITH SIGN
      017724 104405
2875 017726 104401 017734      TYPE      .67$      ;;TYPE ASCIZ STRING
      017732 000416          BR          66$      ;;GET OVER THE ASCIZ
      ;;67$: .ASCIZ  <CRLF><CRLF>/BOARD# REGSTR. VECTOR/<CRLF>
      66$:
2876 017770          INC      LOOP
2877 017774          1$:
      017774 104401 020002      TYPE      .69$      ;;TYPE ASCIZ STRING
      020000 000401          BR          68$      ;;GET OVER THE ASCIZ
      ;;69$: .ASCIZ  <CRLF>
      68$:
2878 020004          MOV      LOOP,-(SP)  ;;SAVE LOOP FOR TYPEOUT
      020004 013746 001322      TYPDS          ;;GO TYPE--DECIMAL ASCII WITH SIGN
      020010 104405
2879 020012 104401 020020      TYPE      .71$      ;;TYPE ASCIZ STRING
      020016 000402          BR          70$      ;;GET OVER THE ASCIZ
      ;;71$: .ASCIZ  / /
      70$:
2880 020024          MOV      (R1)+,-(SP)  ;;SAVE (R1)+ FOR TYPEOUT
      020024 012146          TYPDC          ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
      020026 104402
2881 020030 104401 020036      TYPE      .73$      ;;TYPE ASCIZ STRING
      020034 000402          BR          72$      ;;GET OVER THE ASCIZ
      ;;73$: .ASCIZ  / /
      72$:
2882 020042          MOV      (R2)+,-(SP)  ;;SAVE (R2)+ FOR TYPEOUT
      020042 012246          TYPDC          ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
      020044 104402
2883 020046 023737 001322 001022      CMP      LOOP,QTYBRD
2884 020054 001403          BEQ      PRTFIN

```

2885	020056	005237	001322		INC	LOOP	
2886	020062	000744			BR	1\$	
2887	020064			PRTFIN:			
	020064	104401	020072		TYPE	.65\$::TYPE ASCIZ STRING
	020070	000402			BR	64\$::GET OVER THE ASCIZ
				::65\$:	.ASCIZ	<CRLF><CRLF>	
	020076			64\$:			
2888	020076	000207			RTS	PC	
2889							
2890	020100	012701	001024	TBLEDT:	MOV	#REGADR,R1	
2891	020104	012702	001076		MOV	#VECADR,R2	
2892	020110	005037	001322		CLR	LOOP	
2893	020114			BRDNO:			
	020114	104401	020122		TYPE	.65\$::TYPE ASCIZ STRING
	020120	000425			BR	64\$::GET OVER THE ASCIZ
				::65\$:	.ASCIZ	<CRLF><CRLF>/HOW MANY BOARDS DO YOU WANT TO TEST? /	
	020174			64\$:			
2894	020174	104411			RDDEC		
2895	020176	012637	001360		MOV	(SP)+,ANSWER	
2896	020202	005737	001360		TST	ANSWER	
2897	020206	001742			BEQ	BRDNO	
2898	020210	023727	001360	JUU020	CMP	ANSWER,#20	
2899	020216	003423			BLE	CORRCT	
2900	020220	104401	020226		TYPE	.67\$::TYPE ASCIZ STRING
	020224	000416			BR	66\$::GET OVER THE ASCIZ
				::67\$:	.ASCIZ	<CRLF>/MAX BOARDS OF 16 EXCEEDED/	
	020262			66\$:			
2901	020262	000137	020536		JMP	EDTFIN	
2902	020266	013737	001360	001022	CORRCT:	MOV	ANSWER,QTYBRD
2903	020274	005237	001322		INC	LOOP	
2904	020300			ENTRY:			
	020300	104401	020306		TYPE	.65\$::TYPE ASCIZ STRING
	020304	000405			BR	64\$::GET OVER THE ASCIZ
				::65\$:	.ASCIZ	<CRLF><CRLF>/BOARD /	
	020320			64\$:			
2905	020320	013746	001322		MOV	LOOP,-(SP)	::SAVE LOOP FOR TYPEOUT
	020324	104405			TYPDS		::GO TYPE--DECIMAL ASCII WITH SIGN
2906	020326	104401	020334		TYPE	.67\$::TYPE ASCIZ STRING
	020332	000413			BR	66\$::GET OVER THE ASCIZ
				::67\$:	.ASCIZ	<CRLF>/ REGISTER ADDRESS: /	
	020362			66\$:			
2907	020362	011146			MOV	(R1),-(SP)	::SAVE (R1) FOR TYPEOUT
	020364	104402			TYPDC		::GO TYPE--OCTAL ASCII(ALL DIGITS)
2908	020366	104401	020374		TYPE	.69\$::TYPE ASCIZ STRING
	020372	000402			BR	68\$::GET OVER THE ASCIZ
				::69\$:	.ASCIZ	/ /	
	020400			68\$:			
2909	020400	104410			RDOCT		
2910	020402	012637	001360		MOV	(SP)+,ANSWER	
2911	020406	005737	001360		TST	ANSWER	
2912	020412	001403			BEQ	2\$	
2913							
2914	020414	013721	001360		MOV	ANSWER,(R1)+	::INSTALL NEW # IN TABLE
2915	020420	000401			BR	3\$	
2916	020422	005721		2\$:	TST	(R1)+	
2917	020424			3\$:			
	020424	104401	020432		TYPE	.71\$::TYPE ASCIZ STRING

```
020430 000411
2918 020454 011246
020456 104402
2919 020460 104401 020466
020464 000402
2920 020472 104410
2921 020474 012637 001360
2922 020500 005737 001360
2923 020504 001403
2924
2925 020506 013722 001360
2926 020512 000401
2927 020514 005722
2928
2929 020516 023737 001322 001022
2930 020524 001404
2931 020526 005237 001322
2932 020532 000137 020300
2933 020536 000207

      BR      70$      ::GET OVER THE ASCIZ
      .ASCIZ /VECTOR ADDRESS: /
      MOV     (R2),-(SP)  ::SAVE (R2) FOR TYPEOUT
      TYPOC   ::GO TYPE--OCTAL ASCII(ALL DIGITS)
      TYPE    ,73$      ::TYPE ASCIZ STRING
      BR      72$      ::GET OVER THE ASCIZ
      .ASCIZ / /
      RDOCT
      MOV     (SP)+,ANSWER
      TST    ANSWER
      BEQ    4$
      MOV     ANSWER,(R2)+
      BR      5$
      TST    (R2)+
      4$:
      CMP     LOOP,0TYBRD
      BEQ    EDTFIN
      INC    LOOP
      JMP    ENTRY
      EDTFIN: RTS      PC
```

```

2935      .SBTTL BURST DATA LATE CALIBRATION ROUTINE
2936
2937 020540      BRSTD L:
      020540 104401 020546      TYPE      .65$      ;;TYPE ASCIZ STRING
      020544 000426      BR      64$      ;;GET OVER THE ASCIZ
      ;;65$: .ASCIZ <CRLF><CRLF>/DO YOU WISH BURST DATA LATE CALIBRATION?/
2938 020622      ;;64$:
      020622 104401 020630      TYPE      .67$      ;;TYPE ASCIZ STRING
      020626 000412      BR      66$      ;;GET OVER THE ASCIZ
      ;;67$: .ASCIZ <CRLF>/(Y=YES, N=NO): /
      66$:
2939 020654      RDCHR
2940 020656 012637 001360      MOV      (SP)+,ANSWER
2941 020662 023727 001360 000131      CMP      ANSWER,#131      ;YES?
2942 020670 001421      BEQ      3$
2943 020672 023727 001360 000116      CMP      ANSWER,#116      ;NO?
2944 020700 001006      BNE      4$
2945 020702 104401 020710      TYPE      .69$      ;;TYPE ASCIZ STRING
      020706 000402      BR      68$      ;;GET OVER THE ASCIZ
      ;;69$: .ASCIZ /N/<CRLF>
2946 020714      68$:
      020714 000207      RTS      PC
2947 020716      4$:
      020716 104401 020724      TYPE      .71$      ;;TYPE ASCIZ STRING
      020722 000403      BR      70$      ;;GET OVER THE ASCIZ
      ;;71$: .ASCIZ / ??? /
2948 020732      70$:
      020732 000702      BR      BRSTD L
2949
2950 020734      3$:
      020734 104401 020742      TYPE      .73$      ;;TYPE ASCIZ STRING
      020740 000402      BR      72$      ;;GET OVER THE ASCIZ
      ;;73$: .ASCIZ /Y/<CRLF>
2951 020746      72$:
      020746 104401 020754      TYPE      .75$      ;;TYPE ASCIZ STRING
      020752 000426      BR      74$      ;;GET OVER THE ASCIZ
      ;;75$: .ASCIZ <CRLF>/BURST DATA LATE CALIBRATION IN PROGRESS../
2952 021030      74$:
      021030 104401 021036      TYPE      .77$      ;;TYPE ASCIZ STRING
      021034 000414      BR      76$      ;;GET OVER THE ASCIZ
      ;;77$: .ASCIZ <CRLF>/ATTACH SCOPE PROBE.../
2953 021066      76$:
      021066 104401 021074      TYPE      .79$      ;;TYPE ASCIZ STRING
      021072 000424      BR      78$      ;;GET OVER THE ASCIZ
      ;;79$: .ASCIZ <CRLF>/TO ABORT THIS ROUTINE,HALT PROCESSOR../
2954 021144      78$:
      021144 013700 001004      MOV      DEFRAD,R0      ;BURST DATA LATE CALIBR. USES
2955                                ;DEFAULT REG. AND VECTOR ADDRESSES
2956 021150      CMP      (R0)+,(R0)+
2957 021152 010037 001070      MOV      R0,CSR
2958 021156 013700 001006      MOV      DEFVAD,R0
2959 021162 010037 001136      MOV      R0,DRINV
2960 021166 005720      TST      (R0)+
2961 021170 010037 001140      MOV      R0,DRVS
2962 021174 004737 010156      JSR      PC,NORMAL
2963 021200 005077 157664      CLR      @CSR      ;CLR CYCLE BIT
2964 021204 012737 000100 001312      MOV      #100,TIME
    
```



```

2965 021212 052777 000400 157650      BIS      #400,@CSR      ;SET CYCLE BIT
2966
2967 021220 005337 001312      1$:      DEC      TIME      ;WAIT
2968 021224 001375
2969 021226 000764      BR      2$      ;LOOP ALWAYS:SETTING AND CLEARING CYCLE
2970      ;TO ABORT,HALT PROCESSOR
2971
2972      .SBITL  RUN SUMMARY ROUTINE
2973
2974 021230 013700 001022      RUNSUM:  MOV      QTYBRD,R0      ;NO. OF BOARDS TESTED
2975 021234 012701 001024      MOV      #REGADR,R1
2976 021240 012702 001206      MOV      #PASCNT,R2
2977 021244 012703 001246      MOV      #ERRCNT,R3
2978 021250 104401 021256      TYPE     ,65$      ;;TYPE ASCIZ STRING
      BR      64$      ;;GET OVER THE ASCIZ
      ;;65$: .ASCIZ <CRLF>/RUN SUMMARY.../<CRLF>
2979 021300 104401 021306      64$:      TYPE     ,67$      ;;TYPE ASCIZ STRING
      BR      66$      ;;GET OVER THE ASCIZ
      ;;67$: .ASCIZ <CRLF>/BOARD PASCNT ERRCNT/<CRLF>
2980 021336 102146 021336      66$:      1$:      MOV      (R1)+,-(SP)      ;;SAVE (R1)+ FOR TYPEOUT
      TYPOC      ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
2981 021342 012246      MOV      (R2)+,-(SP)      ;;SAVE (R2)+ FOR TYPEOUT
      TYPDS      ;;GO TYPE--DECIMAL ASCII WITH SIGN
2982 021346 012346      MOV      (R3)+,-(SP)      ;;SAVE (R3)+ FOR TYPEOUT
      TYPDS      ;;GO TYPE--DECIMAL ASCII WITH SIGN
2983 021352 104401 001354      TYPE     , $ENULL
2984 021356 005300      DEC      R0
2985 021360 001405      BEQ      2$
2986 021362 104401 021370      TYPE     ,69$      ;;TYPE ASCIZ STRING
      BR      68$      ;;GET OVER THE ASCIZ
      ;;69$: .ASCIZ <CRLF>
      68$:
2987 021372 000761      BR      1$
2988 021374 000005      2$:      RESET
2989 021376 000137 000200      JMP      200
2990 021402 000000      .SAV:    0
2991      .=. +1000
2992 022404 000000      BUFF:    0      ;FOR STACK PGINTER 100 LOCATIONS
2993 022406 022406      XINBUF:  .
2994      .-. +1000
2995 023410 023410      XCHKBU:  .
2996 023412 000000      LEVEL:  .WORD 0
2997      .END
    
```

ANSWER 001360	CYCLE = 000400	LOADA 007726	RUNSUM 021230	T37 011364
ATTN = 020000	DATCHK 010116	LODBUF 007714	R6 = %000006	T37CHK 011524
BAOFCK 010546	DATOCK 010200	LOOP 001322	R7 = %000007	T40 011544
BAR 001066	DDISP = 177570	LSTTBL 017556	SAVCC 013772	T40CHK 011704
BDFAIL 001362	DEFRAD 001004	MAINT = 010000	SAVPC 013770	T41 005360
BDR 001072	DEFVAD 001006	MANCBL 014755	SAVR2 013762	T42 010276
BEGAUT 001670	DIOMEM 001150	MANT 014723	SAVR3 013764	T55 006440
BEGIN 001604	DISPLA 001324	MBD 017256	SAVR4 013766	T56 011716
BIT0 = 000001	DISPRE 000174	MFLOOP 006662	SCOP 010450	T56CHK 012056
BIT1 = 000002	DONE 005556	MSG4 015014	SCOPE = 000004	T57 012100
BIT10 = 002000	DRINL 001074	MSG5 015057	SCOPEA 013774	T57CHK 012234
BIT11 = 004000	DRINV 001136	MSG6 015131	SCOPEB 014016	T60 006632
BIT12 = 010000	DRVS 001140	MSG7 015163	SCOPEC 014030	T60CHK 006762
BIT13 = 020000	DSTATA= 001000	MSG8 015230	SCOPEF 014124	T61 007024
BIT14 = 040000	DSTATB= 002000	MSG9 015310	SCOPREG 014104	T61CHK 007174
BIT15 = 100000	DSTATC= 004000	NEX = 040000	SR 001000	T61.5 007206
BIT2 = 000004	DSWR = 177570	NEXCHK 010406	STACK = 022404	T62CHK 007676
BIT3 = 000010	EDIT 017610	NOCBL 010734	SUSWR 001364	VECADR 001076
BIT4 = 000020	EDTFIN 020536	NOP = 000240	SWINT1 012740	WCLEN 001164
BIT5 = 000040	EIR 001142	NORMAL 010156	SWINT2 013120	WCR 001064
BIT6 = 000100	END 013132	NPRRDY 011012	SWR 001326	WRONG1 007360
BIT7 = 000200	END1 013216	NPR1 001146	SWREG 000176	WRONG2 007510
BIT8 = 000400	ENTRY 020300	NRM 007514	SWT1 112600	XBA16 = 000020
BIT9 = 001000	ERRCHK 010246	NSTAT 006132	SWT2 112760	XBA17 = 000040
BORW 001144	ERRCNT 001246	NXTBRD 013260	TBLEDT 010100	XCHKBU 023410
BRDNO 020114	ERRDO 006534	OFL 015370	TEMP 011306	XINBUF 022406
BRDSET 001656	ERRDO1 006614	OK 007370	TEMPST 011134	X1 006122
BRLEV 006272	ERRMSG 017430	OUT 014476	TEMP2 000310	X4 005560
BRSTD 020540	ERROR = 100000	OUTSWR 014573	TESTNO 001320	X5 004676
BRWAIT 001162	ERRVEC= 000004	PASCNT 001206	TIB 01442	X7 005740
BUFF 022404	FLAG 001200	PINV 006166	TIME 001312	YESCBL 010722
BUFLN 001156	FLIP 012314	PNTERR 001016	TKB 001112	YESMBD 017452
BUSERR- 000004	FLOP 012446	PNTPAS 001014	TKS 001171	SBELL 001344
CABLE 001314	FNCNT 001202	PNTRAD 001010	TKVEC = 000060	SCHARC 015710
CARLF 014567	FNCT1 = 000002	PNTVAD 001012	TNPRO 011174	SCNTG 014552
CBL 010622	FNCT2 = 000004	PRGCRY 001020	TNPR1 010770	SCNTLG 016630
CHKA 007764	FNCT3 = 000010	PRINT 013314	TPB 001176	SCNTLU 016623
CHKBFF 007752	GO = 000001	PROMPT 017464	TPS 001174	SCNTS 014560
CHKBUF 001154	HLT = 104000	PRTBL 017626	TRAPVE= 000034	SCRLF 001351
CKBAR 002116	HLTDET 001316	PRTFIN 020064	TSTRDY 005534	SDBLK 016356
CKBDR 002142	HT = 000011	PSW 001002	TTIN 014500	SDTBL 016346
CKCSR 002130	ICOUNT 014122	P3INT 005720	TTIN1 014520	SENDAD 013250
CKDRIN 002154	IE = 000100	P3INV 005624	TTIN2 014534	SENPAS 014622
CKDRVS 002170	INBUF 001152	P7ERR 006100	TTOUT 015372	SENULL 001354
CKFIN 002336	INBUF1 001204	P7INV 006010	TYPDS = 104405	SFILLC 001342
CKSWR 014144	INCBA 004652	QTYBRD 001022	TYPE = 104401	SFILLS 001341
CKWCR 002104	INCDB 005406	RDCHR = 104406	TYPOC = 104402	SHD = 000003
CNTLU 014250	INCWC 004622	RDDC = 104411	TYPON = 104404	SHIO^T 017174
COMPAR 010132	INNEW 014603	RDLIN = 104407	TYPOS = 104403	SLF 001352
COMPRR 010214	INTA 010016	RDOCT = 104410	T1STR 002070	SMNEW 016646
CONTIN 013760	INTB 011114	RDSW 014140	T26 004250	SMSWR 016635
CORRCT 020266	INTC 011304	RDYCHK 001166	T27 004372	SNULL 001340
COUNT 014136	LDEXIT 007750	READY = 000200	T3 002414	SOCNT 016136
CPUERR 002204	LENCHK 001160	REGADR 001024	T33 010744	SOMODE 016140
CR = 000015	LEVEL 023412	RETURN 014126	T33CLR 011164	SQUES 001350
CRLF = 000200	LF = 000012	RSET 007504	T33STR 010764	SQUEST 014614
CSR 001070	LOAD 001712	RUNPRG 017574	T34CLR 011354	SRDCHR 016366

SYMBOL TABLE

\$RDDEC 016660	\$SWR = 160000	\$TPS 001334	\$TYPE 015474	\$OFILL 016137
\$RDLIN 016506	\$TITLE 014646	\$TRAP 017176	\$TYPEC 015644	.EMPTY 015416
\$RDOCT 017036	\$TKB 001332	\$TRAP2 017220	\$TYPEX 015712	.REST 015462
\$RDSZ = 000007	\$TKS 001330	\$TRP = 000012	\$TYPOC 015740	.REI 015446
\$READ 014306	\$TN = 000001	\$TRPAD 017232	\$TYPON 015754	.RETR 015470
\$SETUP= 000004	\$TPB 001336	\$TTYIN 016614	\$TYPOS 015714	.SAV 021402
\$STUP = 177777	\$TPFLG 001343			

. ABS. 023414 000
 000000 001

ERRORS DETECTED: 0

VIRTUAL MEMORY USED: 20701 WORDS (81 PAGES)

DYNAMIC MEMORY: 20746 WORDS (79 PAGES)

ELAPSED TIME: 00:02:15

DR11W,DR11W SYSMAC.MLB/ML,DR11W.P11