

# DV11

BSC TST & ROM INST EX  
CZDVAC0

AH-8729C-MC  
COPYRIGHT 75-80  
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 15 columns. Each frame contains a small table or data set. The data is very faint and difficult to read, but it appears to be organized in a structured format, possibly representing test results or program instructions. The frames are separated by thin lines, and the overall layout is consistent with standard microfiche formatting.



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  
38  
39  
40  
41  
42  
43  
44  
45  
46  
47  
48  
49  
50

.REM \$

IDENTIFICATION  
-----

PRODUCT CODE: AC-8728C-MC  
PRODUCT NAME: CZDVACO BSC TST & ROM INST EX  
PRODUCT DATE: OCTOBER, 1979  
MAINTAINER: DIAGNOSTIC ENGINEERING

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) 1975,1979 BY DIGITAL EQUIPMENT CORPORATION

THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION.

DIGITAL	PDP	UNIBUS	MASSBUS
DEC	DECUS	DECTAPE	

51  
52  
53  
54  
55  
56  
57  
58  
59  
60  
61  
62  
63  
64  
65  
66  
67  
68  
69  
70  
71  
72  
73  
74  
75  
76  
77  
78  
79  
80  
81  
82  
83  
84  
85  
86  
87  
88  
89  
90  
91  
92  
93  
94  
95  
96  
97  
98  
99  
100  
101

1. ABSTRACT

THE FUNCTION OF THE DV11 DIAGNOSTICS ARE TO VERIFY THAT THE OPTION OPERATES ACCORDING TO SPECIFICATIONS. THE DIAGNOSTICS VERIFY THAT THERE ARE NO MALFUNCTIONS AND THE ALL OPERATIONS OF THE DV11 ARE CORRECT IN ITS ENVIRONMENT.

PARAMETERS MAY BE SET TO ALERT DIAGNOSTICS AS TO THE DV11 CONFIGURATION BY USING THE 'TRIAL' PROGRAM (CZDVE SA:210). ALL QUESTIONS SHOULD BE ANSWERED AND THEN EACH DIAGNOSTIC WILL 'OVERLAY' THESE PARAMETERS WHICH ARE STORED IN THE 'STATUS TABLE' (SEE SECTION 8.4A). THE ALTERNATIVE TO 'TRIAL' PROGRAM IS 'AUTO SIZING' (SEE SECTION 8.5).

CZDVA DOES R/W TESTS ON BOTH PRIMARY REGISTERS AND ALL SECONDARY REGISTERS. TESTS ARE MADE TO VERIFY THAT NO INTER-ACTION BETWEEN SECONDARY REGISTERS OF ANY LINES EXISTS. CZDVA ALSO EXERCISES ALL MICRO-CODE INSTRUCTIONS AND VERIFIES INTERNAL REGISTERS USED BY THE MICRO PROCESSOR. INTERRUPTS AND NPR'S ARE ALSO TESTED IN THIS DIAGNOSTIC. NOTE: FOR EASE OF DIAGNOSIS; ALL (4) LINE CARDS MAY BE PULLED OUT OF THE SYSTEM UNIT AS MAY THE TWO MODEM CONTROL MODULES. ALSO THE DIAGNOSTIC IN NO WAY READS OR USES THE ROMS THAT ARE IN THE DV11.

CURRENTLY THERE ARE SIX OFF LINE DIAGNOSTICS THAT ARE TO BE RUN IN SEQUENCE TO INSURE THAT IF AN ERROR SHOULD OCCUR IT WILL BE DETECTED AT AN EARLY STAGE AND INSURING THAT DIAGNOSIS OF ERROR WILL BE IMMEDIATE TO PROBLEM.

NOTE: ADDITIONAL DIAGNOSTICS MAY BE ADDED IN THE FUTURE.

THE SIX DIAGNOSTICS ARE:

1. CZDVA [REV] BASIC R/W TEST AND ROM INSTRUCTION EXERCISER.
2. CZDVB [REV] STATIC LINE CARD TESTS.
3. CZDVC [REV] 'FREE RUNNING' ROM TESTS PART 1.
4. CZDVD [REV] 'FREE RUNNING' ROM TESTS PART 2.
5. CZDVE [REV] MODEM CONTROL AND CABLE TESTS PLUS MANUAL PARAMETER INPUT.  
[TRIAL PROGRAM]
6. CZDVF [REV] ASYNCHRONOUS LINE CARD TESTS.

2. REQUIREMENTS

2.1 EQUIPMENT

ANY PDP11 FAMILY CPU (WITH MINIMUM 8k MEMORY)  
ASR 33 (OR EQUIVALENT)  
DV11-AA MUX CNTRL UNIT  
AT LEAST ONE OF THE FOLLOWING  
DV11-BA 8 LINE SYNC MODULES  
DV11-BB 8 LINE ASYNC MODULES  
DV11-BC 4 SYNC LINES, 4 ASYNC LINE

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

2. STORAGE

PROGRAM WILL USE ALL 8K OF MEMORY EXCEPT WHERE ABL AND BOOTSTRAP LOADER RESIDE. LOCATION 1500 THRU 1736 ARE ESPECIALLY TO BE NOTED AND TO BE UNTOUCHED BY OPERATOR AFTER DV11 TRIAL PROGRAM HAS BEEN EXECUTED; OR AFTER THE 'AUTO SIZING' HAS BEEN DONE.

3. LOADING PROCEEDURE

3.1 METHOD

ALL PROGRAMS ARE IN ABSOLUTE FORMAT AND ARE LOADED USING THE ABSOLUTE LOADER. NOTE: IF THE DIAGNOSTICS ARE ON A MEDIA SUCH AS DISK ,MAGTAPE,DECIAPE, OR CASSETTE; FOLLOW INSTRUCTIONS FOR THE MONITOR WHICH HAS BEEN PROVIDED ON THAT SPECIFIC MEDIA.

ABSOLUTE LOADER STARTING ADDRESS \*500

MEMORY \* SIZE

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

3.1.1 PLACE ADDRESS OF ABS LOADER INTO SWITCH REGISTER.  
(ALSO PLACE 'HALT' SW UP)

3.1.2 DEPRESS 'LOAD ADDRESS' KEY ON CONSOLE AND RELEASE.

3.1.3 DEPRESS 'START KEY' ON CONSOLE AND RELEASE (PROGRAM SHOULD NOW BE LOADING INTO CPU)

142  
143  
144  
145  
146  
147  
148  
149  
150  
151  
152  
153  
154  
155  
156  
157  
158  
159  
160  
161  
162  
163  
164  
165  
166  
167  
168  
169  
170  
171  
172  
173  
174  
175  
176  
177  
178  
179  
180  
181  
182  
183  
184  
185  
186  
187  
188  
189  
190  
191  
192  
193  
194

4. STARTING PROCEDURE

- A. SET SWITCH REGISTER TO 000200
- B. DEPRESS 'LOAD ADDRESS' KEY AND RELEASE
- C. SET SWR TO ZERO FOR 'AUTO SIZING' OR LEAVE  
LEAVE SWR BIT 7=1 TO USE EXISTING PARAMETERS SET UP BY DV11 TRIAL  
PROGRAM OR A PREVIOUSLY RUN DV11 DIAGNOSTIC THAT USED THE 'AUTO  
SIZING'. (SECTION 7.2 AND 8.4, 8.5 MAY BE HELPFUL)
- D. DEPRESS 'START KEY' AND RELEASE ; THE PROGRAM WILL TYPE THE PROGRAM NAME.  
(IF THIS WAS THE FIRST START UP OF THE PROGRAM) AND ALSO THE FOLLOWING.

'MAP OF DV11 STATUS'  
1500 175000  
1502 000300  
1504 000226  
1506 000062  
1510 000226  
1512 000062  
1514 000226  
1516 000062  
1520 000226  
1522 000062

THE ABOVE IS ONLY AN EXAMPLE. THIS WOULD INDICATE THE STATUS TABLE  
STARTING AT ADD. 1500 IN THE PROGRAM. THE STATUS TABLE MUST BE  
VERIFIED BY THE USER IF AUTO SIZING IS DONE. FOR INFORMATION OF STATUS  
TABLE SEE SECTION 8.4 FOR HELP.

THE PROGRAM WILL TYPE 'R' AND PROCEED TO RUN THE DIAGNOSTIC

4. CONTROL SWITCH SETTINGS

NOTE: IF THERE IS NO READ SWR(177570); SWR MAY BE MODIFIED AT LOC 176  
OR BY HITTING CONTROL 'G' <^G> ON CONSOLE TERMINAL.

- SW 15 SET: HALT ON ERROR
- SW 14 SET: LOOP ON CURRENT TEST
- SW 13 SET: INHIBIT ERROR PRINT OUT
- SW 12 SET: INHIBIT \*\*ALL\*\* TYPE OUT/BELL ON ERROR.
- SW 11 SET: INHIBIT ITERATIONS. (QUICK PASS)
- SW 10 SET: ESCAPE TO NEXT TEST
- SW 09 SET: LOOP WITH CURRENT DATA
- SW 08 SET: CATCH ERROR AND LOOP ON IT
- SW 07 SET: USE PREVIOUS STATUS TABLE. CLR-DO AUTO SIZE.
- SW 06 SET: RESERVED
- SW 05 SET: RESERVED
- SW 04 SET: RESERVED
- SW 03 SET: RESERVED
- SW 02 SET: LOCK ON SELECTED TEST
- SW 01 SET: RESTART PROGRAM AT SELECTED TEST
- SW 00 SET: RESELECT DV11'S DESIRED ACTIVE.

195  
196  
197  
198  
199  
200  
201  
202  
203  
204  
205  
206  
207  
208  
209  
210  
211  
212  
213  
214  
215  
216  
217  
218  
219  
220  
221  
222  
223  
224  
225  
226  
227  
228  
229

4.1.2 SWITCH REGISTER RESTRICTIONS

SW 00 RESELECT DV11'S DESIRED ACTIVE. PLEASE NOTE THAT A MESSAGE IS TYPED OUT FOR SETTING THE SWITCH REGISTER EQUAL TO DV11'S ACTIVE. THIS MEANS IF THE SYSTEM HAS FOUR DV11S; BITS 00,01,02,03 WILL BE SET IN LOC 'DVACTV' FROM THE SWITCH REGISTER. USING THIS SWITCH(SW00) ALTERS THAT LOCATION; THEREFORE IF FOUR DV11S ARE IN THE SYSTEM \*\*\*DO NOT\*\*\* SET SWITCHS GREATER THAN SW 03 IN THE UP POSITION. THIS WOULD BE A FATAL ERROR. DO NOT SELECT MORE ACTIVE DV11S THAN HAS BEEN GIVEN INFORMATION ABOUT IN TRIAL PROGRAM.

METHOD: A: LOAD ADDRESS 200  
B: START WITH SW 00=1  
C: PROGRAM WILL TYPE MESSAGE  
D: SET THE BINARY NUMBER OF DV11S DESIRED ACTIVE EXAMPLE: 1-1 DV11; 3-2 DV11; 7-3 DV11; 17=4 DV11 37=5 DV11 ETC. PRESS CONTINUE.  
E: NUMBER (IF VALID) WILL BE IN DATA LIGHTS (EXCLUDING 11/05)  
F: SET WITH ANY OTHER SWITCH SETTINGS DESIRED. PRESS CONTINUE.

SW 01 RESTART PROGRAM AT SELECTED TEST: IT IS STRONGLY SUGGESTED THAT AT LEAST ONE PASS HAS BEEN MADE BEFORE TRYING TO TEST THAT IS NOT IN THE ORDER OF SEQUENCE THE REASON BEING IS THAT THE PROGRAM HAS TO CLEAR AREAS AND SET UP PARAMETERS. ALSO WHEN A TEST IS SELECTED ALWAYS START AT THE VERY BEGINNING OF THAT TEST.

SW 09 LOOP ON CURRENT DATA: THIS SWITCH WILL ONLY WORK IF CALL 'SCOPI' IS IN THAT TEST. THE REASON BEING THAT MOST TESTS DEAL WITH BLOCKS OF DIFFERENT DATA TO BE SENT OR RECEIVED ALL AT ONCE THUS IN BLOCK DA'A; ONE PATTERN CANN'T BE SINGLED OUT.

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  
263  
264  
265  
266  
267  
268  
269  
270  
271  
272  
273  
274

4.1.3 SWITCH REGISTER PRIORITYS

ERROR SWITCHES

1. SW 12 DELETE PRINT OUT/BELL ON ERROR.
2. SW 13 DELETE ERROR PRINTOUT.
3. SW 15 HALT ON THE ERROR.
4. SW 08 GOTO BEGINNING OF THE TEST(ON ERROR).
5. SW 10 GOTO NEXT TEST(ON ERROR).

SCOPE SWITCHES

1. SW 09 (IF ENABLED BY 'SCOPI') ON AN ERROR; IF AN '\*' IS PRINTED IN FRONT OF THE TEST NO. (EX. \*TEST NO. 10) SW09 IS INCORPORATED IN THAT TEST AND THEREFORE SW09 IS \*USUALLY\* THE BEST SWITCH FOR THE SCOPE LOOP (SW14=0, SW10=0, SW09=1, SW08=0). IF SW09 IS NOT ENABELED; AND THERE IS A \*HARD\* ERROR (CONSTANT); SW08 IS BEST.  
(SW14=1,0, SW10=0, SW09=0, SW08=1). FOR INTERMITTEMT ERRORS; SW14 1 WILL LOOP ON TEST REGARDLESS OF ERROR OR NOT ERROR.  
(SW14 1, SW10=0, SW09=0, SW08=1,0)
2. SW 14
3. SW 11

4.2 STARTING ADDRESS

STARTING ADDRESS IS AT 000200 THERE ARE NO OTHER STARTING ADDRESSES FOR THE DV11 DIAGNOSTICS PREVIOUSLY MENTIONED EXCEPT FOR CZDVE WHICH IS: 000200 FOR THE MODEM CONTROL AND CABLE TESTS AND 000210 FOR THE MANUAL PARAMETER INPUT PROGRAM.

NOTE: IF ADDRESS 000042 IS NON-ZERO THE PROGRAM ASSUMES IT IS UNDER ACT11 OR XXDP CONTROL AND WILL ACT ACCORDINGLY AFTER \*ALL\* AVAILABLE DV11'S ARE TESTED THE PROGRAM WILL RETURN TO 'XXDP' OR 'ACT-11'.

5. OPERATING PROCEDURE

WHEN PROGRAM IS INITIALLY STARTED MESSAGES AS DESCRIBED IN SECTION FOUR WILL BE PRINTED, AND PROGRAM WILL BEGIN RUNNING THE DIAGNOSTIC.

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

5.2 PROGRAM AND/OR OPERATOR ACTION

THE TYPICAL APPROACH SHOULD BE

1. HALT ON ERROR (VIA SW 15-1) WHEN EVER AN ERROR OCCURS.
2. CLEAR SW 15.
3. SET SW 14: (LOOP ON THIS TEST)
4. SET SW 13: (INHIBIT ERROR PRINT OUT)

THE TEST NUMBER AND PC WILL BE TYPED OUT AND POSSIBLY AN ERROR MESSAGE (THIS DEPENDS ON THE TEST) TO GIVE THE OPERATOR AN IDEA AS TO THE SOURCE OF THE PROBLEM. IF IT IS NECESSARY TO KNOW MORE INFORMATION CONCERNING THE ERROR REPORT; LOOK IN THE LISTING FOR THAT TEST NUMBER WHICH WAS TYPED OUT AND THEN NOTE THE PC OF THE ERROR REPORT THIS WAY THE EXACT FUNCTIONING OF THE TEST CAN BE INTERPRETED.

6. ERRORS

AS DESCRIBED PREVIOUSLY THERE WILL ALWAYS BE A TEST NUMBER AND PC TYPED OUT AT THE TIME OF AN ERROR (PROVIDING SW 13-0 AND SW 12-0). IN MOST CASES ADDITIONAL INFORMATION WILL BE SUPPLIED TO THE THE ERROR MESSAGE WHICH IS TO GIVE THE OPERATOR AN INDICATION OF THE ERROR.

6.1 ERROR RECOVERY

IF FOR SOME REASON THE DV11 SHOULD 'HANG THE BUS' (GAIN CONTROL OF BUS SO THAT CONSOLE MANUAL FUNCTIONS ARE INHIBITED) AN INIT OR POWER DOWN/UP IS NECESSARY FOR OPERATOR TO REGAIN CONTROL OF CPU. IF THIS SHOULD HAPPEN; LOOK IN LOCATION 'TSTNO' (ADDRESS 1224) FOR THE NUMBER OF THE TEST THAT WAS RUNNING AT THE TIME OF THE CATASTROPHIC ERROR. IN THIS WAY THE OPERATOR WILL HAVE AN IDEA AS TO WHAT THE DV11 WAS DOING AT THE TIME OF THE ERROR.

7. RESTRICTIONS

7.1 STARTING RESTRICTIONS

SEE SECTION 4. (PLEASE)  
STATUS TABLE SHOULD BE VERIFIED REGARDLESS OF HOW PROGRAM WAS STARTED. ALSO IT IS IMPORTANT TO USE THIS LISTING ALONG WITH THE INFORMATION PRINTED ON THE TTY TO COMPLETELY ISOLATE PROBLEMS.



319  
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

7.2 OPERATING RESTRICTIONS

DV11 TRIAL PROGRAM MUST BE RUN PRIOR TO THE FIRST AND ONLY THE FIRST RUNNING OF ANY DV11 DIAGNOSTIC IF 'AUTO SIZING' IS NOT USED.  
NOTE: IF NO PROGRAM OTHER THAN A DV11 DIAGNOSTIC WAS LOADED AFTER DV11 TRIAL OR IF CORE MEMORY HAS NOT BEEN CHANGED; OR IF THERE IS NO DV11 CONFIGURATION CHANGES; THE DV11 TRIAL PROGRAM NEED NEVER BE RUN AGAIN. HOWEVER IF ANY OF THE ABOVE HAVE BEEN VIOLATED THE DV11 TRIAL PROGRAM MUST BE RUN AGAIN BEFORE RUNNING THE DIAGNOSTICS NOTE: AN ALTERNATIVE TO THE ABOVE IS ATTEMPTING THE 'AUTO SIZING' WHEN PROGRAM IS INITIALLY STARTED WITH SW07=0.

7.3 HARDWARE CONFIGURATION RESTRICTIONS (SYNC LINE CARDS ONLY)

1. HARDWARE MUST BE SET TO FULL DUPLEX
2. ALL LINES OF A PARTICULAR LINE CARD MUST BE CONFIGURED THE SAME.

8. MISCELLANEOUS

8.1 EXECUTION TIME

ALL DV11 DEVICE DIAGNOSTICS WILL GIVE AN 'END PASS' MESSAGE (PROVIDING NO ERRORS AND SW12=0) WITHIN 4 MINS. THIS IS ASSUMING SW11=1 (DELETE ITERATIONS) IS SET TO GIVE THE FASTEST POSSIBLE EXECUTION. THE ACTUAL EXECUTION TIME DEPENDS GREATLY ON THE PDP11 CPU CONFIGURATION.

8.2 PASS COMPLETE

NOTE: \*EVERY\* TIME THE PROGRAM IS STARTED; THE TESTS WILL RUN AS IF SW11 (DELETE ITERATIONS) WAS UP (=1). THIS IS TO 'VERIFY NO \*HARD\* ERRORS' AS SOON AS POSSIBLE. THEREFORE THE FIRST PASS -EACH TIME PROGRAM IS STARTED- WILL BE A 'QUICK PASS' UNTILL ALL DV11'S IN SYSTEM ARE TESTED. WHEN THE DIAGNOSTIC HAS COMPLETED A PASS THE FOLLOWING IS AN EXAMPLE OF THE PRINT OUT TO BE EXPECTED.

END PASS CZDVA\_ CSR: 175000 VEC: 300 PASSES: 000001

NOTE: THE NUMBERS FOR CSR AND VEC ARE NOT NECESSARILY THE VALUES FOR THE DEVICE. THEY ARE ONLY FOR THIS EXAMPLE.

NOTE: CZDVE (MODEM AND CABLE TEST) END PASS MESSAGE IS A LARGE 'END' TYPED OUT ON TTY. PLEASE NOTE THAT EACH CHARACTER PRINTED IS ACTUALLY AND 'END PASS' INDICATION. THIS WAS USED IN PLACE OF 'BELL' BECAUSE IF SW12=1 AND AN ERROR OCCURED THE BELL MAY BE MISTAKEN FOR END PASS. THE PASS EXECUTION IS SO FAST THAT THE STANDARD END PASS WAS TOO LENGTHLY. THEREFORE EACH CHAR IS AN 'END PASS AND THE ENTIRE 'END' IS NOT REQUIRED FOR ACCEPTANCE.

367  
368  
369  
370  
371  
372  
373  
374  
375  
376  
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

8.4 KEY LOCATIONS

RETURN (1212) CONTAINS THE ADDRESS WHERE PROGRAM WILL RETURN WHEN ITERATION COUNT IS REACHED OR IF LOOP ON TEST IS ASSERTED.  
NEXT (1214) CONTAINS THE ADDRESS OF THE NEXT TEST TO BE PERFORMED.  
TSTNO (1224) CONTAINS THE NUMBER OF THE TEST NOW BEING PERFORMED.  
RUN (1302) THE BIT IN 'RUN' ALWAYS POINTS ONE PAST THE DV11 CURRENTLY BEING TESTED. EXAMPLE: (RUN) 1302/00000000100000 MEANS THAT DV11 NO.05 IS THE DV11 NOW RUNNING.

DVCR00-DVCR17  
DVST00-DVST17  
(1500)-(1736)

THESE LOCATIONS CONTAIN THE INFORMATION NEEDED TO TEST UP TO 8 (DECIMAL) DV11S SEQUENTIALY. THEY CONTAIN THE CSR,VECTOR AND STATUS CONCERNING THE CONFIGURATION OF EACH DV11.

DVACTV (1276) EACH BIT SET IN THIS LOCATION INDICATES THAT THE ASSOCIATED DV11 WILL BE TESTED IN TURN. EXAMPLE: (DVACTV) 1276/000000000011111 MEANS THAT DV11 NO. 00,01,02,03,04 WILL BE TESTED. EXAMPLE: (DVACTV) 1276/0000000000010001 MEANS THAT DV11 NO. 00,04 WILL BE TESTED.

DVSCR (1356) CONTAINS THE RECEIVER CSR OF THE CURRENT DV11 UNDER TEST.

L00.03 (1412)  
L04.07 (1414)  
L08.11 (1416)  
L12.15 (1420)

CONTAINS THE STATUS OF THE CURRENT DV11 UNDER TEST.

BIT 15 SET: LINE CARD \*NOT INSTALLED (AND WONT BE TESTED)  
BIT 14 SET: RESERVED  
BIT 13 SET: RESERVED  
BIT 12 SET: ONE SYNC, =0: TWO SYNCs.  
BIT 11 SET: ASYNC LINE CARD, =0 SYNC LINE CARD.  
BIT 10 SET: RESERVED  
BIT 09 SET: BITS PER CHAR. (USED WITH BIT8)  
BIT 08 SET: BITS PER CHAR. (USED WITH BIT9)  
BIT09 BIT08 BITS PER CHAR.

0 0 8  
0 1 7  
1 0 6  
1 1 5

BIT 07-00 SYNC 'A' FOR SPECIFIED LINE CARD.  
BITS 07-00 MUST BE ALL ZEROS FOR TESTING OF AN ASYNC LINE CARD.

416  
417  
418  
419  
420  
421  
422  
423  
424  
425  
426  
427  
428  
429  
430  
431  
432  
433  
434  
435  
436  
437  
438  
439  
440  
441  
442  
443  
444  
445  
446  
447  
448  
449  
450  
451  
452  
453  
454  
455  
456  
457  
458  
459  
460  
461  
462  
463  
464  
465  
466  
467  
468  
469  
470  
471

8.4A MORE ON THAT 'STATUS TABLE' (1500-1736)

'MAP OF DV11 STATUS'

1500	175000
1502	000300
1504	000226
1506	000062
1510	000226
1512	000062
1514	004000
1516	000000
1520	004000
1522	000000

SYNC 'A' AND SYNC 'B' MUST BE SET TO ZEROS FOR AN ASYNC LINE CARD.  
THE ABOVE INFORMATION WILL BE REPEATED FOR EACH OF UP TO 8 DV11'S IN THE SYSTEM (THESE WILL FOLLOW UNDER THIS TABLE). EXPLANATION:

1500 175000 THIS IS THE SYSTEM CONTROL REGISTER FOR THE 1ST DV11 IN THE SYSTEM.  
 1502 000300 THIS IS VECTOR 'A' FOR THE FIRST DV11 IN THE SYSTEM.  
 1504 000226 THIS REPRESENTS 'SYNC A' AND THE SOFTWARE STATUS FOR THE 1ST LINE CARD IN THE 1ST DV11. THE BITS ARE AS FOLLOWS:

BIT 15 SET: LINE CARD \*NOT INSTALLED (AND WONT BE TESTED)  
 BIT 14 SET: RESERVED  
 BIT 13 SET: RESERVED  
 BIT 12 SET: ONE SYNC, =0: TWO SYNCs.  
 BIT 11 SET: ASYNC LINE CARD, =0 SYNC LINE CARD.  
 BIT 10 SET: RESERVED  
 BIT 09 SET: BITS PER CHAR. (USED WITH BIT8)  
 BIT 08 SET: BITS PER CHAR. (USED WITH BIT9)

BIT09	BIT08	BITS PER CHAR.
0	0	8
0	1	7
1	0	6
1	1	5

BIT 07-00 SYNC 'A' FOR SPECIFIED LINE CARD.  
 1506 000062 THIS REPRESENTS 'SYNC B' FOR THE 1ST LINE CARD.  
 1510 000226 THIS IS 'SYNC A' AND LINE STATUS FOR THE 2ND LINE CARD. (FOR BITS DEFINATION SEE EXPLANATION FOR LINE CARD 1).  
 1512 000062 THIS IS 'SYNC B' FOR THE SECOND LINE CARD.  
 1514 000226 THIS IS 'SYNC A' AND LINE STATUS FOR THE 3RD LINE CARD. (FOR BITS DEFINATION SEE EXPLANATION FOR LINE CARD 1).  
 1516 000062 THIS IS 'SYNC B' FOR LINE CARD NO. 3.  
 1520 000226 THIS IS 'SYNC A' AND LINE STATUS FOR THE 4TH LINE CARD. (FOR BITS DEFINATION SEE EXPLANATION FOR LINE CARD 1).  
 1522 000062 THIS IS SYNC B FOR THE 4TH LINE CARD.

THE ABOVE IS REPEATED FOR EACH DV11 IN THE SYSTEM. THE TABLE IS FILLED BY AUTO SIZING OR BY THE MANUAL PARAMETER INPUT PROGRAM AS DESCRIBED PREVIOUSLY. ALSO IF DESIRED BY USER; THE LOCATIONS MAY BE ALTERED BY HAND (TOGGLED IN) TO SUIT THE SPECIFIC CONFIGURATION.



472  
473  
474  
475  
476  
477  
478  
479  
480  
481  
482  
483  
484  
485  
486  
487  
488  
489  
490  
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

8.5 \*\*\* METHOD OF AUTO SIZING \*\*\*

8.5.1 FINDING THE CONTROL STATUS REGISTER.

THE PROGRAM WILL START AT ADDRESS 175000 AND START 'REFERENCING' ADDRESS. IF A NON-EX MEMORY TRAP OCCURS; THE POINTER (HOLDING 175000) IS UPDATED BY 10 AND THE ABOVE IS REPEATED UNTIL ADDRESS 175400 IS REACHED. IF A 'SLAVE SYNC RESPONSE' WAS ISSUED BY THE DV11 (OR ANY OTHER DEVICE) 'NO NXM TRAP'(AND IT(SELO)WAS=0) ; POINTER PLUS 12 (SEL12) IS TESTED TO CONTAIN 177777 (MUST BE EXACTLY 177777); IF A TRAP IS ENCOUNTERED OR IF SEL12 DOES NOT CONTAIN 177777 THE ABOVE UPDATING IS PERFORMED. IF SEL12 WAS EQUAL TO 177777 THE POINTER IS STORED AWAY AND THE ROUTINE CONTINUES AS ABOVE:  
NOTE: IF THE PROGRAM DOES NOT FIND YOUR DV11; SOMETHING IS WRONG AND AUTO SIZING SHOULD NOT BE DONE.

8.5.2 FINDING THE VECTOR

THE VECTOR AREA (ADDRESS 300-776) IS FILLED WITH THE INSTRUCTION IOT AND '+2' (NEXT ADDRESS). BIT7 AND BIT6 (RX INTERRUPT AND RX INTERRUPT IE) ARE SET INTO DVSCR REGISTER; A DELAY IS MADE AND IF NO INTERRUPT OCCURS (BECAUSE OF A BAD DV11) THE PROGRAM ASSUMES VECTOR ADDRESS 300 AND THE PROBLEM SHOULD BE FIXED IN THE DIAGNOSTIC. ONCE THE PROBLEM IS FIXED; THE PROGRAM SHOULD BE SETUP AGAIN TO GET CORRECT VECTOR. IF AN INTERRUPT OCCURED; THE ADDRESS TO WHICH THE DV11 INTERRUPTED TO IS PICKED UP AND REPORTED AS THE VECTOR. NOTE: IF THE VECTOR REPORTED IS NOT THE VECTOR SET UP BY YOU; THERE IS A PROBLEM AND AUTO SIZING SHOULD NOT BE DONE.

8.5.3 PARAMETER ASSUMPTIONS.

SINCE TOO MUCH HARDWARE WOULD NEED TO BE TURNED ON TO SIZE THE REST OF THE PARAMETERS; THE PROGRAM MUST ASSUME THE REMAINING VARIATIONS. THE RESULT IF NOT TO YOUR SPECIFIC CONFIGURATION MAY BE ALTERED BY HANG (TOGGLE IN) IS DESIRED. IN THIS WAY 95% OF THE PARAMETER SETUP WAS DONE BY THE PROGRAM AND 5% BY YOU.

THEREFORE:

- 1) ALL LINE CARDS(4) ARE ASSUMED TO BE INSTALLED.  
SET BIT15 OF STATUS MAP OF ANY (APPROIATE) LINE CARDS MISSING
- 2) TWO SYNC.  
SET BIT12 IF YOU HAVE A 4 LINE GROUP SET FOR 1 SYNC.
- 3) ADJUST BITS 8 AND 9 IN STATUS MAP FOR YOUR CORRECT CONFIG.
- 4) SYNCHRONOUS LINE CARDS INSTALLED.  
SET BIT11 FOR ASYNC LINE AND ZERO SYNC CHARS.
- 5) SYNC 'A'=226 AND SYNC 'B'=062

IN ALL ADJUSTMENTS PLEASE REFER TO SECTION 8.4A FOR GREATER DETAIL.

523  
524  
525  
526  
527  
528  
529  
530  
531  
532  
533  
534  
535  
536  
537

9.0 CHANGE HISTORY

NOTE: HISTORY STARTS WITH REV. C

CZDVACO - 1) ADDED TEST 130 TO VERIFY EXTENDED ADDRESS  
BITS 16 AND 17 UTILIZING ALU FUNCTION.

2) MODIFIED FOR COMPATIBILITY WITH COMMON DV11 DIAGNOSTIC  
OPERATOR FILE (CZDVXX.OPR), AND SYSMAC FILE (CZDVXX.SML).

\$

```
538
539
540      ;*CZDVAC-0/<377>/CZDVACO BSC TST & ROM INST EX
541      ;*COPYRIGHT 1972,1979, DIGITAL EQUIPMENT CORP., MAYNARD, MASS. 01754
542      ;-----
543
544      ;STARTING PROCEDURE
545      ;LOAD PROGRAM
546      ;LOAD ADDRESS 000200
547      ;PRESS START
548      ;PROGRAM WILL TYPE 'CZDVAC-0/<377>/CZDVACO BSC TST & ROM INST EX'
549      ;PROGRAM WILL TYPE 'R' TO INDICATE THAT TESTING HAS STARTED
550      ;AT THE END OF A PASS, PROGRAM WILL TYPE PASS COMPLETE MESSAGE
551      ;AND THEN RESUME TESTING
552
553
554      ;SWITCH REGISTER OPTIONS
555      ;-----
556
557      100000      SW15=100000      :=1,HALT ON ERROR
558      040000      SW14=40000      :=1,LOOP ON CURRENT TEST
559      020000      SW13=20000      :=1,INHIBIT ERROR TIMEOUT
560      010000      SW12=10000      :=1,DELETE TIMEOUT/BELL ON ERROR.
561      004000      SW11=4000       :=1,INHIBIT ITERATIONS
562      002000      SW10=2000       :=1,ESCAPE TO NEXT TEST ON ERROR
563      001000      SW09=1000       :=1,LOOP WITH CURRENT DATA
564      000400      SW08=400        :=1,LOOP ON ERROR
565      000200      SW07=200        :=1, DO 'AUTO SIZING' ON INITIAL START UP.
566      000100      SW06=100
567      000040      SW05=40
568      000020      SW04=20
569      000010      SW03=10
570      000004      SW02=4          ;LOCK ON TEST SELECT
571      000002      SW01=2          ;RESTART PROGRAM AT SELECTED TEST
572      000001      SW00=1          ;RESELECT DV11 DESIRED ACTIVE
573                                     ;NOTE: THIS MUST NOT EXCEED ORIGINAL COUNT
```



```
574
575
576           ;REGISTER DEFINITIONS
577           ;-----
578
579           000000      R0=%0           ;GENERAL REGISTER
580           000001      R1=%1           ;GENERAL REGISTER
581           000002      R2=%2           ;GENERAL REGISTER
582           000003      R3=%3           ;GENERAL REGISTER
583           000004      R4=%4           ;GENERAL REGISTER
584           000005      R5=%5           ;GENERAL REGISTER
585           000006      SP=%6          ;PROCESSOR STACK POINTER
586           000007      PC=%7          ;PROGRAM COUNTER
587
588           ;LOCATION EQUIVALENCIES
589           ;-----
590
591           177776      PS=177776       ;PROCESSOR STATUS WORD
592           001200      STACK=1200     ;START OF PROCESSOR STACK
593
594           100000      BIT15=100000
595           040000      BIT14=40000
596           020000      BIT13=20000
597           010000      BIT12=10000
598           004000      BIT11=4000
599           002000      BIT10=2000
600           001000      BIT9=1000
601           000400      BIT8=400
602           000200      BIT7=200
603           000100      BIT6=100
604           000040      BIT5=40
605           000020      BIT4=20
606           000010      BIT3=10
607           000004      BIT2=4
608           000002      BIT1=2
609           000001      BIT0=1
610
611           ;-----
612           010000      ALU=BIT12
613           020000      RAM=BIT13
614           030000      XFR=BIT13+BIT12
615           040000      NPR=BIT14
616           050000      S.C=BIT14+BIT12
617           060000      BCC=BIT14+BIT13
618           070000      BRB=BIT14+BIT13+BIT12
619           ;-----
620
```

TRAPCATCHER FOR UNEXPECTED INTERRUPTS

SEQ 0015

```
621 :*****
622 :-----
623 :TRAPCATCHER FOR ILLEGAL INTERRUPTS
624 :THE STANDARD 'TRAP CATCHER' IS PLACED
625 :BETWEEN ADDRESS 0 TO ADDRESS 776.
626 :IT LOOKS LIKE 'PC+2 HALT'.
627 :-----
628 :*****
629 :
630 000000 . 0
631 :STANDARD INTERRUPT VECTORS
632 :-----
633 :
634 000024 . 24
635 000024 004402 .PFA:L ;POWER FAIL HANDLER
636 000026 000340 340 ;SERVICE AT LEVEL 7
637 000030 004002 .HLT ;ERROR HANDLER
638 000032 000340 340 ;SERVICE AT LEVEL 7
639 000034 003750 .TRPSRV ;GENERAL HANDLER DISPATCH SERVICE
640 000036 000340 340 ;SERVICE AT LEVEL 7
641 000040 .-40
642 000040 000001 .BLKW 1 ;SAVE FOR ACT-11 OR DDP2
643 000042 000001 .BLKW 1 ;RETURN ADDRESS IF UNDER ACT-11 OR DDP2
644 000044 000001 .BLKW 1 ;SAVE FOR ACT-11 OR DDP2
645 000046 002560 LOGICAL ;FOR USE WITH ACT-11 OR DDP2
646 :
647 000174 .-174
648 000174 000000 LIGHT: 0
649 000176 000176 .=176
650 000176 000000 SSWR: 0
651 :
652 000200 .=200
653 000200 000137 001742 JMP .START ;GO TO START OF PROGRAM
654 :
655 :
656 001000 .=1000
657 001000 005377 055103 053104 MTITLE: .ASCIZ <377><12>/CZDVAC-0/<377>/CZDVACO BSC TST & ROM INST EX/<377>
658 :
659 001200 .-1200
660 001200 177570 LIGHTS:
661 001202 177570 SWR: 177570
662 :INDIRECT POINTERS TO TELETYPE VECTORS AND REGISTERS
663 :-----
664 :
665 001204 177560 TKCSR: 177560 ;TELETYPE KEYBOARD CONTROL REGISTER
666 001206 177562 TKDBR: 177562 ;TELETYPE KEYBOARD DATA BUFFER
667 001210 177564 TPCSR: 177564 ;TELEPRINTER CONTROL REGISTER
668 001212 177566 TPDBR: 177566 ;TELEPRINTER DATA BUFFER
669 :
670 :PROGRAM CONTROL PARAMETERS
671 :-----
672 :
673 001214 000000 FETURN: 0 ;SCOPE ADDRESS FOR LOOP ON TEST
674 001216 000000 NEXT: 0 ;ADDRESS OF NEXT TEST TO BE EXECUTED
675 001220 000000 LOCK: 0 ;ADDRESS FOR LOCK ON CURRENT DATA
```

676	001222	000003	ICOUNT: 3	:NUMBER OF ITERATIONS THAT CURRENT TEST WILL BE EXECUTED
677	001224	000000	LPCNT: 0	:NUMBER OF ITERATIONS COMPLETED
678	001226	000000	TSTNO: 0	:NUMBER OF TEST IN PROGRESS
679	001230	000000	PASCNT: 0	:NUMBER OF PASSES COMPLETED
680	001232	000000	ERRCNT: 0	:TOTAL NUMBER OF ERRORS
681	001234	000000	LSTERR: 0	:PC OF LAST ERROR CALL

682  
683 :PROGRAM VARIABLES  
684 :-----  
685

686	001236	000000	STAT: 0	:DV STATUS WORD STORAGE
687	001240	000000	SYNXX: 0	
688	001242	000000	CLKX: 0	
689	001244	000000	MASKX: 0	
690	001246	000000	TEMP1: 0	:TEMPORARY STORAGE
691	001250	000000	TEMP2: 0	:TEMPORARY STORAGE
692	001252	000000	TEMP3: 0	:TEMPORARY STORAGE
693	001254	000000	TEMP4: 0	:TEMPORARY STORAGE
694	001256	000000	TEMP5: 0	:TEMPORARY STORAGE
695	001260	000000	SAVR0: 0	:R0 STORAGE
696	001262	000000	SAVR1: 0	:R1 STORAGE
697	001264	000000	SAVR2: 0	:R2 STORAGE
698	001266	000000	SAVR3: 0	:R3 STORAGE
699	001270	000000	SAVR4: 0	:R4 STORAGE
700	001272	000000	SAVR5: 0	:R5 STORAGE
701	001274	000000	SAVSP: 0	:STACK POINTER STORAGE
702	001276	000000	SAVPC: 0	:PROGRAM COUNTER STORAGE
703	001300	000001	DVACTV: .BLKB 1	:DV11'S SELECTED ACTIVE.
704	001301	000001	DVNUM: .BLKB 1	:OCTAL NUMBER OF DV11'S.
705	001302	000001	SAVACT: .BLKB 1	:ORIGINAL ACTV. DEVICES.
706	001303	000001	SAVNUM: .BLKB 1	:WORKABLE NUMBER.
707	001304	000001	RUN: .BLKB 1	:POINTER ONE PAST RUNNING DEVICE.
708		001306	.EVEN	
709	001306	001500	CREAM: DV.MAF	:TABLE POINTER.



```

710
711           ;PROGRAM CONTROL FLAGS
712           ;-----
713
714 001310     000          INIFLG: .BYTE 0          ;PROGRAM INITIALIZATION FLAG
715 001311     000          ERRFLG: .BYTE 0          ;ERROR OCCURED FLAG
716 001312     000          LOKFLG: .BYTE 0          ;LOCK ON CURRENT TEST FLAG
717 001313     000          QV.FLG: .BYTE 0          ;QUICK VERIFY FLAG.
718
719           .EVEN
720           $Y-0          ;ON FIRST PASS OF EACH DV11 ITERATIONS WILL BE SUPPRESSE

```

```

721
722           ;DEFINITIONS FOR TRAP SUBROUTINE CALLS
723           ;POINTERS TO SUBROUTINFS CAN BE FOUND
724           ;IN THE TABLE IMMEDIATLY FOLLOWING THE DEFINITIONS

```

```

725           ;:*****
726           ;-----
727           ;TRPTAB:
728 001314     104400        SCOPE=TRAP+0          ;CALL TO SCOPE LOOP AND ITERATION HANDLER
729           .SCOPE
730 001314     002634        SCOPI-TRAP+1          ;CALL TO LOOP ON CURRENT DATA HANDLER
731           .SCOPI
732 001316     003020        TYPE=TRAP+2          ;CALL TO TELETYPE OUTPUT ROUTINE
733           .TYPE
734 001320     003044        INSTR=TRAP+3          ;CALL TO ASCII STRING INPUT ROUTINE
735           .INSTR
736 001322     003120        INSTER=TRAP+4         ;CALL TO INPUT ERROR HANDLER
737           .INSTER
738 001324     003224        PARAM=TRAP+5          ;CALL TO NUMERICAL DATA INPUT ROUTINE
739           .PARAM
740 001326     003244        SAV05=TRAP+6          ;CALL TO REGISTER SAVE ROUTINE
741           .SAV05
742 001330     003444        RES05=TRAP+7          ;CALL TO REGISTER RESTORE ROUTINE
743           .RES05
744 001332     003504        CONVRT=TRAP+10        ;CALL TO DATA OUTPUT ROUTINE
745           .CONVRT
746 001334     003526        CNVRT-TRAP+11        ;CALL TO DATA OUTPUT ROUTINE WITHOUT CR/LF.
747           .CNVRT
748 001336     003542        MSTCLR=TRAP+12        ;CALL TO ISUE A MASTER CLEAR
749           .MSTCLR
750 001340     004556        RAMCLR=TRAP+13        ;CALL TO CLEAR THE RAMS
751           .RAMCLR
752 001342     004516        DELAY=TRAP+14        ;CALL TO VARIABLE DELAY COUNTER
753           .DELAY
754 001344     004476        ROMCLK=TRAP+15        ;CALL TO CLOCK ROM ONCE
755           .ROMCLK
756 001346     004566        DATACLK=TRAP+16      ;CALL TO CLK DATA
757           .DATACLK
758 001350     004576
759
760           ;:*****
761           ;-----

```

```

762 ;DV11 VECTOR AND REGISTER INDIRECT POINTERS
763
764 001352 000000 DVRVEC: 0 ; POINTER TO DV11 RECEIVER INTERRUPT VECTOR
765 001354 000000 DVRLVL: 0 ; POINTER TO DV11 RECEIVER INTERRUPT SERVICE PS
766 001356 000000 DVTVEC: 0 ; POINTER TO DV11 TRANSMITTER INTERRUPT VECTOR
767 001360 000000 DVTLVL: 0 ; POINTER TO DV11 TRANSMITTER INTERRUPT SERVICE PS
768 001362 000000 DVSCR: 0 ; POINTER TO DV11 SYSTEM CONTROL REGISTER
769 001364 000000 DVSCRH: 0 ; POINTER TO DV11 SYSTEM CONTROL REGISTER HIGH BYTE.
770 001366 000000 DVRIC: 0 ; POINTER TO DV11 NEXT RECEIVED CHARACTER REGISTER
771 001370 000000 DVLCR: 0 ; POINTER TO DV11 LINE PRAMETER REGISTER
772 001372 000000 DVSRSH: 0 ; POINTER TO DV11 SECONDARY REGISTER SELECT REGISTER
773 001374 000000 DVSRSH: 0 ; POINTER TO DV11 SECONDARY REGISTER SELECT HIGH BYTE.
774 001376 000000 DVSRA: 0 ; POINTER TO DV11 SECONDARY REGISTER ACCESS REGISTER
775 001400 000000 DVSFR: 0 ; POINTER TO DV11 SPECIAL FUNCTIONS REGISTER
776 001402 000000 DVNSR: 0 ; POINTER TO DV11 NPR STATUS REGISTER
777 001404 000000 RESV16: 0 ; POINTER TO RESERVED REGISTER.
778
779
    
```

```

780 ;DV11 CONTROL INDICATORS FOR CURRENT DV11 UNDER TEST
781 :-----
782
783 001406 000000 MASK.A: .WORD 000 ; LAST CHAR TO TEST AND PARITY MASK FOR LINES 00-03
784 001410 000000 MASK.B: .WORD 000 ; LAST CHAR TO TEST AND PARITY MASK FOR LINES 04-07
785 001412 000000 MASK.C: .WORD 000 ; LAST CHAR TO TEST AND PARITY MASK FOR LINES 08-11
786 001414 000000 MASK.D: .WORD 000 ; LAST CHAR TO TEST AND PARITY MASK FOR LINES 12-15
787
788 001416 010 CLK.A: .BYTE 8. ; NUMBER OF CLOCKS NEEDED FOR ONE CHAR FOR LINES 00-03
789 001417 010 CLK.B: .BYTE 8. ; NUMBER OF CLOCKS NEEDED FOR ONE CHAR FOR LINES 04-07
790 001420 010 CLK.C: .BYTE 8. ; NUMBER OF CLOCKS NEEDED FOR ONE CHAR FOR LINES 08-11
791 001421 010 CLK.D: .BYTE 8. ; NUMBER OF CLOCKS NEEDED FOR ONE CHAR FOR LINES 12-15
792
793 001422 000000 L00.03: 000000 ; PARAMETERS FOR LINES 00-03
794 001424 000000 L04.07: 000000 ; PARAMETERS FOR LINES 04-07
795 001426 000000 L08.11: 000000 ; PARAMETERS FOR LINES 08-11
796 001430 000000 L12.15: 000000 ; PARAMETERS FOR LINES 12-15
797
798 001432 000000 SYNC2A: 000000 ; SYNC 2
799 001434 000000 SYNC2B: 000000 ;
800 001436 000000 SYNC2C: 000000 ;
801 001440 000000 SYNC2D: 000000 ;
    
```

```

802
803 ; SUMMARY
804 :-----
805 : MASK.X 040 5 BITS PER CHAR.
806 : 100 6 BITS PER CHAR.
807 : 200 7 BITS PER CHAR.
808 : 400 8 BITS PER CHAR.
809
810 : CLK.X 005 5 BITS PER CHAR.
811 : 006 6 BITS PER CHAR.
812 : 007 7 BITS PER CHAR.
813 : 010 8 BITS PER CHAR.
814 : IF PARITY IS ENABLED; ADD PLUS ONE TO THE ABOVE "CLK.X"
815 : FOR EACH GROUP THAT PARITY IS ENABLED.
    
```

```
816                                     ;DV11 STATUS TABLE AND ADDRESS ASSIGNMENTS
817                                     ;-----
818
819      001500      001500      . =1500
820      001500      000001      DV.MAP:
821      001500      000001      DVCR00: .BLKW 1      ;CONTROL STATUS REGISTER FOR DV11 NUMBER 00
822      001502      000001      DVTR00: .BLKW 1      ;VECTOR 'A' FOR DV11 NUMBER 00
823      001504      000001      DV00.A: .BLKW 1      ;PARAMETER FOR LINES 00-03 FOR DV11 NUMBER 00
824      001506      000001      SYNA00: .BLKW 1      ;SYNC TWO
825      001510      000001      DV00.B: .BLKW 1      ;PARAMETER FOR LINES 04-07 FOR DV11 NUMBER 00
826      001512      000001      SYNB00: .BLKW 1      ;SYNC TWO
827      001514      000001      DV00.C: .BLKW 1      ;PARAMETER FOR LINES 08-11 FOR DV11 NUMBER 00
828      001516      000001      SYNC00: .BLKW 1      ;SYNC TWO
829      001520      000001      DV00.D: .BLKW 1      ;PARAMETER FOR LINES 12-15 FOR DV11 NUMBER 00
830      001522      000001      SYND00: .BLKW 1      ;SYNC TWO
831
832      001524      000001      DVCR01: .BLKW 1      ;CONTROL STATUS REGISTER FOR DV11 NUMBER 01
833      001526      000001      DVTR01: .BLKW 1      ;VECTOR 'A' FOR DV11 NUMBER 01
834      001530      000001      DV01.A: .BLKW 1      ;PARAMETER FOR LINES 00-03 FOR DV11 NUMBER 01
835      001532      000001      SYNA01: .BLKW 1      ;SYNC TWO
836      001534      000001      DV01.B: .BLKW 1      ;PARAMETER FOR LINES 04-07 FOR DV11 NUMBER 01
837      001536      000001      SYNB01: .BLKW 1      ;SYNC TWO
838      001540      000001      DV01.C: .BLKW 1      ;PARAMETER FOR LINES 08-11 FOR DV11 NUMBER 01
839      001542      000001      SYNC01: .BLKW 1      ;SYNC TWO
840      001544      000001      DV01.D: .BLKW 1      ;PARAMETER FOR LINES 12-15 FOR DV11 NUMBER 01
841      001546      000001      SYND01: .BLKW 1      ;SYNC TWO
842
843      001550      000001      DVCR02: .BLKW 1      ;CONTROL STATUS REGISTER FOR DV11 NUMBER 02
844      001552      000001      DVTR02: .BLKW 1      ;VECTOR 'A' FOR DV11 NUMBER 02
845      001554      000001      DV02.A: .BLKW 1      ;PARAMETER FOR LINES 00-03 FOR DV11 NUMBER 02
846      001556      000001      SYNA02: .BLKW 1      ;SYNC TWO
847      001558      000001      DV02.B: .BLKW 1      ;PARAMETER FOR LINES 04-07 FOR DV11 NUMBER 02
848      001562      000001      SYNB02: .BLKW 1      ;SYNC TWO
849      001564      000001      DV02.C: .BLKW 1      ;PARAMETER FOR LINES 08-11 FOR DV11 NUMBER 02
850      001566      000001      SYNC02: .BLKW 1      ;SYNC TWO
851      001570      000001      DV02.D: .BLKW 1      ;PARAMETER FOR LINES 12-15 FOR DV11 NUMBER 02
852      001572      000001      SYND02: .BLKW 1      ;SYNC TWO
853
854      001574      000001      DVCR03: .BLKW 1      ;CONTROL STATUS REGISTER FOR DV11 NUMBER 03
855      001576      000001      DVTR03: .BLKW 1      ;VECTOR 'A' FOR DV11 NUMBER 03
856      001600      000001      DV03.A: .BLKW 1      ;PARAMETER FOR LINES 00-03 FOR DV11 NUMBER 03
857      001602      000001      SYNA03: .BLKW 1      ;SYNC TWO
858      001604      000001      DV03.B: .BLKW 1      ;PARAMETER FOR LINES 04-07 FOR DV11 NUMBER 03
859      001606      000001      SYNB03: .BLKW 1      ;SYNC TWO
860      001610      000001      DV03.C: .BLKW 1      ;PARAMETER FOR LINES 08-11 FOR DV11 NUMBER 03
861      001612      000001      SYNC03: .BLKW 1      ;SYNC TWO
862      001614      000001      DV03.D: .BLKW 1      ;PARAMETER FOR LINES 12-15 FOR DV11 NUMBER 03
863      001616      000001      SYND03: .BLKW 1      ;SYNC TWO
864
865      001620      000001      DVCR04: .BLKW 1      ;CONTROL STATUS REGISTER FOR DV11 NUMBER 04
866      001622      000001      DVTR04: .BLKW 1      ;VECTOR 'A' FOR DV11 NUMBER 04
867      001624      000001      DV04.A: .BLKW 1      ;PARAMETER FOR LINES 00-03 FOR DV11 NUMBER 04
868      001626      000001      SYNA04: .BLKW 1      ;SYNC TWO
869      001630      000001      DV04.B: .BLKW 1      ;PARAMETER FOR LINES 04-07 FOR DV11 NUMBER 04
870      001632      000001      SYNB04: .BLKW 1      ;SYNC TWO
871      001634      000001      DV04.C: .BLKW 1      ;PARAMETER FOR LINES 08-11 FOR DV11 NUMBER 04
```

```

872 001636 000001 SYNC04: .BLKW 1 ;SYNC TWO
873 001640 000001 DV04.D: .BLKW 1 ;PARAMETER FOR LINES 12-15 FOR DV11 NUMBER 04
874 001642 000001 SYND04: .BLKW 1 ;SYNC TWO
875
876 001644 000001 DVCR05: .BLKW 1 ;CONTROL STATUS REGISTER FOR DV11 NUMBER 05
877 001646 000001 DVTR05: .BLKW 1 ;VECTOR 'A' FOR DV11 NUMBER 05
878 001650 000001 DV05.A: .BLKW 1 ;PARAMETER FOR LINES 00-03 FOR DV11 NUMBER 05
879 001652 000001 SYNA05: .BLKW 1 ;SYNC TWO
880 001654 000001 DV05.B: .BLKW 1 ;PARAMETER FOR LINES 04-07 FOR DV11 NUMBER 05
881 001656 000001 SYN05: .BLKW 1 ;SYNC TWO
882 001660 000001 DV05.C: .BLKW 1 ;PARAMETER FOR LINES 08-11 FOR DV11 NUMBER 05
883 001662 000001 SYNC05: .BLKW 1 ;SYNC TWO
884 001664 000001 DV05.D: .BLKW 1 ;PARAMETER FOR LINES 12-15 FOR DV11 NUMBER 05
885 001666 000001 SYND05: .BLKW 1 ;SYNC TWO
886
887 001670 000001 DVCR06: .BLKW 1 ;CONTROL STATUS REGISTER FOR DV11 NUMBER 06
888 001672 000001 DVTR06: .BLKW 1 ;VECTOR 'A' FOR DV11 NUMBER 06
889 001674 000001 DV06.A: .BLKW 1 ;PARAMETER FOR LINES 00-03 FOR DV11 NUMBER 06
890 001676 000001 SYNA06: .BLKW 1 ;SYNC TWO
891 001700 000001 DV06.B: .BLKW 1 ;PARAMETER FOR LINES 04-07 FOR DV11 NUMBER 06
892 001702 000001 SYN06: .BLKW 1 ;SYNC TWO
893 001704 000001 DV06.C: .BLKW 1 ;PARAMETER FOR LINES 08-11 FOR DV11 NUMBER 06
894 001706 000001 SYNC06: .BLKW 1 ;SYNC TWO
895 001710 000001 DV06.D: .BLKW 1 ;PARAMETER FOR LINES 12-15 FOR DV11 NUMBER 06
896 001712 000001 SYND06: .BLKW 1 ;SYNC TWO
897
898 001714 000001 DVCR07: .BLKW 1 ;CONTROL STATUS REGISTER FOR DV11 NUMBER 07
899 001716 000001 DVTR07: .BLKW 1 ;VECTOR 'A' FOR DV11 NUMBER 07
900 001720 000001 DV07.A: .BLKW 1 ;PARAMETER FOR LINES 00-03 FOR DV11 NUMBER 07
901 001722 000001 SYNA07: .BLKW 1 ;SYNC TWO
902 001724 000001 DV07.B: .BLKW 1 ;PARAMETER FOR LINES 04-07 FOR DV11 NUMBER 07
903 001726 000001 SYN07: .BLKW 1 ;SYNC TWO
904 001730 000001 DV07.C: .BLKW 1 ;PARAMETER FOR LINES 08-11 FOR DV11 NUMBER 07
905 001732 000001 SYNC07: .BLKW 1 ;SYNC TWO
906 001734 000001 DV07.D: .BLKW 1 ;PARAMETER FOR LINES 12-15 FOR DV11 NUMBER 07
907 001736 000001 SYND07: .BLKW 1 ;SYNC TWO
908
909 001740 000000 DV.END: 000000
910
911 ;PROGRAM INITIALIZATION
912 ;LOCK OUT INTERRUPTS
913 ;SET UP PROCESSOR STACK
914 ;SET UP POWER FAIL VECTOR
915 ;CLEAR PROGRAM CONTROL FLAGS AND COUNTS
916 ;TYPE TITLE MESSAGE
917
918 001742 012737 000340 177776 .START: MOV #340,PS ;LOCK OUT INTERRUPTS
919 001750 012706 001200 MOV #STACK,SP ;SET UP STACK
920 001754 012737 004402 000024 MOV #.PFAIL,@#24 ;SET UP POWER FAIL VECTOR
921 001762 113737 001301 001303 MOVSB DVNUM,SAVNUM ;SAVE NUMBER OF DEVICES IN SYSTEM.
922 001770 005037 001230 CLR PASCNT ;CLEAR PASS COUNT
923 001774 105037 001311 CLRB ERRFLG ;CLEAR ERROR FLAG
924 002000 105037 001313 CLRB QV.FLG ;ZERO QUICK VERIFY FLAG
925 002004 012737 001500 001306 MOV #DV.MAP,CREAM ;GET MAP POINTER.
926 002012 112737 000001 001304 MOVSB #1,RUN ;POINT POINTER TO FIRST DEVICE.
927 002020 005037 001232 CLR ERRCNT ;CLEAR ERROR COUNT

```

928	002024	005037	001234		CLR	LSTERR		:CLEAR LAST ERROR POINTER
929	002030	012737	000001	001226	MOV	#1,TSTNO		:SET UP FOR TEST 1
930	002036	012737	001742	001214	MOV	#.START,RETURN		:SET UP FOR POWER FAIL BEFORE
931								:TESTING STARTS
932	002044	105737	001310		TSTB	INIFLG		:HAS INITIALIZATION BEEN PERFORMED
933	002050	001063			BNE	1\$		:BR IF YES
934	002052	013746	000004		MOV	4,-(SP)		
935	002056	013746	000006		MOV	6,-(SP)		
936	002062	005037	000006		CLR	6		
937	002066	012737	002104	000004	MOV	#80\$,4		
938	002074	005777	177102		TST	@SWR		
939	002100	000240			NOP			
940	002102	000407			BR	81\$		
941	002104	022626			CMP	(SP)+,(SP)+	80\$:	
942	002106	012737	000174	001200	MOV	#LIGHT,LIGHTS		
943	002114	012737	000176	001202	MOV	#SSWR,SWR		
944	002122	012637	000006		MOV	(SP)+,6	81\$:	
945	002126	012637	000004		MOV	(SP)+,4		
946	002132	104402	001000		TYPE	,MTITLE		:TYPE TITLE MESSAGE
947	002136	105137	001310		COMB	INIFLG		:IF NOT SET FLAG AND DO
948	002142	105777	177034		TSTB	@SWR		:BIT7=1??
949	002146	100402			BMI	16\$		:BR IF NO AUTO SIZE
950	002150	004737	006626		JSR	PC,CSRMAP		:GO DO THE AUTO SIZE
951	002154	104402	005461		TYPE	,XHEAD	16\$:	:TYPE HEADER
952	002160	012737	001500	001246	MOV	#DV.MAP,TEMP1		:SET POINTER
953	002166	017737	177054	001250	MOV	@TEMP1,TEMP2	5\$:	:SET DATA
954	002174	022737	177777	001250	CMP	#177777,TEMP2		:ALL DONE?
955	002202	001406			BEQ	1\$		:BR IF YES
956	002204	104410			CONVRT			
957	002206	005506			XSTATQ			
958	002210	062737	000002	001246	ADD	#2,TEMP1		:UPDATE POINTER
959	002216	000763			BR	5\$		
960	002220	005737	000042		TST	@#42	1\$:	:IS PROGRAM RUNNING UNDER MONITOR
961	002224	001030			BNE	3\$		:BR IF YES
962	002226	032777	000001	176746	BIT	#SW00,@SWR		:SELECT SPECIFIC DEVICES??
963	002234	001424			BEQ	3\$		:BR IF NO.
964	002236	104402	005402		TYPE	,MNEW		:TYPE THE MESSAGE.
965	002242	005000			CLR	R0		:ZERO DATA LIGHTS
966	002244	000000			HALT			:WAIT FOR USER TO TELL WHAT DEVICES TO RUN
967	002246	127737	176730	001302	CMPB	@SWR,SAVACT		:IS THE NUMBER VALID?
968	002254	101404			BLOS	2\$		:BR IF NUMBER IS OK.
969	002256	104402	005243		TYPE	,MERR3		:TELL USER OF INVALID NUMBER.
970	002262	000000			HALT			:STOP EVERY THING.
971	002264	000776			BR	.-2		:RESTART THE PROGRAM AGAIN.
972	002266	117737	176710	001300	MOVB	@SWR,DVACTV	2\$:	:GET NEW DEVICE PATTERN
973	002274	113700	001300		MOVB	DVACTV,R0		:SHOW THE USER WHAT HE SELECTED.
974	002300	042700	177400		BIC	#*C<377>,R0		:USE ONLY LOW BYTE.
975	002304	000000			HALT			:CONTINUE DYNAMIC SWITCHES.
976	002306	012700	000300		MOV	#300,R0	3\$:	:PREPARE TO CLEAR THE FLOATING
977	002312	012701	000302		MOV	#302,R1		:VECTOR AREA. 300-776
978	002316	010120			MOV	R1,(R0)+	4\$:	:START PUTTING 'PC+2 - HALT'
979	002320	005021			CLR	(R1)+		:IN VECTOR AREA.
980	002322	022021			CMP	(R0)+,(R1)+		:POP POINTERS
981	002324	022700	001000		CMP	#1000,R0		:ALL DONE??
982	002330	001372			BNE	4\$		:BR IF NO.
983								

```

984                                     ;TEST START AND RESTART
985                                     ;-----
986
987 002332 012737 000340 177776 .BEGIN: MOV #340,PS           ;LOCK OUT INTERRUPTS
988 002340 012706 001200          MOV #STACK,SP         ;SET UP STACK
989 002344 005737 000042          TST @#42             ;IS PROGRAM UNDER MONITOR CONTROL
990 002350 001023          BNE 3$              ;BR IF YES
991 002352 032777 000004 176622  BIT #BIT2,@SWR        ;CHECK FOR LOCK ON TEST
992 002360 001411          BEQ 1$              ;BR IF NO LOCK DESIRED.
993 002362 104402 005301          TYPE ,MLOCK         ;TYPE LOCK SELECTED.
994 002366 012737 000240 002702  MOV #NOP,TTST        ;ADJUST SCOPE ROUTINE.
995 002374 012737 000240 002704  MOV #NOP,TTST+2     ;SET UP TO LOCK
996 002402 000406          BR 2$              ;CONTINUE ALONG.
997 002404 013737 003014 002702 1$: MOV BRW,TTST        ;PREPARE NORMAL SCOPE ROUTINE
998 002412 013737 003016 002704  MOV BRX,TTST+2     ;LOCK NOT SELECTED, SET UP FOR NORMAL SCOPE LOOP
999 002420
1000 002420 012737 005666 001214 2$: MOV #CYCLE,RETURN    ;START AT 'CYCLE' FIND WHICH DEVICE TO TEST
1001 002426 104402 005171          TYPE ,MR            ;TYPE R
1002 002432 000177 176556          JMP @RETURN        ;START TESTING

```



```

1003          :END OF PASS
1004          :TYPE NAME OF TEST
1005          :UPDATE PASS COUNT
1006          :CHECK FOR EXIT TO ACT-11
1007          :RESTART TEST
1008
1009 002436 000005          .EOP:  RESET          ;MAKE THE WORLD CLEAN AGAIN.
1010 002440 005037 001234  CLR          LSTERR          ;CLEAR LAST ERROR PC
1011 002444 105037 001311  CLR          ERRFLG          ;CLEAR ERROR FLAG
1012 002450 005237 001230  INC          PASCNT          ;UPDATE PASS COUNT
1013 002454 013777 001230 176516 MOV          PASCNT,@LIGHTS ;DISPLAY PASS COUNT
1014 002462 104402 005145  TYPE          ,MEPASS        ;TYPE END PASS
1015 002466 104402 005330  TYPE          ,MCSR          ;TYPE CSR
1016 002472 104411 002604  CNVRT         ,XCSR          ;SHOW IT
1017 002476 104402 005336  TYPE          ,MVECX         ;TYPE VECTOR
1018 002502 104411 002612  CNVRT         ,XVEC          ;SHOW IT
1019 002506 104402 005344  TYPE          ,MPASSX        ;TYPE PASSES
1020 002512 104411 002620  CNVRT         ,XPASS         ;SHOW IT
1021 002516 104402 005355  TYPE          ,MERRX        ;TYPE ERRORS
1022 002522 104411 002626  CNVRT         ,XERR          ;SHOW IT
1023 002526 105337 001303  DECB         SAVNUM         ;ARE ALL DEVICES TESTED?
1024 002532 001017          BNE          RESTR          ;BR IF NO.
1025 002534 112737 000377 001313 MOV          #377,@V.FLG     ;SET THE QUICK VERIFY FLAG.
1026 002542 113737 001301 001303 MOV          DVNUM,SAVNUM    ;RESTORE THE COUNT
1027 002550 013701 000042          MOV          @#42,R1        ;CHECK FOR ACT-11 OR DDP
1028 002554 001406          BEQ          RESTR          ;IF NOT, CONTINUE TESTING
1029 002556 000005          RESET          ;STOP THE SHOW--CLEAR THE WORLD
1030 002560
1031 002560 004711          LOGICAL: JSR          PC,(R1)
1032 002562 000240          NOP
1033 002564 000240          NOP
1034 002566 000240          NOP
1035 002570 000240          NOP
1036 002572 012737 005666 001214 RESTR: MOV          #CYCLE,RETURN
1037 002600 000137 005666          JMP          CYCLE
1038 002604 000001          XCSR:  1
1039 002606          006          .BYTE          6,2
1040 002610 001362          DVSCR
1041 002612 000001          XVEC:  1
1042 002614          003          .BYTE          3,2
1043 002616 001352          DVRVEC
1044 002620 000001          XPASS: 1
1045 002622          006          .BYTE          6,2
1046 002624 001230          PASCNT
1047 002626 000001          XERR:  1
1048 002630          006          .BYTE          6,2
1049 002632 001232          ERRCNT
1050
1051          :SCOPE LOOP AND INTERATION HANDLER
1052          :-----
1053
1054 002634          .SCOPE:
1055 002634 022737 177570 001202 CMP          #177570,SWR     ;IS THERE A REAL SWR?
1056 002642 001411          BEQ          64$          ;BR IF YES
1057 002644 017746 176336 MOV          @TKDBR,-(SP)    ;SAVE KEYBOARD CHAR
1058 002650 042716 000200 BIC          #BIT7,(SP)     ;CLEAR PARITY BIT

```

```

1059 002654 122726 000007      CMPB   #7,(SP)+      ;WAS IT CNTRL 'G' ?
1060 002660 001002      BNE    .+6          ;BR IF NO.
1061 002662 004737 004640      JSR    PC,SERV.G    ;SERVICE 'CNTRL 'G''.
1062 002666 005037 001234      64$:  CLR    LSTERR     ;CLEAR LAST ERROR PC.
1063 002672 010016      MOV    RO,(SP)      ;SAVE RO ON THE STACK
1064 002674 032777 040000 176300  BIT    #BIT14,@SWR  ;'LOOP ON THIS TEST'?
1065 002702 001407      TTST: BEQ    1$          ;BR IF NO. (IF LOCK SW01=1; THIS LOC =240)
1066 002704 000437      BR    3$           ;GOTO 3$ (IF LOCK SW01=1; THIS LOC =240)
1067 002706 105777 176272      TSTB  @TKCSR       ;KEYBOARD DONE?
1068 002712 100034      BPL   3$          ;BR IF NO. (LOCK: HIT KEY TO GOTO NEXT TEST)
1069 002714 017700 176266      MOV    @TKDBR,RO   ;CLEAR DONE BIT
1070 002720 000415      BR    2$          ;CONTINUE
1071 002722 032777 004000 176252  1$:  BIT    #SW11,@SWR  ;DE ETE ITERATION? (QUICK PASS)
1072 002730 001011      BNE   2$          ;BR IF YES
1073 002732 105737 001313      TSTB  QV.FLG      ;HAVE PASSES BEECOMPLETED?
1074 002736 001406      BEQ   2$          ;BR IF QUICK PASS.
1075 002740 005237 001224      INC    LPCNT       ;UPDATE ITERATION COUNTER
1076 002744 023737 001224 001222  CMP    LPCNT,ICOUNT ;ARE ALL ITERATIONS DONE??
1077 002752 001014      BNE   3$          ;BR IF NOT YET
1078 002754 105037 001311      2$:  CLRB  ERRFLG     ;PREPARE FOR NEW TEST
1079 002760 005037 001224      CLR    LPCNT       ;START ICOUNTER AT 0
1080 002764 005037 001220      CLR    LOCK
1081 002770 012737 000020 001222  MOV    #20,ICOUNT  ;RESET ITERATIONS
1082 002776 013737 001216 001214  MOV    NEXT,RETURN ;GET NEXT TEST
1083 003004 011600      3$:  MOV    (SP),RO    ;POP RO OFF OF THE STACK
1084 003006 022626      POP2SP ;FAKE AN 'RTI'
1085 003010 000177 176200      JMP   @RETURN      ;GO DO THE TEST
1086 003014 001407      BRW: 1407
1087 003016 000437      BRX: 437
1088
1089      ;CHECK FOR FREEZE ON CURRENT DATA
1090      ;-----
1091
1092 003020 032777 001000 176154  .SCOPI: BIT    #SW09,@SWR  ;IS SW09=1(SET)?
1093 003026 001405      BEQ   1$          ;BR IF NOT SET.
1094 003030 005737 001220      TST   LOCK
1095 003034 001402      BEQ   1$
1096 003036 013716 001220      MOV    LOCK,(SP)  ;GOTO THE ADDRESS IN LOCK.
1097 003042 000002      1$:  RTI             ;GO BACK.
1098
1099      ;TELETYPE OUTPUT ROUTINE
1100      ;-----
1101
1102 003044 010546      .TYPE: MOV    R5,-(SP) ;SAVE R5 ON THE STACK.
1103 003046 017605      MOV    @2(SP),R5  ;GET ADDRESS OF MESSAGE.
1104 003052 062766 000002 000002  ADD    #2,2(SP)   ;POP OVER ADDRESS.
1105 003060 032777 010000 176114  1$:  BIT    #SW12,@SWR ;INHIBIT ALL PRINT OUT??
1106 003066 001012      BNE   3$          ;BR IF NO PRINT OUT WANTED (SW12=1)
1107 003070 105715      TSTB  (R5)        ;IS NUMBER MINUS? (MSB=1(BIT7))
1108 003072 100002      BPL   2$          ;BR IF NUMBER IS PLUS
1109 003074 104402 005104      TYPE  ,MCRLF     ;TYPE A CR/LF!
1110 003100 105777 176104      2$:  TSTB  @TPCSR    ;TTY READY?
1111 003104 100375      BPL   2$          ;BR IF NO.
1112 003106 112577 176100      MOVB  (R5)+,@TPDBR ;PRINT CURRENT CHAR.
1113 003112 001362      BNE   1$          ;IF NOT ZERO KEEP PRINTING.
1114 003114 012605      3$:  MOV    (SP)+,R5 ;END OF OUTPUT. RESTORE R5

```

```

1115 003116 000002          RTI          ;GO HOME
1116          ;-----
1117
1118 003120 017346          .INSTR: MOV      R3,-(SP)          ;SAVE R3 ON STACK
1119 003122 010440          MOV      R4,-(SP)          ;SAVE R4 ON STACK
1120 003124 017637 000004 003142  MOV      @4(SP),.MSG
1121 003132 062766 000002 000004  ADD      #2,4(SP)
1122 003140 104402          .INST1: TYPE
1123 003142 000000          .MSG: 0
1124 003144 012704 005520          MOV      #INBUF,R4
1125 003150 012703 000007          MOV      #7,R3
1126 003154 105777 176024          1$:  TSTB     @TKCSR
1127 003160 100375          BPL      1$
1128 003162 117714 176020          MOVB     @TKDBR,(R4)
1129 003166 142714 000200          BICB     #200,(R4)
1130 003172 122427 000015          CMPB     (R4)+,#15
1131 003176 001417          BEQ      INSTR2
1132 003200 105777 176004          2$:  TSTB     @TPCSR
1133 003204 100375          BPL      2$
1134 003206 017777 175774 175776  MOV      @TKDBR,@TPDBR
1135 003214 005303          DEC      R3
1136 003216 001356          BNE      1$
1137 003220 012604          MOV      (SP)+,R4
1138 003222 012603          MOV      (SP)+,R3
1139 003224 104402 005100          .INSTE: TYPE ,MQM
1140 003230 010346          MOV      R3,-(SP)
1141 003232 010446          MOV      R4,-(SP)
1142 003234 000741          BR       .INST1
1143 003236 012604          INSTR2: MOV     (SP)+,R4          ;RESTORE R4
1144 003240 012603          MOV     (SP)+,R3          ;RESTORE R3
1145 003242 000002          RTI
1146
1147          ;CONVERT ASCII STRING TO OCTAL
1148          ;-----
1149
1150 003244 010546          .PARAM: MOV     R5,-(SP)
1151 003246 010446          MOV     R4,-(SP)
1152 003250 016605 000004          MOV     4(SP),R5
1153 003254 012537 003434          MOV     (R5)+,LOLIM
1154 003260 012537 003436          MOV     (R5)+,HILIM
1155 003264 012537 003440          MOV     (R5)+,DEVADR
1156 003270 112537 003442          MOVB    (R5)+,LOBITS
1157 003274 112537 003443          MOVB    (R5)+,ADRCNT
1158 003300 010566 000004          MOV     R5,4(SP)
1159 003304 005005          PARAM1: CLR     R5
1160 003306 012704 005520          MOV     #INBUF,R4
1161 003312 122714 000015          CMPB    #15,(R4)
1162 003316 001420          BEQ     PARERR
1163 003320 121427 000060          1$:  CMPB    (R4),#60
1164 003324 002415          BLT     PARERR
1165 003326 121427 000067          CMPB    (R4),#67
1166 003332 003012          BGT     PARERR
1167 003334 142714 000060          BICB    #60,(R4)
1168 003340 152405          BISB    (R4)+,R5
1169 003342 122714 000015          CMPB    #15,(R4)
1170 003346 001406          BEQ     LIMITS

```

```

1171 003350 006305          ASL    R5
1172 003352 006305          ASL    R5
1173 003354 006305          ASL    R5
1174 003356 000760          BR     1$
1175 003360 104404          PARERR: INSTER
1176 003362 000750          BR     PARAM1
1177
1178                          ;TEST TO SEE IF NUMBER IS WITHIN LIMITS
1179                          :-----
1180
1181 003364 020537 003436          LIMITS: CMP    R5,HILIM
1182 003370 101373          BHI    PARERR
1183 003372 020537 003434          CMP    R5,LOLIM
1184 003376 103770          BLO    PARERR
1185 003400 133705 003442          BITB  LOBITS,R5
1186 003404 001365          BNE    PARERR
1187
1188                          ;STORE NUMBER AT SPECIFIED ADDRESS
1189
1190 003406 013704 003440          1$:  MOV    DEVADR,R4
1191 003412 010524          MOV    R5,(R4)+
1192 003414 062705 000002          ADD    #2,R5
1193 003420 105337 003443          DECB  ADRCNT
1194 003424 001372          BNE    1$
1195 003426 012604          MOV    (SP)+,R4
1196 003430 012605          MOV    (SP)+,R5
1197 003432 000002          RTI
1198 003434 000000          LOLIM: 0
1199 003436 000000          HILIM: 0
1200 003440 000000          DEVADR: 0
1201 003442 000000          LOBITS: 0
1202          ADRCNT=LOBITS+1
1203
1204                          ;SAVE PC OF TEST THAT FAILED AND R0-R5
1205                          :-----
1206
1207 003444 016637 000004 001276 .SAV05: MOV    4(SP),SAVPC      ;SAVE R7 (PC)
1208
1209                          ;SAVE R0-R5
1210
1211 003452 010537 001272          SV05: MOV    R5,SAVR5      ;SAVE R5
1212 003456 010437 001270          MOV    R4,SAVR4      ;SAVE R4
1213 003462 010337 001266          MOV    R3,SAVR3      ;SAVE R3
1214 003466 010237 001264          MOV    R2,SAVR2      ;SAVE R2
1215 003472 010137 001262          MOV    R1,SAVR1      ;SAVE R1
1216 003476 010037 001260          MOV    R0,SAVR0      ;SAVE R0
1217 003502 000002          RTI                  ;LEAVE.
1218
1219                          ;RESTORE R0-R5
1220
1221 003504 013700 001260          .RES05: MOV    SAVR0,R0      ;RESTORE R0
1222 003510 013701 001262          MOV    SAVR1,R1      ;RESTORE R1
1223 003514 013702 001264          MOV    SAVR2,R2      ;RESTORE R2
1224 003520 013703 001266          MOV    SAVR3,R3      ;RESTORE R3
1225 003524 013704 001270          MOV    SAVR4,R4      ;RESTORE R4
1226 003530 013705 001272          MOV    SAVR5,R5      ;RESTORE R5

```

```

1227 003534 000002          RTI          ;LEAVF
1228
1229          ;CONVERT OCTAL NUMBER TO ASCII AND OUTPUT TO TELEPRINTER
1230          ;-----
1231
1232 003536 104402 005104    .CONVR: TYPE      ,MCRLF
1233 003542 010046          .CNVRT: MOV        R0,-(SP)
1234 003544 010146          MOV        R1,-(SP)
1235 003546 010346          MOV        R3,-(SP)
1236 003550 010446          MOV        R4,-(SP)
1237 003552 010546          MOV        R5,-(SP)
1238 003554 017601 000012    MOV        @12(SP),R1
1239 003560 062766 000002 000012    ADD        #2,12(SP)
1240 003566 012137 003742    MOV        (R1)+,WRDCNT
1241 003572 112137 003744    1$: MOVB      (R1)+,CHRCNT
1242 003576 112137 003745    MOVB      (R1)+,SPACNT
1243 003602 013137 003746    MOV        @ (R1)+,BINWRD
1244 003606 013704 003746    2$: MOV        BINWRD,R4
1245 003612 113705 003744    MOVB      CHRCNT,R5
1246 003616 012700 005562    MOV        #TEMP,R0
1247 003622 010403    3$: MOV        R4,R3
1248 003624 042703 177770    BIC        #177770,R3
1249 003630 062703 000060    ADD        #060,R3
1250 003634 110320    MOVB      R3,(R0)+
1251 003636 000241    CLC
1252 003640 006004    ROR        R4
1253 003642 000241    CLC
1254 003644 006004    ROR        R4
1255 003646 000241    CLC
1256 003650 006004    ROR        R4
1257 003652 005305    DEC        R5
1258 003654 001362    BNE        3$
1259 003656 012703 005624    MOV        #MDATA,R3
1260 003662 114023    4$: MOVB      -(R0),(R3)+
1261 003664 105337 003744    DECB      CHRCNT
1262 003670 001374    BNE        4$
1263 003672 105737 003745    TSTB      SPACNT
1264 003676 001405    BEQ        6$
1265 003700 112723 000040    5$: MOVB      #040,(R3)+
1266 003704 105337 003745    DECB      SPACNT
1267 003710 001373    BNE        5$
1268 003712 105013    6$: CLRB      (R3)
1269 003714 104402 005624    TYPE      ,MDATA
1270 003720 005337 003742    DEC        WRDCNT
1271 003724 001322    BNE        1$
1272 003726 012605    MOV        (SP)+,R5
1273 003730 012604    MOV        (SP)+,R4
1274 003732 012603    MOV        (SP)+,R3
1275 003734 012601    MOV        (SP)+,R1
1276 003736 012600    MOV        (SP)+,R0
1277 003740 000002    RTI
1278 003742 000000    WRDCNT: 0
1279 003744 000000    CHRCNT: 0
1280          003745    SPACNT=CHRCNT+1
1281 003746 000000    BINWRD: 0
1282

```

```

1283
1284 ;TRAP DISPATCH SERVICE
1285 ;ARGUMENT OF TRAP IS EXTRACTED
1286 ;AND USED AS OFFSET TO OBTAIN POINTER
1287 ;TO SELECTED SUBROUTINE
1288
1289 003750 C11646 .TRPSR: MOV (SP),-(SP) ;GET PC OF RETURN
1290 003752 162716 000002 SUB #2,(SP) ;=PC OF TRAP
1291 003756 017616 000000 MOV @ (SP),(SP) ;GET TRP
1292 003762 006316 TRPOK: ASL (SP) ;MULTIPLY TRAP ARG BY 2
1293 003764 042716 177001 BIC #177001,(SP) ;CLEAR UNWANTED BITS
1294 003770 062716 001314 ADD #.TRPTAB,(SP) ;POINTER TO SUBROUTINE ADDRESS
1295 003774 017616 000000 MOV @ (SP),(SP) ;SUBROUTINE ADDRESS
1296 004000 000136 JMP @ (SP)+ ;GO TO SUBROUTINE
1297
1298 ;ERROR HANDLER
1299 ;-----
1300
1301 004002 .HLT:
1302 004002 022737 177570 001202 CMP #177570,SWR ;IS THERE A REAL SWR?
1303 004010 001411 BEQ 64$ ;BR IF YES
1304 004012 017746 175170 MOV @TKDBR, -(SP) ;SAVE KEYBOARD CHAR
1305 004016 042716 000200 BIC #BIT7,(SP) ;CLEAR PARITY BIT
1306 004022 122726 000007 CMPSB #7,(SP)+ ;WAS IT CNTRL 'G' ?
1307 004026 001002 BNE .+6 ;BR IF NO.
1308 004030 004737 004640 JSR PC,SERV.G ;SERVICE 'CNTRL 'G''.
1309 004034 032777 010000 175140 64$: BIT #SW12,@SWR ;BELL ON ERROR?
1310 004042 001406 BEQ XB$ ;BR IF NO BELL
1311 004044 105777 175140 TSTB @TPCSR ;TTY READY.
1312 004050 100003 BPL XB$ ;DON'T WAIT IF TTY NOT READY.
1313 004052 112777 000207 175132 MOVB #207,@TPDBR ;PUSH A BELL AT THE TTY.
1314 004060 032777 020000 175114 XB$: BIT #SW13,@SWR ;DELETE ERROR PRINT OUT?
1315 004066 001105 BNE HALTS ;BR IF NO PRINT OUT WANTED.
1316 004070 021637 001234 CMP (SP),LSTERR ;WAS THIS ERROR FOUND LAST TIME?
1317 004074 001404 BEQ 1$ ;BR IF YES
1318 004076 011637 001234 MOV (SP),LSTERR ;RECORD BEING HERE
1319 004102 105037 001311 CLRB ERRFLG ;PREPARE HEADER
1320 004106 104406 1$: SAVOS ;SAVE ALL PROC REGISTERS
1321 004110 011605 MOV (SP),R5 ;GET THE PC OF ERROR
1322 004112 162705 000002 SUB #2,R5 ;GET ADDRESS OF TRAP CALL
1323 004116 011504 MOV (R5),R4 ;GET HLT INSTRUCTION
1324 004120 006304 ASL R4 ;MULT BY TWO
1325 004122 061504 ADD (R5),R4 ;DOUBLE IT
1326 004124 006304 ASL R4 ;MULT AGAIN
1327 004126 042704 177001 BIC #177001,R4 ;CLEAR JUNK
1328 004132 062704 033520 ADD #.ERRTAB,R4 ;GET POINTER
1329 004136 012437 004252 MOV (R4)+,ERRMSG ;GET ERROR MESSAGE
1330 004142 012437 004264 MOV (R4)+,DATAHD ;GET DATA HEADRER
1331 004146 011437 004276 MOV (R4),DATABP ;GET DATA TABLE
1332 004152 105737 001311 TSTB ERRFLG ;TYPE HEADREER
1333 004156 001403 BEQ TYPMSG ;BR IF YES
1334 004160 005737 004276 TST DATABP ;DOES DATA TABLE EXIST?
1335 004164 001040 BNE TYPDAT ;BR IF YES.
1336 004166 104402 005104 TYPMSG: TYPE ,MCRLF
1337 004172 104402 005104 TYPE ,MCRLF
1338 004176 005737 001220 TST LOCK

```



```

1339 004202 001402          BEQ      1$
1340 004204 104402 005400    TYPE    ,MASTEK
1341 004210 104402 005366    1$:    TYPE    ,MTSTN
1342 004214 104411 004374    CNVRT   ,XTSTN          ;SHOW IT
1343 004220 104402 005454    TYPE    ,MERRPC        ;TYPE PC.
1344 004224 104411 004366    CNVRT   ,ERTABO         ;SHOW IT
1345 004230 104402 005104    TYPE    ,MCRLF         ;GIVE A CR/LF
1346 004234 112737 177777 001311 MOVB    #-1,ERRFLG     ;NO MORE HEADER UNLESS NO DATA TABLE.
1347 004242 005737 004252    TST     ERRMSG        ;IS THERE AN ERROR MESSAGE?
1348 004246 001402          BEQ      WRKO.FM       ;BR IF NO.
1349 004250 104402          TYPE
1350 004252 000000    ERRMSG: 0            ;ERROR MESSAGE
1351 004254          WRKO.FM:
1352 004254 005737 004264    TST     DATAHD      ;DATA HEADER?
1353 004260 001402          BEQ      TYPDAT      ;BR IF NO
1354 004262 104402          TYPE
1355 004264 000000    DATAHD: 0          ;DATA HEADER
1356 004266 005737 004276    TYPDAT: TST     DATABP ;DATA TABLE?
1357 004272 001402          BEQ      RESREG     ;BR IF NO.
1358 004274 104410          CNVRT
1359 004276 000000    DATABP: 0            ;DATA TABLE
1360 004300 104407          RESREG: RESO5      ;RESTORE PROC REGISTERS
1361 004302 005777 174674    HALTS:  TST     @SWR   ;HALT ON ERROR?
1362 004306 100005          BPL     EXITER      ;BR IF NO HALT ON ERROR
1363 004310 010046          PUSHRO ;SAVE RO
1364 004312 016600 000002    MOV     2(SP),RO     ;SHOW ERROR PC IN DATA LIGHTS
1365 004316 000000          HALT
1366 004320 012600          POPRO  ;GET RO
1367 004322 005237 001232    EXITER: INC     ERRCNT ;UPDATE ERROR COUNT
1368 004326 032777 000400 174646 BIT     #SW08,@SWR   ;GOTO TOP OF TEST?
1369 004334 001007          BNE     1$          ;BR IF YES
1370 004336 032777 002000 174636 BIT     #SW10,@SWR   ;GOTO NEXT TEST?
1371 004344 001407          BEQ     2$          ;BR IF NO
1372 004346 013737 001216 001214 MOV     NEXT,RETURN  ;SET FOR NEXT TEST
1373 004354 012706 001200    1$:    MOV     #STACK,SP  ;RESET SP
1374 004360 000177 174630    JMP     @RETJRN     ;GOTO SPECIFIED TEST
1375 004364 000002          2$:    RTI
1376 004366 000001          ERTABO: 1          ;RETURN
1377 004370          .BYTE 6,2
1378 004372 001276          SAVPC
1379 004374 000001          XTSTN: 1
1380 004376          .BYTE 3,2
1381 004400 001226          TSTNO
1382          ;ENTER HERE ON POWER FAILURE
1383          ;-----
1384
1385
1386 004402          .PFAIL:
1387 004402 012737 004414 000024 MOV     #RESTART,24 ;SET UP FOR POWER UP TRAP
1388 004410 000000          HALT           ;HALT ON POWER DOWN NORMAL
1389 004412 000777          BR
1390
1391          ;PROCESSOR WILL TRAP HERE WHEN POWER IS RESTORED
1392
1393 004414          RESTAR:
1394 004414 012737 004402 000024 MOV     #.PFAIL,24 ;SET UP FOR POWER FAILURE

```

```

1395 004422 012706 001200      MOV      #STACK,SP      ;RESET THE STACK POINTER
1396 004426 005037 005562      CLR      TEMP           ;READY FOR TIMER
1397 004432 005237 005562      INC      TEMP           ;PLUS ONE TO THE TIMER.
1398 004436 001375              BNE      -4             ;BR IF MORE TO GO
1399 004440 104402 005107      TYPE     ,MPFAIL       ;TYPE THE MESSAGE
1400 004444 104411 004470      CNVRT   ,PFTAB         ;TELL WHAT TEST TO RETURN TO.
1401 004450 105037 001311      CLR     ERRFLG         ;START CLEAN
1402 004454 005037 001234      CLR     LSTERR         ;.....
1403 004460 104412              MSTCLR                    ;START CLEAN UP OF DEVICE
1404 004462 104413              RAMCLR                    ;CLEAR IT ALL!
1405 004464 000177 174524      JMP     @RETURN        ;START DOING THAT TEST AGAIN.
1406 004470 000001              PFTAB: 1
1407 004472      003      002      .BYTE 3,2
1408 004474 001226              .DELAY: TSTNO
1409 004476 010046              .MOV    R0,-(SP)
1410 004500 013700 004514      .MOV    1$,R0
1411 004504 005300              .DEC    R0
1412 004506 001376              .BNE    -2
1413 004510 012600              .MOV    (SP)+,R0
1414 004512 000002              .RTI
1415 004514 000036              1$: 30.
1416
1417 004516              .RAMCLR:
1418 004516 012777 004000 174636      .MOV    #MRESET,@DVSCR ;ISSUE A MASTER CLEAR
1419 004524 010146              .MOV    R1,-(SP)       ;SAVE R1 ON THE STACK
1420 004526 010446              .MOV    R4,-(SP)       ;SAVE R4 ON THE STACK
1421 004530 013701 001372      .MOV    DVSR5,R1       ;GET SECONDARY SEL. REG.
1422 004534 013704 001376      .MOV    DVSR4,R4       ;GET SECONDARY REGISTER ACCESS REG.
1423 004540 005014              1$: .CLR    (R4)         ;ZERO THE SECONDARY REGISTER.
1424 004542 062711 170361      .ADD    #^C<BIT11+BIT10+BIT9+BIT8+BIT3+BIT2+BIT1+BIT0>+BIT0,(R1)
1425 004546 001374              .BNE    1$
1426 004550 012604              .MOV    (SP)+,R4       ;RESTORE R4
1427 004552 012601              .MOV    (SP)+,R1       ;RESTORE R1
1428 004554 000002              .RTI
1429
1430 004556              .MSTCLR:
1431 004556 012777 004000 174576      .MOV    #MRESET,@DVSCR ;ISSUE MASTER CLEAR.
1432 004564 000002              .RTI
1433
1434 004566              .ROMCLK:
1435 004566 052777 000002 174566      .BIS    #BIT1,@DVSCR
1436 004574 000002              .RTI
1437
1438 004576              .DATACLK:
1439 004576 010046              .MOV    R0,-(SP)
1440 004600 005000              .CLR    R0
1441 004602 052777 000400 174560      .BIS    #BIT8,@DVLCR
1442 004610 017737 174554 004636      1$: .MOV    @DVLCR,3$
1443 004616 106037 004637      .RORB   3$+1
1444 004622 103003              .BCC    2$
1445 004624 005200              .INC    R0
1446 004626 001370              .BNE    1$
1447 004630 104000              .HLT    0
1448 004632 012600              2$: .MOV    (SP)+,R0
1449 004634 000002              .RTI
1450 004636 000001              3$: .BLKW 1

```

```

1451
1452 004640 032777 004000 174336 SERV.G: BIT #4000,@TKCSR ;RX BUSY?
1453 004646 001374 BNE SERV.G ;BR IF YES
1454 004650 017737 174326 005072 MOV @SWR,90$ ;SAVE (SWR).
1455 004656 013777 005072 174316 1$: MOV 90$,@SWR
1456 004664 104402 005052 TYPE ,89$
1457 004670 104411 005064 (NVRT ,88$
1458 004674 104402 005074 TYPE ,91$
1459 004700 105777 174300 TSTB @TKCSR ;WAIT FOR DONE.
1460 004704 100375 BPL , -4
1461 004706 017746 174274 MOV @TKDDBR,-(SP)
1462 004712 042716 000200 BIC #BIT7,(SP)
1463 004716 122726 000015 (MPB #15,(SP)+
1464 004722 001450 BEQ 5$
1465 004724 005077 174252 CLR @SWR
1466 004730 105777 174254 2$: TSTB @TPCSR
1467 004734 100375 BPL , -4
1468 004736 016677 177776 174246 MOV -2(SP),@TPDDBR
1469 004744 000241 CLC
1470 004746 006177 174230 ROL @SWR
1471 004752 006177 174224 ROL @SWR
1472 004756 006177 174220 ROL @SWR
1473 004762 103735 BCS 1$ ERROR
1474 004764 026627 177776 000060 (MP -2(SP),#60
1475 004772 002731 BLT 1$
1476 004774 026627 177776 000067 (MP -2(SP),#67
1477 005002 003325 BGT 1$
1478 005004 042766 177770 177776 BIC #^C<7>,-2(SP)
1479 005012 056677 177776 174162 BIS -2(SP),@SWR
1480 005020 105777 1741 TSTB @TKCSR
1481 005024 100375 BPL , -4
1482 005026 017746 174154 MOV @TKDDBR,-(SP)
1483 005032 042716 000200 BIC #BIT7,(SP)
1484 005036 122726 000015 (MPB #15,(SP)+
1485 005042 001332 BNE 2$
1486 005044 104402 005104 5$: TYPE ,MCRLF
1487 005050 000207 RTS PC
1488
1489 005052 020377 051450 051127 89$: .ASCIZ <377>? (SWR)=/?
1490 005060 036451 000057
1491 .EVEN
1492 005064 000001 88$: 1
1493 005066 006 000 .BYTE 6,0
1494 005070 005072 90$
1495 005072 000000 90$: .WORD 0
1496 005074 036457 000057 91$: .ASCIZ ?/-/?
1497 .EVEN
1498 005100 020040 000077 MQM: .ASCIZ / ?/
(2) 005104 005015 000 MCRLF: .ASCIZ <15><12>
(2) 005107 377 053520 020122 MPFAIL: .ASCIZ <377>/PWR FAILED. RESTART AT TEST /
(2) 005145 377 047105 020104 MEPASS: .ASCIZ <377>/END PASS CZDVA-C /
(2) 005171 377 000122 MR: .ASCIZ <377>/R/
(2) 005174 050377 047522 051107 MERR2: .ASCIZ <377>/PROGRAM INDICATES NO DEVICES PRESENT./
(2) 005243 377 047111 052523 MERR3: .ASCIZ <377>/INSUFFICIENT DATA./
(2) 005267 377 042524 052123 MTSTPC: .ASCIZ <377>/TEST PC-/
(2) 005301 377 047514 045503 MLOCK: .ASCIZ <377>/LOCK ON SELECTED TEST/

```

```

(2) 005330 051503 035122 000040 MCSR: .ASCIZ /CSR: /
(2) 005336 042526 035103 000040 MVEC: .ASCIZ /VEC: /
(2) 005344 040520 051523 051505 MPASS: .ASCIZ /PASSES: /
(2) 005355 105 051122 051117 MERR: .ASCIZ /ERRORS: /
(2) 005366 042524 052123 047040 MTSTN: .ASCIZ /TEST NO: /
(2) 005400 000052 MASTE: .ASCIZ /*/
(2) 005402 051777 052105 051440 MNEW: .ASCIZ <377>/SET SWITCH REG TO DV11'S DESIRED ACTIVE./
(2) 005454 041520 020072 000 MERRP: .ASCIZ /PC: /
(2) 005461 377 040515 020120 XHEAD: .ASCIZ <377>/MAP OF DV11 STATUS/<377>
(2) .EVEN
(2) 005506 000002 XSTATQ: 2
1499 005510 006 003 .BYTE 6,3
1500 005512 001246 TEMP1
1501 005514 006 002 .BYTE 6,2
1502 005516 001250 TEMP2
1503 .EVEN
1504
1505 ;BUFFERS FOR INPUT-OUTPUT
1506
1507 005520 000000 INBUF: 0
1508 005562 .+.40
1509 005562 000000 TEMP: 0
1510 005624 .+.40
1511 005624 000000 MDATA: 0
1512 005666 .+.40

```

```

1513
1514
1515          ;ROUTINE USED TO 'CYCLE' THROUGH UP TO EIGHT DV11'S
1516          ;THIS ROUTINE SETS UP THE CONTROL ADDRESS FOR THE DIAGNOSTIC
1517          ;AND RUNS THE SPECIFIED DV11'S.  THIS ROUTINE *MUST*
1518          ;BE RUN FIRST BEFORE ENTERING THE DIAGNOSTIC FOR THE
1519          ;SETUP NECESSARY.
1520
1521
1522 005666 105737 001300          CYCLE:  TSTB   DVACTV      ;ARE ANY DV11'S TO BE TESTED?
1523 005672 001004          BNE     1$          ;BR IF OK.
1524 005674 104402 005174          TYPE   ,MERR2      ;NO DV11'S SELECTED!.
1525 005700 000000          HALT                                ;STOP THE SHOW.
1526 005702 000776          BR     -2          ;DISQUALIFY CONT. SW.
1527 005704 133737 001304 001300 1$:  BITB   RUN,DVACTV  ;IS THIS ONE 'ACTIVE'
1528 005712 001020          BNE     2$          ;BR IF GOOD ONE FOUND.
1529 005714 000241          CLC                                ;CLEAR PROC. CARRY BIT.
1530 005716 106137 001304          ROLB   RUN          ;UPDATE POINTER
1531 005722 105537 001304          ADCB   RUN          ;CATCH CARRY FROM RUN
1532 005726 062737 000024 001306  ADD    #24,CREAM    ;UPDATE ADDRESS POINTER.
1533 005734 022737 001740 001306  CMP    #DV.END,CREAM
1534 005742 001360          BNE     1$          ;KEEP GOING; NOT ALL TESTED FOR.
1535 005744 012737 001500 001306  MOV    #DV.MAP,CREAM ;RESET ADDRESS POINTER.
1536 005752 000754          BR     1$          ;KEEP LOOKING FOR ACTIVE DV11
1537 005754 000241          2$:  CLC                                ;CLEAR PROC. CARRY.
1538 005756 106137 001304          ROLB   RUN          ;UPDATE POINTER.
1539 005762 105537 001304          ADCB   RUN          ;CATCH CARRY.
1540 005766 013700 001306          MOV    CREAM,RO    ;GET ADDRESS POINTER.
1541 005772 062737 000024 001306  ADD    #24,CREAM    ;UPDATE.
1542 006000 022737 001740 001306  CMP    #DV.END,CREAM
1543
1544 006006 001003          BNE     3$          ;ALL DONE?
1545 006010 012737 001500 001306  MOV    #DV.MAP,CREAM ;BR IF NO.
1546 006016 012037 001362          3$:  MOV    (RO)+,DVSCR  ;RESTORE POINTER.
1547 006022 012037 001352          MOV    (RO)+,DVRVEC ;LOAD SYSTEM CTRL. REG
1548 006026 012037 001422          MOV    (RO)+,L00.03 ;LOAD VECTOR
1549 006032 012037 001432          MOV    (RO)+,SYNC2A ;GET LINE PARAMETERS. 00-03
1550 006036 012037 001424          MOV    (RO)+,L04.07 ;
1551 006042 012037 001434          MOV    (RO)+,SYNC2B ;
1552 006046 012037 001426          MOV    (RO)+,L08.11 ;
1553 006052 012037 001436          MOV    (RO)+,SYNC2C ;
1554 006056 012037 001430          MOV    (RO)+,L12.15 ;
1555 006062 012037 001440          MOV    (RO)+,SYNC2D ;
1556 006066 012700 000002          MOV    #2,RO       ;SAVE CORE THIS WAY.
1557 006072 013737 001362 001364  MOV    DVSCR,DVSCRH ;GET SYS CTRL. REG HIGH BYTE.
1558 006100 005237 001364          INC    DVSCRH       ;GOT IT.
1559 006104 013737 001364 001366  MOV    DVSCRH,DVRIC ;GET NXT REC. CHAR REG.
1560 006112 005237 001366          INC    DVRIC        ;GOT IT
1561 006116 013737 001366 001370  MOV    DVRIC,DVLCR  ;GET LN. PAR.REG.
1562 006124 060037 001370          ADD    RO,DVLCR     ;GOT IT
1563 006130 013737 001370 001372  MOV    DVLCR,DVSRS  ;GET SEC. REG. SEL. REG.
1564 006136 060037 001372          ADD    RO,DVSRS     ;GOT IT
1565 006142 013737 001372 001374  MOV    DVSRS,DVSRSH ;GET HIGH BYTE.
1566 006150 005237 001374          INC    DVSRSH       ;GOT IT
1567 006154 013737 001374 001376  MOV    DVSRSH,DVSRA ;SEC. REG. ACCESS.
1568 006162 005237 001376          INC    DVSRA        ;GOT IT

```

```

1569 006166 013737 001376 001400      MOV      DVSRA,DVSFR      ;SPEC. FUN. REG.
1570 006174 060037 001400              ADD      RO,DVSFR        ;
1571 006200 013737 001400 001402      MOV      DVSFR,DVNSR     ;NPR STAT. REG.
1572 006206 060037 001402              ADD      RO,DVNSR       ;
1573 006212 013737 001402 001404      MOV      DVNSR,RESV16    ;RESERVED REG
1574 006220 060037 001404              ADD      RO,RESV16      ;
1575
1576 006224 013737 001352 001354      MOV      DVRVEC,DVRLVL   ;PTY LVL
1577 006232 060037 001354              ADD      RO,DVRLVL      ;
1578 006236 013737 001354 001356      MOV      DVRLVL,DVTVEC   ;TX VEC
1579 006244 060037 001356              ADD      RO,DVTVEC      ;
1580 006250 013737 001356 001360      MOV      DVTVEC,DVTLVL   ;TX LVL
1581 006256 060037 001360              ADD      RO,DVTLVL      ;
1582
1583 006262 012700 001422      MOV      #L00.03,R0      ;LOAD STAU0 00-03
1584 006266 012701 001406      MOV      #MASK.A,R1      ;PREPARE MASK.
1585 006272 012702 001416      MOV      #CLK.A,R2      ;PREPARE CLOCKS
1586 006276 004737 006516      JSR      PC,FIX.00      ;GO AND CALCULATE CONFIGURATION.
1587
1588 006302 012700 001424      MOV      #L04.07,R0      ;LOAD STAU0 00-03
1589 006306 012701 001410      MOV      #MASK.B,R1      ;PREPARE MASK.
1590 006312 012702 001417      MOV      #CLK.B,R2      ;PREPARE CLOCKS
1591 006316 004737 006516      JSR      PC,FIX.00      ;GO AND CALCULATE CONFIGURATION.
1592
1593 006322 012700 001426      MOV      #L08.11,R0      ;LOAD STAU0 00-03
1594 006326 012701 001412      MOV      #MASK.C,R1      ;PREPARE MASK.
1595 006332 012702 001420      MOV      #CLK.C,R2      ;PREPARE CLOCKS
1596 006336 004737 006516      JSR      PC,FIX.00      ;GO AND CALCULATE CONFIGURATION.
1597
1598 006342 012700 001430      MOV      #L12.15,R0      ;LOAD STAU0 00-03
1599 006346 012701 001414      MOV      #MASK.D,R1      ;PREPARE MASK.
1600 006352 012702 001421      MOV      #CLK.D,R2      ;PREPARE CLOCKS
1601 006356 004737 006516      JSR      PC,FIX.00      ;GO AND CALCULATE CONFIGURATION.
1602 006362 032777 000002 172612    BIT      #SW01,@SWR
1603 006370 001445              BEQ      7$
1604 006372
1605 006372 005737 000042      4$:    TST      @#42
1606 006376 001042              BNE      7$
1607 006400 104402 005104      TYPE    ,MCRLF
1608 006404 104403
1609 006406 005366      INSTR
1610 006410 104405      MTSTN
1611 006412 000001      PARAM
1612 006414 001000      1
1613 006416 001226      1000
1614 006420 000          TSTNO
1615 006421 001          0
1616 006422 012700 007260      .BYTE  1
1617 006426 022710      .BYTE  1
1618 006430 012737      5$:    MOV      #TST1,R0
1619 006432 001015      CMP      (PC)+,(R0)
1620 006434 023760 001226 000002    MOV      (PC)+,@(PC)+
1621 006442 001011      BNE      6$
1622 006444 022760 001226 000004    CMP      TSTNO,2(R0)
1623 006452 001005      BNE      6$
1624 006454 010037 001214      MOV      #TSTNO,4(R0)
                                BNE      6$
                                MOV      RO,RETURN

```



```

1625 006460 104402 005104          TYPE      ,MCRLF
1626 006464 000412          BR        8$
1627 006466 005720          6$:      TST      (R0)+
1628 006470 020027 031354      CMP      R0,#TLAST+10
1629 006474 001354          BNE      5$
1630 006476 104402 005100          TYPE      ,MQM
1631 006502 000733          BR        4$
1632 006504 012737 007260 001214 7$:      MOV      #TST1,RETURN ;PREPARE RETURN ADDRESS
1633 006512 000177 172476      8$:      JMP      @RETURN      ;GO START TESTING.
1634
1635 006516 011003          FIX.00: MOV      (R0),R3      ;GET PARAMETERS.
1636 006520 042703 176377      BIC      #^C<1400>,R3 ;CLEAR JUNK.
1637 006524 005703          TST      R3          ;TEST FOR EIGHT BITS.
1638 006526 001005          BNE      1$          ;BR IF NOT 8 BITS.
1639 006530 012711 000400      MOV      #400,(R1)   ;SET FOR 8 BITS PER CHAR
1640 006534 112712 000010      MOVB     #8,(R2)
1641 006540 000424          BR        4$
1642 006542 022703 000400      1$:      CMP      #400,R3     ;CHECK FOR SEVEN BITS.
1643 006546 001005          BNE      2$          ;BR IF NOT 7 BITS.
1644 006550 112711 000200      MOVB     #200,(R1)
1645 006554 112712 000007      MOVB     #7,(R2)
1646 006560 000414          BR        4$
1647 006562 022703 001000      2$:      CMP      #1000,R3   ;CHECK FOR SIX BITS.
1648 006566 001005          BNE      3$          ;BR IF NOT SIX BITS.
1649 006570 112711 000300      MOVB     #300,(R1)
1650 006574 112712 000006      MOVB     #6,(R2)
1651 006600 000404          BR        4$
1652 006602 112711 000340      3$:      MOVB     #340,(R1)  ;IF NONE OF THE ABOVE; MUST BE 5 BITS.
1653 006606 112712 000005      MOVB     #5,(R2)
1654 006612 032710 040000      4$:      BIT      #PARBIT,(R0) ;PARITY ENABLED?
1655 006616 001401          BEQ      5$          ;IF =0; THEN NO PARITY.
1656 006620 105212          INCB     (R2)        ;PLUS ONE TO THE CLOCK!
1657 006622 000207          5$:      RTS      PC
1658
1659          ;*ROUTINE USED TO 'AUTO SIZE' THE DV11
1660          ;*CSR AND VECTOR.
1661          ;*NOTE: THE CSR MAY BE ANY WHERE IN THE FLOATING
1662          ;*      ADDRESS RANGE (175000:175400)
1663          ;*      AND THE VECTOR MAY BE ANY WHERE IN THE
1664          ;*      FLOATING VECTOR RANGE (300:770)
1665          ;*
1666
1667          AUTO.SIZE:
1668          RESET
1669 006624 000005          CSRMAP: MOV      #DV.MAP,R2 ;INSURE A BUS INIT.
1670 006632 005022 001500      1$:      CLR      (R2)+      ;LOAD MAP POINTER.
1671 006634 022702 001740      CMP      #DV.END,R2 ;ZERO ENTIRE MAP
1672 006640 001374          BNE      1$          ;ALL DONE?
1673 006642 105037 001301      CLRB     DVNUM      ;BR IF NO
1674 006646 012702 001500      MOV      #DV.MAP,R2 ;SET OCTAL NUMBER OF DV11'S TO 0
1675 006652 012701 175000      MOV      #175000,R1
1676 006656 012737 007076 000004      MOV      #6$,@#4
1677 006664 005711          2$:      TST      (R1)
1678 006666 001037          BNE      3$          ;SET FOR FIRST ADDRESS TO BE TESTED
1679 006670 022761 177777 000012      CMP      #177777,12(R1) ;SET FOR NON-EXISTANT DEVICE TIME OUT
1680 006676 001033          BNE      3$          ;IF DV11 DVSCR S/B 0
1680          ;IF NO DEV ; TRAP TO 4. IF NO BIT 8 THEN NO DV11
1680          ;IF DV11 THEN DV5FR S/B ALL 1'S ON INIT
1680          ;BR IF NOT DV11

```

```

1681 006700 005761 000016          TST    16(R1)          ;IF DV11 THEN RESV16 S/B ALL 0'S
1682 006704 001030          BNE    3$             ;BR IF NOT DV11
1683          ;AT THIS POINT IT IS ASSUMED THAT R1 HOLDS A DV11 CSR ADDRESS.
1684 006706 010122          MOV    R1,(R2)+      ;STORE CSR IN CORE TABLE.
1685 006710 005722          TST    (R2)+         ;POP OVER VECTOR STORE AREA
1686 006712 052722 000226      BIS    #226,(R2)+    ;SET LINE CARD 1 STAT AND SYNC
1687 006716 052722 000062      BIS    #62,(R2)+    ;
1688 006722 052722 000226      BIS    #226,(R2)+    ;SET LINE CARD 2 STAT AND SYNC
1689 006726 052722 000062      BIS    #62,(R2)+    ;
1690 006732 052722 000226      BIS    #226,(R2)+    ;SET LINE CARD 3 STAT AND SYNC
1691 006736 052722 000062      BIS    #62,(R2)+    ;
1692 006742 052722 000226      BIS    #226,(R2)+    ;SET LINE CARD 4 STAT AND SYNC
1693 006746 052722 000062      BIS    #62,(R2)+    ;
1694 006752 105237 001301      INCB   DVNUM         ;UPDATE DEVICE COUNTER
1695 006756 122737 000010 001301  CMPB   #10,DVNUM     ;ARE MAX. NO. OF DEV FOUND?
1696 006764 001405          BEQ    100$          ;YES DON'T LOOK FOR ANY MORE.
1697 006766 062701 000010      3$:   ADD    #10,R1    ;UPDATE CSR POINTER ADDRESS
1698 006772 022701 175400      CMP    #175400,R1
1699 006776 001332          BNE    2$           ;BR IF MORE ADDRESS TO CHECK.
1700 007000 012722 177777      100$: MOV    #177777,(R2)+ ;TERMINATER.
1701 007004 105037 001300      CLRB  DVACTV
1702 007010 105737 001301      TSTB  DVNUM         ;WERE ANY DV11'S FOUND AT ALL?
1703 007014 001423          BEQ    5$           ;ERROR AUTO SIZER FOUND NO DV11'S IN THIS SYS.
1704 007016 113701 001301      MOVB  DVNUM,R1
1705 007022 110137 001303      MOVB  R1,SAVNUM     ;SAVE NUMBER OF DEVICES
1706 007026 000241      4$:   CLC
1707 007030 106137 001300      ROLB  DVACTV        ;GENERATE ACTIVE REGISTER OF DEVICES.
1708 007034 105237 001300      INCB  DVACTV        ;SET THE BIT
1709 007040 005301          DEC   R1
1710 007042 001371          BNE    4$           ;BR IF MORE TO GENERATE
1711 007044 012737 000006 000004  MOV    #6,@#4       ;RESTORE TRAP VECTOR
1712 007052 113737 001300 001302  MOVB  DVACTV,SAVACT ;SAVE ACTIVE REGISTER
1713 007060 000137 007104          JMP    VECMAP       ;GO FIND THE VECTOR NOW.
1714 007064 104402 005174      5$:   TYPE ,MERR2    ;NOTIFY OPR THAT NO DV11'S FOUND.
1715 007070 005000          CLR   R0           ;MAKE DATA LIGHTS ZERO
1716 007072 000000          HALT
1717 007074 000776          BR    -2           ;DISABLE CONT. SW.
1718 007076 012716 006766      6$:   MOV    #3$,(SP) ;ENTERED BY NON-EXISTANT TIME-OUT.
1719 007102 000002          RTI              ;RETURN TO MAINSTREAM
1720
1721 007104 012737 000340 000022  VECMAP: MOV    #340,@#22 ;SET IOT TRAP PRIO TO 7
1722 007112 012737 007234 000020      MOV    #4$,@#20    ;SET IOT TRAP VECTOR
1723 007120 012702 001500          MOV    #DV.MAP,R2  ;SET SOFTWARE POINTER
1724 007124 012700 000300          MOV    #300,R0     ;FLOATING VECTORS START HERE.
1725 007130 012701 000302          MOV    #302,R1     ;PC OF IOT INSTR.
1726 007134 010120      1$:   MOV    R1,(R0)+    ;START FILLING VECTOR AREA
1727 007136 012721 000004          MOV    #4,(R1)+   ;WITH .+2; IOT
1728 007142 022021          CMP    (R0)+,(R1)+ ;ADD 2 TO R0 +R1
1729 007144 020127 001000      CMP    R1,#1000
1730 007150 101771          BLOS  1$           ;BR IF MORE TO FILL
1731 007152 113737 001300 001246      MOVB  DVACTV,TEMP1 ;STORE TEMPORALLY
1732 007160 006037 001246      2$:   ROR    TEMP1     ;BRING OUT A BIT
1733 007164 103034          BCC   5$           ;BR IF ALL DONE
1734 007166 005037 177776          CLR   PS          ;ZERO CPU PRIO
1735 007172 012772 001300 000000      MOV    #BIT9+BIT7+BIT6,@(R2)
1736 007200 005000          CLR   R0          ;ATTEMPT TO FORCE AN INTERRUPT
    
```

```

1737 007202 005200          INC      R0          ;STALL
1738 007204 001375          BNE     .-2          ;          FOR TIME TO INTERUPT
1739 007206 052762 000300 000002  BIS     #300,2(R2)   ;NO INTERUPT ASSUME 300 AND FIX DV11 LATER
1740 007214 042772 176777 000000 3$:    BIC     #^C<BIT9>,@(R2)
1741 007222 005072 000000          CLR     @(R2)
1742 007226 062702 000024          ADD     #24,R2      ;POP SOFTWARE POINTER
1743 007232 000752          BR      2$          ;KEEP GOING
1744 007234 051662 000002 4$:    BIS     (SP),2(R2)   ;GET VECTOR ADDRESS
1745 007240 042762 000007 000002  BIC     #7,2(R2)    ;CLEAR JUNK
1746 007246 022626          CMP     (SP)+,(SP)+ ;POP IOT JUNK OFF STACK
1747 007250 012716 007214          MOV     #3$, (SP)  ;SET FOR RETURN
1748 007254 000002          RTI
1749 007256 000207 5$:    RTS      PC      ;ALL DONE WITH 'AUTO SIZING'
1750

```

1751 :\*\*\*\*\* TEST 1 \*\*\*\*\*  
1752 :\*VERIFY THAT ADDRESSING DEVICE DOES \*NOT\* CAUSE  
1753 :\*A TIME-OUT TRAP.  
1754 :\*\*\*\*\*

1755 : TEST 1  
1756 :-----  
1757 TST1: MOV #1,TSTNO  
1758 007260 012737 000001 001226 MOV #TST2,NEXT  
1759 007266 012737 007370 001216 MOV #1\$,LOCK  
1760 007274 012737 007326 001220 MOV #8,R0 ;SET FOR MAX. 8 PRI. REGISTERS  
1761 007302 012700 000010 MOV DVSCR,R1 ;GET FIRST PRI. ADDRESS  
1762 007306 013701 001362 MOV #2\$,4 ;SET FOR TIME-OUT TRAP.  
1763 007312 012737 007362 000004 MOV #340,6 ;SAFE GUARD.  
1764 007320 012737 000340 000006 1\$: TST (R1) ;REFERENCE THE ADDRESS.  
1765 007326 005711 NOP ;STALL  
1766 007330 000240 NOP ; FOR TIME.  
1767 007332 000240 SCOP1 ;IF SW09=1; GOTO 1\$  
1768 007334 104401 ADD #2,R1 ;UPDATE TO NEXT ADDRESS.  
1769 007336 062701 000002 DEC R0 ;ARE ALL ADDRESS CHECKED?  
1770 007342 005300 BNE 1\$ ;BR IF NO.  
1771 007344 001370 MOV #6,4 ;RESET TRAP ZONE.  
1772 007346 012737 000006 000004 CLR @#6  
1773 007354 005037 000006  
1774 007360 104400 2\$: SCOPE ;SCOPE THIS TEST  
1775 007362 011602 MOV (SP),R2 ;SAVE THE TRAP PC  
1776 007364 104001 HLT 1 ;REPORT TIME-OUT TRAP  
1777 007366 000002 RTI ;RETURN TO MAIN PROGRAM

1778 :\*\*\*\*\* TEST 2 \*\*\*\*\*  
1779 :\*PRIMARY REGISTER ADDRESSING TEST  
1780 :\*LOAD EACH PRIMARY REGISTER WITH A  
1781 :\*DIFFERENT NUMBER AND VERIFY EACH  
1782 :\*WAS INDIVIDUALLY ADDRESSED.  
1783 :\*\*\*\*\*

1784 : TEST 2  
1785 :-----  
1786 TST2: MOV #2,TSTNO  
1787 007370 012737 000002 001226 MOV #TST3,NEXT  
1788 007376 012737 007622 001216 MOV #1\$,LOCK  
1789 007404 012737 007440 001220 MOV #3\$,R0 ;SET DATA TABLE POINTER  
1790 007412 012700 007602 MOV DVSCR,R3 ;SET DV POINTER  
1791 007416 013703 001362 CLR (R3) ;START REG AT ZERO  
1792 007422 005013 CLR @DVSRS ;ZERO SEC. REG SEL.  
1793 007424 005077 171742 CLR @DVSRA ;ZERO SEC REG ACCESS  
1794 007426 005077 171742 MOV #8,R2 ;SET FOR EIGHT PRIMARY REGISTERS.  
1795 007434 012702 000010 1\$: MOV (R0),R5 ;PUT DATA INTO EXPECTED  
1796 007436 011005 MOV R5,(R3) ;WRITE EXPECTED INTO DV REGISTER  
1797 007440 0110513 MOV (R3),R4 ;READ REGISTER INTO FOUND LOC  
1798 007442 011304 CMP R5,R4 ;DOES EXPECTED=RECEIVED?  
1799 007444 020504 BEQ 64\$ ;BR IF YES  
1800 007446 001401 HLT 3 ;THIS IS A DATA ERROR \*NOT\* A DUEL ADDRESSING ERROR. NOT  
1801 007452 104003 64\$: SCOP1 ;SW09=1?  
1802 007454 104401 CMP (R0)+,(R3)+ ;POP DATA POINTERS AND HRDW POINTER  
1803 007456 022023 CMP R3,DVNSR ;DON'T DO THE DVNSR!  
1804 007458 104401  
1805 007460 020337 001402

```
1807 007464 001002          BNE      4$          ;BR IF NOT DVNSR
1808 007466 022320          CMP      (R3)+,(R0)+ ;POP POINTERS AROUND DVNSR
1809 007470 005302          DEC      R2          ;UPDATE REGISTER COUNTER
1810 007472 020337 001370  4$:      CMP      R3,DVLCR  ;DON'T DO THE DVLCR!
1811 007476 001002          BNE      6$          ;BR IF NOT THE DVLCR
1812 007500 022320          CMP      (R3)+,(R0)+ ;POP POINTERS AROUND THE DVLCR
1813 007502 005302          DEC      R2          ;UPDATE THE REGISTER COUNTER
1814 007504 005302          DEC      R2          ;DITTO
1815 007506 001354          BNE      1$          ;BR IF MORE TO GO
1816                          ;CHECK DUEL ADDRESSING.....
1817 007510 012700 007602    MOV      #3$,R0      ;SET DATA POINTER
1818 007514 013703 001362    MOV      DVSCR,R3    ;SET HRDW POINTER
1819 007520 012737 007532 00'220  MOV      #2$,LOCK    ;SET IF SW09=1
1820 007526 012702 000010    MOV      #8,,R2      ;SET EIGHT PRIMARY REGISTERS
1821 007532 011005          2$:      MOV      (R0),R5     ;LOAD DATA INTO EXPECTED
1822 007534 011304          MOV      (R3),R4     ;READ THE DV REGISTER
1823 007536 020504          CMP      R5,R4      ;DOES THE DATA COMPARE
1824 007540 001401          BEQ      65$        ;BR IF OK
1825 007542 104003          HLT      3          ;NOW THIS WAS A DUEL ADDRESSING ERROR.
1826 007544 104401          65$:     SCOPE1      ;SW09=1?
1827 007546 022023          CMP      (R0)+,(R3)+ ;POP POINTERS
1828 007550 020337 001402    CMP      R3,DVNSR   ;DON'T DO THE DVNSR
1829 007554 001002          BNE      5$          ;BR IF NOT DVNSR
1830 007556 022320          CMP      (R3)+,(R0)+ ;POP POINTERS
1831 007560 005302          DEC      R2          ;SET REG COUNTER
1832 007562 020337 001370  5$:      CMP      R3,DVLCR  ;DON'T DO THE DVLCR
1833 007566 001002          BNE      7$          ;BR IF NOT DVLCR
1834 007570 022320          CMP      (R3)+,(R0)+ ;POP POINTERS
1835 007572 005302          DEC      R2          ;SET REG POINTER
1836 007574 005302          7$:      DEC      R2          ;DITTO
1837 007576 001355          BNE      2$          ;BR IF MORE TO GO
1838 007600 104400          SCOPE          ;SCOPE THIS TEST
1839 007602 000010          3$:      .WORD    000010    ;DVSCR
1840 007604 000000          .WORD    000000    ;DVRIC
1841 007606 000000          .WORD    SKIP      ;DVLCR
1842 007610 001400          .WORD    001400    ;DVSRS
1843 007612 000300          .WORD    000300    ;DVSRA
1844 007614 100000          .WORD    100000    ;DVSFR
1845 007616 000000          .WORD    SKIP      ;DVNSR
1846 007620 000060          .WORD    000060    ;RESV16
```

```
***** TEST 3 *****
; *SYSTEM CONTROL REGISTER READ/WRITE TEST.
; *SET BIT2, VERIFY BIT2 WAS SET.
; *CLEAR BIT2, VERIFY BIT2 WAS CLEARED.
; *****
```

; TEST 3

```
1857 007622 012737 000003 001226  TST3:  MOV      #3,TSTNO
1858 007630 012737 007676 001216    MOV      #TST4,NEXT
1859 007636 013703 001362          MOV      DVSCR,R3    ;SET REGISTER TO BE TESTED.
1860 007642 012705 000004          MOV      #BIT2,R5    ;SET 'EXPECTED'
1861 007646 010513          MOV      R5,(R3)     ;WRITE THE REGISTER.
1862 007650 011304          MOV      (R3),R4     ;READ THE REGISTER.
```

```
1863 007652 020504      CMP      R5,R4      ;R5=GOOD; R4=UNKNOWN.
1864 007654 001401      BEQ      1$         ;ARE THEY THE SAME?
1865 007656 104003      HLT      3          ;COMPARISON ERROR.
1866 007660 040513      1$: BIC      R5,(R3)  ;CLEAR BIT2
1867 007662 011304      MOV      (R3),R4    ;READ THE REGISTER.
1868 007664 005005      CLR      R5         ;SET 'EXPECTED'
1869 007666 020504      CMP      R5,R4      ;R5=GOOD; R4=?
1870 007670 001401      BEQ      2$         ;BR IF OK
1871 007672 104003      HLT      3          ;COMPARISON ERROR
1872 007674 104400      2$: SCOPE          ;SCOPE THIS TEST
```

```
:***** TEST 4 *****
:*SYSTEM CONTROL REGISTER READ/WRITE TEST.
:*SET BIT3, VERIFY BIT3 WAS SET.
:*CLEAR BIT3, VERIFY BIT3 WAS CLEARED.
:*****
```

: TEST 4

```
1882 -----
1883 007676 012737 000004 001226 TST4: MOV      #4,TSTNO
1884 007704 012737 007752 001216      MOV      #TST5,NEXT
1885 007712 013703 001362      MOV      DVSCR,R3      ;SET REGISTER TO BE TESTED.
1886 007716 012705 000010      MOV      #BIT3,R5      ;SET 'EXPECTED'
1887 007722 010513      MOV      R5,(R3)       ;WRITE THE REGISTER.
1888 007724 011304      MOV      (R3),R4       ;READ THE REGISTER.
1889 007726 020504      CMP      R5,R4         ;R5=GOOD; R4=UNKNOWN.
1890 007730 001401      BEQ      1$           ;ARE THEY THE SAME?
1891 007732 104003      HLT      3            ;COMPARISON ERROR.
1892 007734 040513      1$: BIC      R5,(R3)  ;CLEAR BIT3
1893 007736 011304      MOV      (R3),R4       ;READ THE REGISTER.
1894 007740 005005      CLR      R5           ;SET 'EXPECTED'
1895 007742 020504      CMP      R5,R4         ;R5=GOOD; R4=?
1896 007744 001401      BEQ      2$           ;BR IF OK
1897 007746 104003      HLT      3            ;COMPARISON ERROR
1898 007750 104400      2$: SCOPE          ;SCOPE THIS TEST
```

```
:***** TEST 5 *****
:*SYSTEM CONTROL REGISTER READ/WRITE TEST.
:*SET BIT4, VERIFY BIT4 WAS SET.
:*CLEAR BIT4, VERIFY BIT4 WAS CLEARED.
:*****
```

: TEST 5

```
1908 -----
1909 007752 012737 000005 001226 TST5: MOV      #5,TSTNO
1910 007760 012737 010026 001216      MOV      #TST6,NEXT
1911 007766 013703 001362      MOV      DVSCR,R3      ;SET REGISTER TO BE TESTED.
1912 007772 012705 000020      MOV      #BIT4,R5      ;SET 'EXPECTED'
1913 007776 010513      MOV      R5,(R3)       ;WRITE THE REGISTER.
1914 010000 011304      MOV      (R3),R4       ;READ THE REGISTER.
1915 010002 020504      CMP      R5,R4         ;R5=GOOD; R4=UNKNOWN.
1916 010004 001401      BEQ      1$           ;ARE THEY THE SAME?
1917 010006 104003      HLT      3            ;COMPARISON ERROR.
1918 010010 040513      1$: BIC      R5,(R3)  ;CLEAR BIT4
```



```
1919 010012 011304      MOV      (R3),R4      ;READ THE REGISTER.  
1920 010014 005005      CLR      R5           ;SET 'EXPECTED'  
1921 010016 020504      CMP      R5,R4       ;R5=GOOD; R4=?  
1922 010020 001401      BEQ      2$          ;BR IF OK  
1923 010022 104003      HLT      3           ;COMPARISON ERROR  
1924 010024 104400      2$:      SCOPE       ;SCOPE THIS TEST  
1925  
1926  
1927
```

```
:***** TEST 6 *****  
:*SYSTEM CONTROL REGISTER READ/WRITE TEST.  
:*SET BITS, VERIFY BITS WAS SET.  
:*CLEAR BITS, VERIFY BITS WAS CLEARED.  
:*****
```

: TEST 6

```
1934  
1935 010026 012737 000006 001226 TST6:  MOV      #6,TSTNO  
1936 010034 012737 010102 001216      MOV      #TST7,NEXT  
1937 010042 013703 001362      MOV      DVSCR,R3     ;SET REGISTER TO BE TESTED.  
1938 010046 012705 000040      MOV      #BIT5,R5     ;SET 'EXPECTED'  
1939 010052 010513      MOV      R5,(R3)      ;WRITE THE REGISTER.  
1940 010054 011304      MOV      (R3),R4      ;READ THE REGISTER.  
1941 010056 020504      CMP      R5,R4       ;R5=GOOD; R4=UNKNOWN.  
1942 010060 001401      BEQ      1$          ;ARE THEY THE SAME?  
1943 010062 104003      HLT      3           ;COMPARISON ERROR.  
1944 010064 040513      1$:      BIC      R5,(R3)     ;CLEAR BITS  
1945 010066 011304      MOV      (R3),R4      ;READ THE REGISTER.  
1946 010070 005005      CLR      R5           ;SET 'EXPECTED'  
1947 010072 020504      CMP      R5,R4       ;R5=GOOD; R4=?  
1948 010074 001401      BEQ      2$          ;BR IF OK  
1949 010076 104003      HLT      3           ;COMPARISON ERROR  
1950 010100 104400      2$:      SCOPE       ;SCOPE THIS TEST  
1951  
1952
```

```
:***** TEST 7 *****  
:*SYSTEM CONTROL REGISTER READ/WRITE TEST.  
:*SET BIT6, VERIFY BIT6 WAS SET.  
:*CLEAR BIT6, VERIFY BIT6 WAS CLEARED.  
:*****
```

: TEST 7

```
1953  
1954  
1955  
1956  
1957  
1958  
1959  
1960  
1961 010102 012737 000007 001226 TST7:  MOV      #7,TSTNO  
1962 010110 012737 010156 001216      MOV      #TST10,NEXT  
1963 010116 013703 001362      MOV      DVSCR,R3     ;SET REGISTER TO BE TESTED.  
1964 010122 012705 000100      MOV      #BIT6,R5     ;SET 'EXPECTED'  
1965 010126 010513      MOV      R5,(R3)      ;WRITE THE REGISTER.  
1966 010130 011304      MOV      (R3),R4      ;READ THE REGISTER.  
1967 010132 020504      CMP      R5,R4       ;R5=GOOD; R4=UNKNOWN.  
1968 010134 001401      BEQ      1$          ;ARE THEY THE SAME?  
1969 010136 104003      HLT      3           ;COMPARISON ERROR.  
1970 010140 040513      1$:      BIC      R5,(R3)     ;CLEAR BIT6  
1971 010142 011304      MOV      (R3),R4      ;READ THE REGISTER.  
1972 010144 005005      CLR      R5           ;SET 'EXPECTED'  
1973 010146 020504      CMP      R5,R4       ;R5=GOOD; R4 ?  
1974 010150 001401      BEQ      2$          ;BR IF OK
```

1975 010152 104003 HLT 3 :COMPARISON ERROR  
1976 010154 104400 2\$: SCOPE :SCOPE THIS TEST  
1977  
1978  
1979

:\*\*\*\*\* TEST 10 \*\*\*\*\*  
:\*TEST THAT BIT 7(RECEIVER INTERRUPT)  
:\*CANNOT BE WRITTEN WHEN BIT 9 (SYSTEM MAINTAINCE)  
:\*IS NOT SET.  
:\*THEN VERIFY THAT BIT 7 CAN BE WRITTEN WHEN  
:\*BIT 9 IS SET.  
:\*\*\*\*\*

: TEST 10  
-----

1989 010156 012737 000010 001226 TST10: MOV #10,TSTNO  
1990 010164 012737 010252 001216 MOV #TST11,NEXT  
1991 010172 013703 001362 MOV DVSCR,R3 :SET REGISTER POINTER INTO R3  
1992 010176 005005 CLR R5 :SET EXPECTED RESULT  
1993 010200 012713 000200 MOV #BIT7,(R3) :ATTEMPT TO SET BIT 7  
1994 010204 011304 MOV (R3),R4 :READ BACK REGISTER  
1995 010206 001401 BEQ 1\$ :BIT 7 SHOULD NOT BE SET  
1996 010210 104003 HLT 3 :DVSCR NOT ALL 0'S  
1997 010212 012705 001200 1\$: MOV #BIT7+BIT9,R5 :SET BIT 7+BIT9 IN THE DVSCR  
1998 010216 010513 MOV R5,(R3) :  
1999 010220 011304 MOV (R3),R4 :READ REGISTER  
2000 010222 020504 CMP R5,R4 :IS BIT 9 AND BIT 7 SET?  
2001 010224 001401 BEQ 2\$ :BR IF OK  
2002 010226 104003 HLT 3 :DVSCR ERROR  
2003 010230 042705 000200 2\$: BIC #BIT7,R5 :CLEAR BIT 7  
2004 010234 042713 000200 BIC #BIT7,(R3) :DITTO  
2005 010240 011304 MOV (R3),R4 :READ REGISTER  
2006 010242 020504 CMP R5,R4 :COMPARE OK?  
2007 010244 001401 BEQ 3\$ :BR IF YES  
2008 010246 104003 HLT 3 :DVSCR ERROR  
2009 010250 104400 3\$: SCOPE :SCOPE THIS TEST  
2010  
2011

:\*\*\*\*\* TEST 11 \*\*\*\*\*  
:\*SYSTEM CONTROL REGISTER READ/WRITE TEST.  
:\*SET BIT8, VERIFY BIT8 WAS SET.  
:\*CLEAR BIT8, VERIFY BIT8 WAS CLEARED.  
:\*\*\*\*\*

: TEST 11  
-----

2020 010252 012737 000011 001226 TST11: MOV #11,TSTNO  
2021 010260 012737 010326 001216 MOV #TST12,NEXT  
2022 010266 013703 001362 MOV DVSCR,R3 :SET REGISTER TO BE TESTED.  
2023 010272 012705 000400 MOV #BIT8,R5 :SET 'EXPECTED'  
2024 010276 010513 MOV R5,(R3) :WRITE THE REGISTER.  
2025 010300 011304 MOV (R3),R4 :READ THE REGISTER.  
2026 010302 020504 CMP R5,R4 :R5=GOOD; R4=UNKNOWN.  
2027 010304 001401 BEQ 1\$ :ARE THEY THE SAME?  
2028 010306 104003 HLT 3 :COMPARISON ERROR.  
2029 010310 040513 1\$: BIC R5,(R3) :CLEAR BIT8  
2030 010312 011304 MOV (R3),R4 :READ THE REGISTER.

```
2031 010314 005005 CLR R5 ;SET 'EXPECTED'  
2032 010316 020504 CMP R5,R4 ;R5=GOOD; R4=?  
2033 010320 001401 BEQ 2$ ;BR IF OK  
2034 010322 104003 HLT 3 ;COMPARISON ERROR  
2035 010324 104400 2$: SCOPE ;SCOPE THIS TEST  
2036  
2037  
2038
```

```
:***** TEST 12 *****  
:*SYSTEM CONTROL REGISTER READ/WRITE TEST.  
:*SET BIT9, VERIFY BIT9 WAS SET.  
:*CLEAR BIT9, VERIFY BIT9 WAS CLEARED.  
:*****
```

: TEST 12

```
2046 010326 012737 000012 001226 TST12: MOV #12,TSTNO  
2047 010334 012737 010402 001216 MOV #TST13,NEXT  
2048 010342 013703 001362 MOV DVSCR,R3 ;SET REGISTER TO BE TESTED.  
2049 010346 012705 001000 MOV #BIT9,R5 ;SET 'EXPECTED'  
2050 010352 010513 MOV R5,(R3) ;WRITE THE REGISTER.  
2051 010354 011304 MOV (R3),R4 ;READ THE REGISTER.  
2052 010356 020504 CMP R5,R4 ;R5=GOOD; R4=UNKNOWN.  
2053 010360 001401 BEQ 1$ ;ARE THEY THE SAME?  
2054 010362 104003 HLT 3 ;COMPARISON ERROR.  
2055 010364 040513 1$: BIC R5,(R3) ;CLEAR BIT9  
2056 010366 011304 MOV (R3),R4 ;READ THE REGISTER.  
2057 010370 005005 CLR R5 ;SET 'EXPECTED'  
2058 010372 020504 CMP R5,R4 ;R5=GOOD; R4=?  
2059 010374 001401 BEQ 2$ ;BR IF OK  
2060 010376 104003 HLT 3 ;COMPARISON ERROR  
2061 010400 104400 2$: SCOPE ;SCOPE THIS TEST  
2062  
2063
```

```
:***** TEST 13 *****  
:*SYSTEM CONTROL REGISTER READ/WRITE TEST.  
:*SET BIT10, VERIFY BIT10 WAS SET.  
:*CLEAR BIT10, VERIFY BIT10 WAS CLEARED.  
:*****
```

: TEST 13

```
2072 010402 012737 000013 001226 TST13: MOV #13,TSTNO  
2073 010410 012737 010456 001216 MOV #TST14,NEXT  
2074 010416 013703 001362 MOV DVSCR,R3 ;SET REGISTER TO BE TESTED.  
2075 010422 012705 002000 MOV #BIT10,R5 ;SET 'EXPECTED'  
2076 010426 010513 MOV R5,(R3) ;WRITE THE REGISTER.  
2077 010430 011304 MOV (R3),R4 ;READ THE REGISTER.  
2078 010432 020504 CMP R5,R4 ;R5=GOOD; R4=UNKNOWN.  
2079 010434 001401 BEQ 1$ ;ARE THEY THE SAME?  
2080 010436 104003 HLT 3 ;COMPARISON ERROR.  
2081 010440 040513 1$: BIC R5,(R3) ;CLEAR BIT10  
2082 010442 011304 MOV (R3),R4 ;READ THE REGISTER.  
2083 010444 005005 CLR R5 ;SET 'EXPECTED'  
2084 010446 020504 CMP R5,R4 ;R5=GOOD; R4=?  
2085 010450 001401 BEQ 2$ ;BR IF OK  
2086 010452 104003 HLT 3 ;COMPARISON ERROR
```

2087 010454 104400 2\$: SCOPE ;SCOPE THIS TEST

2088  
2089  
2090 :\*\*\*\*\* TEST 14 \*\*\*\*\*  
2091 :\*SYSTEM CONTROL REGISTER READ/WRITE TEST.  
2092 :\*SET BIT12, VERIFY BIT12 WAS SET.  
2093 :\*CLEAR BIT12, VERIFY BIT12 WAS CLEARED.  
2094 :\*\*\*\*\*

2095  
2096 : TEST 14

2097  
2098 010456 012737 000014 001226 TST14: MOV #14,TSTNO  
2099 010464 012737 010532 001216 MOV #TST15,NEXT  
2100 010472 013703 001362 MOV DVSCR,R3 ;SET REGISTER TO BE TESTED.  
2101 010476 012705 010000 MOV #BIT12,R5 ;SET 'EXPECTED'  
2102 010502 010513 MOV R5,(R3) ;WRITE THE REGISTER.  
2103 010504 011304 MOV (R3),R4 ;READ THE REGISTER.  
2104 010506 020504 CMP R5,R4 ;R5=GOOD; R4=UNKNOWN.  
2105 010510 001401 BEQ 1\$ ;ARE THEY THE SAME?  
2106 010512 104003 HLT 3 ;COMPARISON ERROR.  
2107 010514 040513 1\$: BIC R5,(R3) ;CLEAR BIT12  
2108 010516 011304 MOV (R3),R4 ;READ THE REGISTER.  
2109 010520 005005 CLR R5 ;SET 'EXPECTED'  
2110 010522 020504 CMP R5,R4 ;R5 GOOD; R4=?  
2111 010524 001401 BEQ 2\$ ;BR IF OK  
2112 010526 104003 HLT 3 ;COMPARISON ERROR  
2113 010530 104400 2\$: SCOPE ;SCOPE THIS TEST

2114  
2115  
2116 :\*\*\*\*\* TEST 15 \*\*\*\*\*  
2117 :\*SYSTEM CONTROL REGISTER READ/WRITE TEST.  
2118 :\*SET BIT13, VERIFY BIT13 WAS SET.  
2119 :\*CLEAR BIT13, VERIFY BIT13 WAS CLEARED.  
2120 :\*\*\*\*\*

2121  
2122 : TEST 15

2123  
2124 010532 012737 000015 001226 TST15: MOV #15,TSTNO  
2125 010540 012737 010606 001216 MOV #TST16,NEXT  
2126 010546 013703 001362 MOV DVSCR,R3 ;SET REGISTER TO BE TESTED.  
2127 010552 012705 020000 MOV #BIT13,R5 ;SET 'EXPECTED'  
2128 010556 010513 MOV R5,(R3) ;WRITE THE REGISTER.  
2129 010560 011304 MOV (R3),R4 ;READ THE REGISTER.  
2130 010562 020504 CMP R5,R4 ;R5=GOOD; R4=UNKNOWN.  
2131 010564 001401 BEQ 1\$ ;ARE THEY THE SAME?  
2132 010566 104003 HLT 3 ;COMPARISON ERROR.  
2133 010570 040513 1\$: BIC R5,(R3) ;CLEAR BIT13  
2134 010572 011304 MOV (R3),R4 ;READ THE REGISTER.  
2135 010574 005005 CLR R5 ;SET 'EXPECTED'  
2136 010576 020504 CMP R5,R4 ;R5=GOOD; R4=?  
2137 010600 001401 BEQ 2\$ ;BR IF OK  
2138 010602 104003 HLT 3 ;COMPARISON ERROR  
2139 010604 104400 2\$: SCOPE ;SCOPE THIS TEST

2140  
2141  
2142 :\*\*\*\*\* TEST 16 \*\*\*\*\*

2143 ;\*SYSTEM CONTROL REGISTER READ/WRITE TEST.  
2144 ;\*SET BIT3+BIT0, VERIFY BIT3+BIT0 WAS SET.  
2145 ;\*CLEAR BIT3+BIT0, VERIFY BIT3+BIT0 WAS CLEARED.  
2146 ;\*\*\*\*\*  
2147

; TEST 16

2150	010606	012737	000016	001226	TST16: MOV #16,TSTNO	
2151	010614	012737	010662	001216	MOV #TST17,NEXT	
2152	010622	013703	001362		MOV DVSCR,R3	;SET REGISTER TO BE TESTED.
2153	010626	012705	000011		MOV #BIT3+BIT0,R5	;SET 'EXPECTED''
2154	010632	010513			MOV R5,(R3)	;WRITE THE REGISTER.
2155	010634	011304			MOV (R3),R4	;READ THE REGISTER.
2156	010636	020504			CMP R5,R4	;R5=GOOD; R4=UNKNOWN.
2157	010640	001401			BEQ 1\$	;ARE THEY THE SAME?
2158	010642	104003			HLT 3	;COMPARISON ERROR.
2159	010644	040513		1\$:	BIC R5,(R3)	;CLEAR BIT3+BIT0
2160	010646	011304			MOV (R3),R4	;READ THE REGISTER.
2161	010650	005005			CLR R5	;SET 'EXPECTED''
2162	010652	020504			CMP R5,R4	;R5=GOOD; R4=?
2163	010654	001401			BEQ 2\$	;BR IF OK
2164	010656	104003			HLT 3	;COMPARISON ERROR
2165	010660	104400		2\$:	SCOPE	;SCOPE THIS TEST

;\*\*\*\*\* TEST 17 \*\*\*\*\*  
;\*TEST THAT BIT 15(NPR STATUS OVERFLOW INTERUPT)  
;\*CANNOT BE WRITTEN WHEN BIT 9 (SYSTEM MAINTAINCE`  
;\*IS NOT SET.  
;\*THEN VERIFY THAT BIT 15 CAN BE WRITTEN WHEN  
;\*BIT 9 IS SET.  
;\*\*\*\*\*

; TEST 17

2178	010662	012737	000017	001226	TST17: MOV #17,TSTNO	
2179	010670	012737	010756	001216	MOV #TST20,NEXT	
2180	010676	013703	001362		MOV DVSCR,R3	;SET REGISTER POINTER INTO R3
2181	010702	005005			CLR R5	;SET EXPECTED RESULT
2182	010704	012713	100000		MOV #BIT15,(R3)	;ATTEMPT TO SET BIT 15
2183	010710	011304			MOV (R3),R4	;READ BACK REGISTER
2184	010712	001401			BEQ 1\$	;BIT 15 SHOULD NOT BE SET
2185	010714	104003			HLT 3	;DVSCR NOT ALL 0'S
2186	010716	012705	101000	1\$:	MOV #BIT15+BIT9,R5	;SET BIT 15+BIT9 IN THE DVSCR
2187	010722	010513			MOV R5,(R3)	
2188	010724	011304			MOV (R3),R4	;READ REGISTER
2189	010726	020504			CMP R5,R4	;IS BIT 9 AND BIT 15 SET?
2190	010730	001401			BEQ 2\$	;BR IF OK
2191	010732	104003			HLT 3	;DVSCR ERROR
2192	010734	042705	100000	2\$:	BIC #BIT15,R5	;CLEAR BIT 15
2193	010740	042713	100000		BIC #BIT15,(R3)	;DITTO
2194	010744	011304			MOV (R3),R4	;READ REGISTER
2195	010746	020504			CMP R5,R4	;COMPARE OK?
2196	010750	001401			BEQ 3\$	;BR IF YES
2197	010752	104003			HLT 3	;DVSCR ERROR
2198	010754	104400		3\$:	SCOPE	;SCOPE THIS TEST

2199  
2200 :\*\*\*\*\* TEST 20 \*\*\*\*\*  
2201 :\*TEST THAT BIT8 IN DVSCR CLEARS  
2202 :\*BIT7 OF DVSCR.  
2203 :\*\*\*\*\*

2204  
2205 : TEST 20  
2206 :-----  
2207 010756 012737 000020 001226 TST20: MOV #20,TSTNO  
2208 010764 012737 011034 001216 MOV #TST21,NEXT  
2209 010772 013703 001362 MOV DVSCR,R3 ;GET POINTER  
2210 010776 012705 000400 MOV #BIT8,R5 ;GET GOOD  
2211 011002 012713 000200 MOV #BIT7,(R3) ;SET BIT7  
2212 011006 052713 000400 BIS #BIT8,(R3) ;SET BIT8  
2213 011012 027777 170350 170346 CMP @DVRIC,@DVRIC ;WASTE TIME  
2214 011020 011304 MOV (R3),R4 ;READ SCR  
2215 011022 020504 CMP R5,R4 ;BIT8 ONLY SET?  
2216 011024 001401 BEQ 1\$ ;BR IF YES  
2217 011026 104003 HLT 3 ;BIT8 NOT ONLY BIT SET OR 8 ISN'T SET.  
2218 011030 005013 1\$: CLR (R3) ;LEAVE REG=0  
2219 011032 104400 SCOPE ;SCOPE

2220  
2221 :\*\*\*\*\* TEST 21 \*\*\*\*\*  
2222 :\*RECEIVER INTERRUPT CHARACTER REGISTER TEST  
2223 :\*TEST THAT RECEIVER INTERRUPT CHARACTER REGISTER CANNOT BE WRITTEN.  
2224 :\*WRITE RECEIVER INTERRUPT CHARACTER REGISTER WITH ALL 1'S  
2225 :\*AND VERIFY THAT ALL 0'S ARE READ BACK.  
2226 :\*\*\*\*\*

2227 : TEST 21  
2228 :-----  
2229  
2230 TST21: MOV #21,TSTNO  
2231 011034 012737 000021 001226 MOV #TST22,NEXT  
2232 011042 012737 011074 001216 MOV DVRIC,R3  
2233 011050 013703 001366  
2234  
2235 011054 005005 CLR R5 ;GET THE REGISTER.  
2236 011056 012713 177777 MOV #-1,(R3) ;SET EXPECTED (ZERO)  
2237 011062 011304 MOV (R3),R4 ;WRITE REGISTER WITH ALL 1'S  
2238 011064 020504 CMP R5,R4 ;READ THE REGISTER.  
2239 011066 001401 BEQ 1\$ ;IS THE REGISTER EQUAL TO ZERO.  
2240 011070 104003 HLT 3 ;BR IF OK.  
2241 011072 104400 1\$: SCOPE ;REGISTER NOT ZERO.  
;SCOPE THIS TEST.

2242  
2243 :\*\*\*\*\* TEST 22 \*\*\*\*\*  
2244 :\*LINE CONTROL REGISTER READ/WRITE TEST.  
2245 :\*SET BIT9, VERIFY BIT9 WAS SET.  
2246 :\*CLEAR BIT9, VERIFY BIT9 WAS CLEARED.  
2247 :\*\*\*\*\*

2248 : TEST 22  
2249 :-----  
2250  
2251 TST22: MOV #22,TSTNO  
2252 011074 012737 000022 001226 MOV #TST23,NEXT  
2253 011102 012737 011160 001216 MOV DVLCR,R3 ;SET REGISTER TO BE TESTED.  
2254 011110 013703 001370

```
2255 011114 012705 001000      MOV      #BIT9,R5      ;SET 'EXPECTED'.
2256 011120 010513              MOV      R5,(R3)      ;WRITE THE REGISTER.
2257 011122 011304              MOV      (R3),R4      ;READ THE REGISTER.
2258 011124 042704 000063      BIC      #BIT5+BIT4+BIT1+BIT0,R4 ;CLEAR UNWANTED BITS
2259 011130 020504              CMP      R5,R4        ;R5=GOOD; R4=UNKNOWN.
2260 011132 001401              BEQ      1$           ;ARE THEY THE SAME?
2261 011134 104003              HLT      3            ;COMPARISON ERROR.
2262 011136 040513 1$:      BIC      R5,(R3)      ;CLEAR BIT9
2263 011140 011304              MOV      (R3),R4      ;READ THE REGISTER.
2264 011142 042704 000063      BIC      #BIT5+BIT4+BIT1+BIT0,R4 ;CLEAR UNWANTED BITS
2265 011146 005005              CLR      R5           ;SET 'EXPECTED'
2266 011150 020504              CMP      R5,R4        ;R5=GOOD; R4=?
2267 011152 001401              BEQ      2$           ;BR IF OK
2268 011154 104003              HLT      3            ;COMPARISON ERROR
2269 011156 104400 2$:      SCOPE          ;SCOPE THIS TEST
```

```
2270
2271
2272 ;***** TEST 23 *****
2273 ;*LINE CONTROL REGISTER READ/WRITE TEST.
2274 ;*SET BIT10, VERIFY BIT10 WAS SET.
2275 ;*CLEAR BIT10, VERIFY BIT10 WAS CLEARED.
2276 ;*****
```

```
2277
2278 ; TEST 23
2279 ;-----
2280 011160 012737 000023 001226 TST23: MOV      #23,TSTNO
2281 011166 012737 011244 001216      MOV      #TST24,NEXT
2282 011174 013703 001370              MOV      DVLCR,R3    ;SET REGISTER TO BE TESTED.
2283 011200 012705 002000              MOV      #BIT10,R5   ;SET 'EXPECTED'.
2284 011204 010513              MOV      R5,(R3)     ;WRITE THE REGISTER.
2285 011206 011304              MOV      (R3),R4     ;READ THE REGISTER.
2286 011210 042704 000063      BIC      #BIT5+BIT4+BIT1+BIT0,R4 ;CLEAR UNWANTED BITS
2287 011214 020504              CMP      R5,R4        ;R5=GOOD; R4=UNKNOWN.
2288 011216 001401              BEQ      1$           ;ARE THEY THE SAME?
2289 011220 104003              HLT      3            ;COMPARISON ERROR.
2290 011222 040513 1$:      BIC      R5,(R3)      ;CLEAR BIT10
2291 011224 011304              MOV      (R3),R4      ;READ THE REGISTER.
2292 011226 042704 000063      BIC      #BIT5+BIT4+BIT1+BIT0,R4 ;CLEAR UNWANTED BITS
2293 011232 005005              CLR      R5           ;SET 'EXPECTED'
2294 011234 020504              CMP      R5,R4        ;R5=GOOD; R4=?
2295 011236 001401              BEQ      2$           ;BR IF OK
2296 011240 104003              HLT      3            ;COMPARISON ERROR
2297 011242 104400 2$:      SCOPE          ;SCOPE THIS TEST
```

```
2298
2299 ;***** TEST 24 *****
2300 ;*LINE CONTROL REGISTER READ/WRITE TEST.
2301 ;*SET BIT11, VERIFY BIT11 WAS SET.
2302 ;*CLEAR BIT11, VERIFY BIT11 WAS CLEARED.
2303 ;*****
```

```
2304
2305 ; TEST 24
2306 ;-----
2307
2308 011244 012737 000024 001226 TST24: MOV      #24,TSTNO
2309 011252 012737 011330 001216      MOV      #TST25,NEXT
2310 011260 013703 001370              MOV      DVLCR,R3    ;SET REGISTER TO BE TESTED.
```

```
2311 011264 012705 004000      MOV      #BIT11,R5      ;SET 'EXPECTED '  
2312 011270 010513      MOV      R5,(R3)      ;WRITE THE REGISTER.  
2313 011272 011304      MOV      (R3),R4      ;READ THE REGISTER.  
2314 011274 042704 000063      BIC      #BIT5+BIT4+BIT1+BIT0,R4 ;CLEAR UNWANTED BITS  
2315 011300 020504      CMP      R5,R4      ;R5=GOOD; R4=UNKNOWN.  
2316 011302 001401      BEQ      1$      ;ARE THEY THE SAME?  
2317 011304 104003      HLT      3      ;COMPARISON ERROR.  
2318 011306 040513      1$:      BIC      R5,(R3)      ;CLEAR BIT11  
2319 011310 011304      MOV      (R3),R4      ;READ THE REGISTER.  
2320 011312 042704 000063      BIC      #BIT5+BIT4+BIT1+BIT0,R4 ;CLEAR UNWANTED BITS  
2321 011316 005005      CLR      R5      ;SET 'EXPECTED'  
2322 011320 020504      CMP      R5,R4      ;R5=GOOD; R4=?  
2323 011322 001401      BEQ      2$      ;BR IF OK  
2324 011324 104003      HLT      3      ;COMPARISON ERROR  
2325 011326 104400      2$:      SCOPE      ;SCOPE THIS TEST
```

```
2326  
2327  
2328  
2329  
2330  
2331  
2332  
2333  
2334  
2335  
2336  
2337  
2338  
2339  
2340  
2341  
2342  
2343  
2344  
2345  
2346  
2347  
2348  
2349  
2350  
2351  
2352  
2353  
2354  
2355  
2356  
2357  
2358  
2359  
2360  
2361  
2362  
2363
```

\*\*\*\*\* TEST 25 \*\*\*\*\*  
\*LINE CONTRCL REGISTER READ/WRITE TEST.  
\*SET BIT12, VERIFY BIT12 WAS SET.  
\*CLEAR BIT12, VERIFY BIT12 WAS CLEARED.  
\*\*\*\*\*

: TEST 25

```
2336 011330 012737 000025 001226 TST25: MOV      #25,TSTNO  
2337 011336 012737 011414 001216      MOV      #TST26,NEXT  
2338 011344 013703 001370      MOV      DVLCR,R3      ;SET REGISTER TO BE TESTED.  
2339 011350 012705 010000      MOV      #BIT12,R5      ;SET 'EXPECTED '  
2340 011354 010513      MOV      R5,(R3)      ;WRITE THE REGISTER.  
2341 011356 011304      MOV      (R3),R4      ;READ THE REGISTER.  
2342 011360 042704 000063      BIC      #BIT5+BIT4+BIT1+BIT0,R4 ;CLEAR UNWANTED BI'S  
2343 011364 020504      CMP      R5,R4      ;R5=GOOD; R4=UNKNOWN.  
2344 011366 001401      BEQ      1$      ;ARE THEY THE SAME?  
2345 011370 104003      HLT      3      ;COMPARISON ERROR.  
2346 011372 040513      1$:      BIC      R5,(R3)      ;CLEAR BIT12  
2347 011374 011304      MOV      (R3),R4      ;READ THE REGISTER.  
2348 011376 042704 000063      BIC      #BIT5+BIT4+BIT1+BIT0,R4 ;CLEAR UNWANTED BITS  
2349 011402 005005      CLR      R5      ;SET 'EXPECTED'  
2350 011404 020504      CMP      R5,R4      ;R5=GOOD; R4=?  
2351 011406 001401      BEQ      2$      ;BR IF OK  
2352 011410 104003      HLT      3      ;COMPARISON ERROR  
2353 011412 104400      2$:      SCOPE      ;SCOPE THIS TEST
```

: TEST 26

```
2354  
2355  
2356  
2357  
2358  
2359  
2360  
2361  
2362  
2363  
2364 011414 012737 000026 001226 TST26: MOV      #26,TSTNO  
2365 011422 012737 011500 001216      MOV      #TST27,NEXT  
2366 011430 013703 001370      MOV      DVLCR,R3      ;SET REGISTER TO BE TESTED.
```

\*\*\*\*\* TEST 26 \*\*\*\*\*  
\*LINE CONTROL REGISTER READ/WRITE TEST.  
\*SET BIT13, VERIFY BIT13 WAS SET.  
\*CLEAR BIT13, VERIFY BIT13 WAS CLEARED.  
\*\*\*\*\*



```
2367 011434 012705 020000      MOV      #BIT13,R5      ;SET 'EXPECTED '  
2368 011440 010513      MOV      R5,(R3)       ;WRITE THE REGISTER.  
2369 011442 011304      MOV      (R3),R4       ;READ THE REGISTER.  
2370 011444 042704 000063      BIC      #BIT5+BIT4+BIT1+BIT0,R4 ;CLEAR UNWANTED BITS  
2371 011450 020504      CMP      R5,R4        ;R5=GOOD; R4=UNKNOWN.  
2372 011452 001401      BEQ      1$           ;ARE THEY THE SAME?  
2373 011454 104003      HLT      3            ;COMPARISON ERROR.  
2374 011456 040513      1$:      BIC      R5,(R3)       ;CLEAR BIT13  
2375 011460 011304      MOV      (R3),R4       ;READ THE REGISTER.  
2376 011462 042704 000063      BIC      #BIT5+BIT4+BIT1+BIT0,R4 ;CLEAR UNWANTED BITS  
2377 011466 005005      CLR      R5           ;SET 'EXPECTED'  
2378 011470 020504      CMP      R5,R4        ;R5=GOOD; R4=?  
2379 011472 001401      BEQ      2$           ;BR IF OK  
2380 011474 104003      HLT      3            ;COMPARISON ERROR  
2381 011476 104400      2$:      SCOPE          ;SCOPE THIS TEST
```

```
2382  
2383  
2384  
2385  
2386  
2387  
2388  
2389  
2390  
2391  
2392  
2393  
2394  
2395  
2396  
2397  
2398  
2399  
2400  
2401  
2402  
2403  
2404  
2405  
2406  
2407  
2408  
2409  
2410  
2411  
2412  
2413  
2414  
2415  
2416  
2417  
2418  
2419  
2420  
2421  
2422
```

```
***** TEST 27 *****  
*LINE CONTROL REGISTER READ/WRITE TEST.  
*SET BIT14, VERIFY BIT14 WAS SET.  
*CLEAR BIT14, VERIFY BIT14 WAS CLEARED.  
*****
```

: TEST 27

```
-----  
TST27: MOV      #27,TSTNO  
MOV      #TST30,NEXT  
MOV      DVLCR,R3      ;SET REGISTER TO BE TESTED.  
MOV      #BIT14,R5     ;SET 'EXPECTED '  
MOV      R5,(R3)       ;WRITE THE REGISTER.  
MOV      (R3),R4       ;READ THE REGISTER.  
BIC      #BIT5+BIT4+BIT1+BIT0,R4 ;CLEAR UNWANTED BITS  
CMP      R5,R4        ;R5=GOOD; R4=UNKNOWN.  
BEQ      1$           ;ARE THEY THE SAME?  
HLT      3            ;COMPARISON ERROR.  
1$:      BIC      R5,(R3)       ;CLEAR BIT14  
MOV      (R3),R4       ;READ THE REGISTER.  
BIC      #BIT5+BIT4+BIT1+BIT0,R4 ;CLEAR UNWANTED BITS  
CLR      R5           ;SET 'EXPECTED'  
CMP      R5,R4        ;R5=GOOD; R4=?  
BEQ      ?            ;BR IF OK  
HLT      3            ;COMPARISON ERROR  
2$:      SCOPE          ;SCOPE THIS TEST
```

```
***** TEST 30 *****  
*SECONDARY REGISTER SELECTOR READ/WRITE TEST.  
*SET BIT0, VERIFY BIT0 WAS SET.  
*CLEAR BIT0, VERIFY BIT0 WAS CLEARED.  
*****
```

: TEST 30

```
-----  
TST30: MOV      #30,TSTNO  
MOV      #TST31,NEXT  
MOV      DVSRS,R3      ;SET REGISTER TO BE TESTED.
```

```
2423 011604 012705 000001      MOV      #BIT0,R5      ;SET 'EXPECTED '  
2424 011610 010513      MOV      R5,(R3)      ;WRITE THE REGISTER.  
2425 011612 011304      MOV      (R3),R4      ;READ THE REGISTER.  
2426 011614 020504      CMP      R5,R4        ;R5=GOOD; R4=UNKNOWN.  
2427 011616 001401      BEQ      1$           ;ARE THEY THE SAME?  
2428 011620 104003      HLT      3            ;COMPARISON ERROR.  
2429 011622 040513      1$:      BIC      R5,(R3)      ;CLEAR BIT0  
2430 011624 011304      MOV      (R3),R4      ;READ THE REGISTER.  
2431 011626 005005      CLR      R5           ;SET 'EXPECTED'  
2432 011630 020504      CMP      R5,R4        ;R5=GOOD; R4=?  
2433 011632 001401      BEQ      2$           ;BR IF OK  
2434 011634 104003      HLT      3            ;COMPARISON ERROR  
2435 011636 104400      2$:      SCOPE          ;SCOPE THIS TEST  
2436  
2437  
2438
```

```
:***** TEST 31 *****  
:*SECONDARY REGISTER SELECTOR READ/WRITE TEST.  
:*SET BIT1, VERIFY BIT1 WAS SET.  
:*CLEAR BIT1, VERIFY BIT1 WAS CLEARED.  
:*****
```

: TEST 31

```
2445  
2446 011640 012737 000031 001226 TST31: MOV      #31,TSTNO  
2447 011646 012737 011714 001216      MOV      #TST32,NEXT  
2448 011654 013703 001372      MOV      DVSRS,R3      ;SET REGISTER TO BE TESTED.  
2449 011660 012705 000002      MOV      #BIT1,R5      ;SET 'EXPECTED '  
2450 011664 010513      MOV      R5,(R3)      ;WRITE THE REGISTER.  
2451 011666 011304      MOV      (R3),R4      ;READ THE REGISTER.  
2452 011670 020504      CMP      R5,R4        ;R5=GOOD; R4=UNKNOWN.  
2453 011672 001401      BEQ      1$           ;ARE THEY THE SAME?  
2454 011674 104003      HLT      3            ;COMPARISON ERROR.  
2455 011676 040513      1$:      BIC      R5,(R3)      ;CLEAR BIT1  
2456 011700 011304      MOV      (R3),R4      ;READ THE REGISTER.  
2457 011702 005005      CLR      R5           ;SET 'EXPECTED'  
2458 011704 020504      CMP      R5,R4        ;R5=GOOD; R4=?  
2459 011706 001401      BEQ      2$           ;BR IF OK  
2460 011710 104003      HLT      3            ;COMPARISON ERROR  
2461 011712 104400      2$:      SCOPE          ;SCOPE THIS TEST  
2462  
2463
```

```
:***** TEST 32 *****  
:*SECONDARY REGISTER SELECTOR READ/WRITE TEST.  
:*SET BIT2, VERIFY BIT2 WAS SET.  
:*CLEAR BIT2, VERIFY BIT2 WAS CLEARED.  
:*****
```

: TEST 32

```
2471  
2472 011714 012737 000032 001226 TST32: MOV      #32,TSTNO  
2473 011722 012737 000032 001216      MOV      #TST33,NEXT  
2474 011730 013703 000000      MOV      DVSRS,R3      ;SET REGISTER TO BE TESTED.  
2475 011734 012705 000004      MOV      #BIT2,R5      ;SET 'EXPECTED '  
2476 011740 010513      MOV      R5,(R3)      ;WRITE THE REGISTER.  
2477 011742 011304      MOV      (R3),R4      ;READ THE REGISTER.  
2478 011744 020504      CMP      R5,R4        ;R5=GOOD; R4=UNKNOWN.
```

```
2479 011746 001401 BEQ 1$ ;ARE THEY THE SAME?  
2480 011750 104003 HLT 3 ;COMPARISON ERROR.  
2481 011752 040513 1$: BIC R5,(R3) ;CLEAR BIT2  
2482 011754 011304 MOV (R3),R4 ;READ THE REGISTER.  
2483 011756 005005 CLR R5 ;SET 'EXPECTED'  
2484 011760 020504 CMP R5,R4 ;R5=GOOD; R4=?  
2485 011762 001401 BEQ 2$ ;BR IF OK  
2486 011764 104003 HLT 3 ;COMPARISON ERROR  
2487 011766 104400 2$: SCOPE ;SCOPE THIS TEST  
2488  
2489  
2490  
2491  
2492  
2493  
2494  
2495  
2496  
2497
```

```
:***** TEST 33 *****  
:*SECONDARY REGISTER SELECTOR READ/WRITE TEST.  
:*SET BIT3, VERIFY BIT3 WAS SET.  
:*CLEAR BIT3, VERIFY BIT3 WAS CLEARED.  
:*****
```

: TEST 33

```
2498 011770 012737 000033 001226 TST33: MOV #33,TSTNO  
2499 011776 012737 012044 001216 MOV #TST34,NEXT  
2500 012004 013703 001372 MOV DVSRS,R3 ;SET REGISTER TO BE TESTED.  
2501 012010 012705 000010 MOV #BIT3,R5 ;SET 'EXPECTED'  
2502 012014 010513 MOV R5,(R3) ;WRITE THE REGISTER.  
2503 012016 011304 MOV (R3),R4 ;READ THE REGISTER.  
2504 012020 020504 CMP R5,R4 ;R5=GOOD; R4=UNKNOWN.  
2505 012022 001401 BEQ 1$ ;ARE THEY THE SAME?  
2506 012024 104003 HLT 3 ;COMPARISON ERROR.  
2507 012026 040513 1$: BIC R5,(R3) ;CLEAR BIT3  
2508 012030 011304 MOV (R3),R4 ;READ THE REGISTER.  
2509 012032 005005 CLR R5 ;SET 'EXPECTED'  
2510 012034 020504 CMP R5,R4 ;R5=GOOD; R4=?  
2511 012036 001401 BEQ 2$ ;BR IF OK  
2512 012040 104003 HLT 3 ;COMPARISON ERROR  
2513 012042 104400 2$: SCOPE ;SCOPE THIS TEST  
2514  
2515
```

```
:***** TEST 34 *****  
:*SECONDARY REGISTER SELECTOR READ/WRITE TEST.  
:*SET BIT8, VERIFY BIT8 WAS SET.  
:*CLEAR BIT8, VRFIFY BIT8 WAS CLEARED.  
:*****
```

: TEST 34

```
2524 012044 012737 000034 001226 TST34: MOV #34,TSTNO  
2525 012052 012737 012120 001216 MOV #TST35,NEXT  
2526 012060 013703 001372 MOV DVSRS,R3 ;SET REGISTER TO BE TESTED.  
2527 012064 012705 000400 MOV #BIT8,R5 ;SET 'EXPECTED'  
2528 012070 010513 MOV R5,(R3) ;WRITE THE REGISTER.  
2529 012072 011304 MOV (R3),R4 ;READ THE REGISTER.  
2530 012074 020504 CMP R5,R4 ;R5=GOOD; R4=UNKNOWN.  
2531 012076 001401 BEQ 1$ ;ARE THEY THE SAME?  
2532 012100 104003 HLT 3 ;COMPARISON ERROR.  
2533 012102 040513 1$: BIC R5,(R3) ;CLEAR BIT8  
2534 012104 011304 MOV (R3),R4 ;READ THE REGISTER.
```

```
2535 012106 005005 CLR R5 ;SET 'EXPECTED'  
2536 012110 020504 CMP R5,R4 ;R5=GOOD; R4=?  
2537 012112 001401 BEQ 2$ ;BR IF OK  
2538 012114 104003 HLT 3 ;COMPARISON ERROR  
2539 012116 104400 2$: SCOPE ;SCOPE THIS TEST
```

```
:***** TEST 35 *****  
:*SECONDARY REGISTER SELECTOR READ/WRITE TEST.  
:*SET BIT9, VERIFY BIT9 WAS SET.  
:*CLEAR BIT9, VERIFY BIT9 WAS CLEARED.  
:*****
```

: TEST 35

```
2549 2550 012120 012737 000035 001226 TST35: MOV #35,TSTNO  
2551 012126 012737 012174 001216 MOV #TST36,NEXT  
2552 012134 013703 001372 MOV DVSRS,R3 ;SET REGISTER TO BE TESTED.  
2553 012140 012705 001000 MOV #BIT9,R5 ;SET 'EXPECTED'  
2554 012144 010513 MOV R5,(R3) ;WRITE THE REGISTER.  
2555 012146 011304 MOV (R3),R4 ;READ THE REGISTER.  
2556 012150 020504 CMP R5,R4 ;R5=GOOD; R4=UNKNOWN.  
2557 012152 001401 BEQ 1$ ;ARE THEY THE SAME?  
2558 012154 104003 HLT 3 ;COMPARISON ERROR.  
2559 012156 040513 1$: BIC R5,(R3) ;CLEAR BIT9  
2560 012160 011304 MOV (R3),R4 ;READ THE REGISTER.  
2561 012162 005005 CLR R5 ;SET 'EXPECTED'  
2562 012164 020504 CMP R5,R4 ;R5=GOOD; R4=?  
2563 012166 001401 BEQ 2$ ;BR IF OK  
2564 012170 104003 HLT 3 ;COMPARISON ERROR  
2565 012172 104400 2$: SCOPE ;SCOPE THIS TEST
```

```
:***** TEST 36 *****  
:*SECONDARY REGISTER SELECTOR READ/WRITE TEST.  
:*SET BIT10, VERIFY BIT10 WAS SET.  
:*CLEAR BIT10, VERIFY BIT10 WAS CLEARED.  
:*****
```

: TEST 36

```
2575 2576 012174 012737 000036 001226 TST36: MOV #36,TSTNO  
2577 012202 012737 012250 001216 MOV #TST37,NEXT  
2578 012210 013703 001372 MOV DVSRS,R3 ;SET REGISTER TO BE TESTED.  
2579 012214 012705 002000 MOV #BIT10,R5 ;SET 'EXPECTED'  
2580 012220 010513 MOV R5,(R3) ;WRITE THE REGISTER.  
2581 012222 011304 MOV (R3),R4 ;READ THE REGISTER.  
2582 012224 020504 CMP R5,R4 ;R5=GOOD; R4=UNKNOWN.  
2583 012226 001401 BEQ 1$ ;ARE THEY THE SAME?  
2584 012230 104003 HLT 3 ;COMPARISON ERROR.  
2585 012232 040513 1$: BIC R5,(R3) ;CLEAR BIT10  
2586 012234 011304 MOV (R3),R4 ;READ THE REGISTER.  
2587 012236 005005 CLR R5 ;SET 'EXPECTED'  
2588 012240 020504 CMP R5,R4 ;R5=GOOD; R4=?  
2589 012242 001401 BEQ 2$ ;BR IF OK  
2590 012244 104003 HLT 3 ;COMPARISON ERROR
```

2591 012246 104400 2\$: SCOPE ;SCOPE THIS TEST

2592  
2593  
2594 :\*\*\*\*\* TEST 37 \*\*\*\*\*  
2595 :\*SECONDARY REGISTER SELECTOR READ/WRITE TEST.  
2596 :\*SET BIT11, VERIFY BIT11 WAS SET.  
2597 :\*CLEAR BIT11, VERIFY BIT11 WAS CLEARED.  
2598 :\*\*\*\*\*  
2599

2600 ; TEST 37  
2601 -----

2602 012250 012737 000037 001226 TST37: MOV #37,TSTNO  
2603 012256 012737 012324 001216 MOV #TST40,NEXT  
2604 012264 013703 001372 MOV DVSRS,R3 ;SET REGISTER TO BE TESTED.  
2605 012270 012705 004000 MOV #BIT11,R5 ;SET 'EXPECTED'  
2606 012274 010513 MOV R5,(R3) ;WRITE THE REGISTER.  
2607 012276 011304 MOV (R3),R4 ;READ THE REGISTER.  
2608 012300 020504 CMP R5,R4 ;R5=GOOD; R4=UNKNOWN.  
2609 012302 001401 BEQ 1\$ ;ARE THEY THE SAME?  
2610 012304 104003 HLT 3 ;COMPARISON ERROR.  
2611 012306 040513 1\$: BIC R5,(R3) ;CLEAR BIT11  
2612 012310 011304 MOV (R3),R4 ;READ THE REGISTER.  
2613 012312 005005 CLR R5 ;SET 'EXPECTED'  
2614 012314 020504 CMP R5,R4 ;R5=GOOD; R4=?  
2615 012316 001401 BEQ 2\$ ;BR IF OK  
2616 012320 104003 HLT 3 ;COMPARISON ERROR  
2617 012322 104400 2\$: SCOPE ;SCOPE THIS TEST

2618  
2619 :\*\*\*\*\* TEST 40 \*\*\*\*\*  
2620 :\*SECONDARY REGISTER ACCESS REG. READ/WRITE TEST  
2621 :\*SET EACH INDIVIDUAL BIT; VERIFY EACH INDIVIDUAL BIT SET.  
2622 :\*CLEAR EACH INDIVIDUAL BIT; VERIFY CLEAR.  
2623 :\*\*\*\*\*  
2624  
2625

2626 ; TEST 40  
2627 -----  
2628

2629 012324 012737 000040 001226 TST40: MOV #40,TSTNO  
2630 012332 012737 012426 001216 MOV #TST41,NEXT  
2631 012340 013703 001376 MOV DVSRA,R3 ;SET CSR POINTER INTO R3  
2632 012344 012705 177777 MOV #-1,R5 ;SET EXPECTED TO ALL 1'S  
2633 012350 010513 MOV R5,(R3) ;WRITE REGISTER WITH EXPECTED DATA  
2634 012352 011304 MOV (R3),R4 ;READ THE REGISTER  
2635 012354 020504 CMP R5,R4 ;WAS READ EQUAL TO WRITTEN??  
2636 012356 001401 BEQ 1\$ ;BR IF YES  
2637 012360 104003 HLT 3 ;PRIMARY REGISTER READ/WRITE TEST  
2638 012362 005005 1\$: CLR R5 ;SET DATA TO ZERO  
2639 012364 010513 MOV R5,(R3) ;WRITE REGISTER  
2640 012366 011304 MOV (R3),R4 ;READ REGISTER  
2641 012370 020504 CMP R5,R4 ;REGISTER WRITTEN OK?  
2642 012372 001401 BEQ 2\$ ;BR IF YES  
2643 012374 104003 HLT 3 ;PRIMARY REGISTER DATA ERROR  
2644 012376 012705 000001 2\$: MOV #1,R5 ;SET FOR 'FLOATING 1'  
2645 012402 010513 3\$: MOV R5,(R3) ;WRITE DATA  
2646 012404 011304 MOV (R3),R4 ;READ DATA

```
2647 012406 020504      CMP      R5,R4      ;DATA OK?
2648 012410 001401      BEQ      4$         ;BR IF YES
2649 012412 104003      HLT      3         ;DATA ERROR
2650 012414 000241      4$:      CLC         ;ZERO CPU CARRY
2651 012416 006105      ROL      R5        ;UPDATE DATA
2652 012420 001370      BNE      3$         ;BR IF MORE TO GO
2653 012422 005013      CLR      (R3)      ;LEAVE REG ALL 0'S
2654 012424 104400      SCOPE          ;SCOPE THIS TEST.
```

```
2655
2656
2657 ;***** TEST 41 *****
2658 ;*SPECIAL FUNCT. REGISTER READ/WRITE TEST
2659 ;*SET EACH INDIVIDUAL BIT; VERIFY EACH INDIVIDUAL BIT SET.
2660 ;*CLEAR EACH INDIVIDUAL BIT; VERIFY CLEAR.
2661 ;*****
```

```
2662 ; TEST 41
2663
2664 ;-----
2665 TST41:
2666 012426 012737 000041 001226 MOV      #41,TSTNO
2667 012434 012737 012536 001216 MOV      #TST42,NEXT
2668 012442 012777 000010 166712 MOV      #BIT3,@DVSCR ;SET SOURCE SEL
2669 012450 013703 001400      MOV      DVSFR,R3 ;SET CSR POINTER INTO R3
2670 012454 012705 177777      MOV      #-1,R5 ;SET EXPECTED TO ALL 1'S
2671 012460 010513      MOV      R5,(R3) ;WRITE REGISTER WITH EXPECTED DATA
2672 012462 011304      MOV      (R3),R4 ;READ THE REGISTER
2673 012464 020504      CMP      R5,R4 ;WAS READ EQUAL TO WRITTEN??
2674 012466 001401      BEQ      1$         ;BR IF YES
2675 012470 104003      HLT      3         ;PRIMARY REGISTER READ/WRITE TEST
2676 012472 005005      1$:      CLR      R5        ;SET DATA TO ZERO
2677 012474 010513      MOV      R5,(R3) ;WRITE REGISTER
2678 012476 011304      MOV      (R3),R4 ;READ REGISTER
2679 012500 020504      CMP      R5,R4 ;REGISTER WRITTEN OK?
2680 012502 001401      BEQ      2$         ;BR IF YES
2681 012504 104003      HLT      3         ;PRIMARY REGISTER DATA ERROR
2682 012506 012705 000001 2$:      MOV      #1,R5 ;SET FOR 'FLOATING 1'
2683 012512 010513      3$:      MOV      R5,(R3) ;WRITE DATA
2684 012514 011304      MOV      (R3),R4 ;READ DATA
2685 012516 020504      CMP      R5,R4 ;DATA OK?
2686 012520 001401      BEQ      4$         ;BR IF YES
2687 012522 104003      HLT      3         ;DATA ERROR
2688 012524 000241      4$:      CLC         ;ZERO CPU CARRY
2689 012526 006105      ROL      R5        ;UPDATE DATA
2690 012530 001370      BNE      3$         ;BR IF MORE TO GO
2691 012532 005013      CLR      (R3)      ;LEAVE REG ALL 0'S
2692 012534 104400      SCOPE          ;SCOPE THIS TEST.
```

```
2693
2694
2695 ;***** TEST 42 *****
2696
2697 ;*NPR STATUS REG. TEST
2698 ;*TEST THAT NPR STATUS REG. CANNOT BE WRITTEN
2699 ;*READ THE NPR STATUS REG. AND STORE THE DATA;
2700 ;*COMPLEMENT THE DATA AND WRITE THE NPR STATUS REG.
2701 ;*VERIFYING THAT THE NPR STATUS REG. DID NOT CHANGE.
2702 ;*****
```

2703  
2704  
2705  
2706 012536 012737 000042 001226  
2707 012544 012737 012600 001216  
2708 012552 013703 001402  
2709  
2710 012556 011305  
2711 012560 010504  
2712 012562 005104  
2713 012564 010413  
2714 012566 011304  
2715 012570 020504  
2716 012572 001401  
2717 012574 104003  
2718 012576 104400

: TEST 42

```
-----  
TST42:  MOV    #42,TSTNO  
        MOV    #TST43,NEXT  
        MOV    DVNSR,R3  
        MOV    (R3),R5      ;GET THE REGISTER.  
        MOV    R5,R4      ;READ THE REGISTER INTO R5  
        COM    R4          ;SAVE REG INTO R4  
        MOV    R4,(R3)     ;MAKE R4 OPPOSITE TO REGISTER  
        MOV    (R3),R4     ;WRITE THE REGISTER WITH THE COMPLIMENT  
        CMP    R5,R4      ;READ THE REGISTER.  
        BEQ    18         ;IS THE REGISTER EQUAL TO ZERO.  
        HLT    3          ;BR IF OK.  
18:     HLT    3          ;REGISTER NOT ZERO.  
        SCOPE           ;SCOPE THIS TEST.
```

\*\*\*\*\* TEST 43 \*\*\*\*\*  
: \*DV11 RESERVED REGISTER READ/WRITE TEST.  
: \*SET BIT0, VERIFY BIT0 WAS SET.  
: \*CLEAR BIT0, VERIFY BIT0 WAS CLEARED.  
\*\*\*\*\*

2720  
2721  
2722  
2723  
2724  
2725  
2726  
2727  
2728  
2729 012600 012737 000043 001226  
2730 012606 012737 012654 001216  
2731 012614 013703 001404  
2732 012620 012705 000001  
2733 012624 010513  
2734 012626 011304  
2735 012630 020504  
2736 012632 001401  
2737 012634 104003  
2738 012636 040513  
2739 012640 011304  
2740 012642 005005  
2741 012644 020504  
2742 012646 001401  
2743 012650 104003  
2744 012652 104400

: TEST 43

```
-----  
TST43:  MOV    #43,TSTNO  
        MOV    #TST44,NEXT  
        MOV    RESV16,R3   ;SET REGISTER TO BE TESTED.  
        MOV    #BIT0,R5   ;SET 'EXPECTED'  
        MOV    R5,(R3)    ;WRITE THE REGISTER.  
        MOV    (R3),R4    ;READ THE REGISTER.  
        CMP    R5,R4      ;R5=GOOD; R4=UNKNOWN.  
        BEQ    18         ;ARE THEY THE SAME?  
        HLT    3          ;COMPARISON ERROR.  
18:     BIC    R5,(R3)    ;CLEAR BIT0  
        MOV    (R3),R4    ;READ THE REGISTER.  
        CLR    R5         ;SET 'EXPECTED'  
        CMP    R5,R4      ;R5=GOOD; R4=?  
        BEQ    28         ;BR IF OK  
        HLT    3          ;COMPARISON ERROR  
28:     HLT    3          ;SCOPE THIS TEST  
        SCOPE
```

\*\*\*\*\* TEST 44 \*\*\*\*\*  
: \*DV11 RESERVED REGISTER READ/WRITE TEST.  
: \*SET BIT1, VERIFY BIT1 WAS SET.  
: \*CLEAR BIT1, VERIFY BIT1 WAS CLEARED.  
\*\*\*\*\*

2745  
2746  
2747  
2748  
2749  
2750  
2751  
2752  
2753  
2754  
2755 012654 012737 000044 001226  
2756 012662 012737 012730 001216  
2757 012670 013703 001404  
2758 012674 012705 000002

: TEST 44

```
-----  
TST44:  MOV    #44,TSTNO  
        MOV    #TST45,NEXT  
        MOV    RESV16,R3   ;SET REGISTER TO BE TESTED.  
        MOV    #BIT1,R5   ;SET 'EXPECTED'  
        HLT    3
```

```
2759 012700 010513      MOV      R5,(R3)      ;WRITE THE REGISTER.
2760 012702 011304      MOV      (R3),R4      ;READ THE REGISTE ..
2761 012704 020504      CMP      R5,R4        ;R5=GOOD; R4=UNKNOWN.
2762 012706 001401      BEQ      1$           ;ARE THEY THE SAME?
2763 012710 104003      HLT      3            ;COMPARISON ERROR.
2764 012712 040513      1$:      BIC      R5,(R3)    ;CLEAR BIT1
2765 012714 011304      MOV      (R3),R4      ;READ THE REGISTER.
2766 012716 005005      CLR      R5           ;SET 'EXPECTED'
2767 012720 020504      CMP      R5,R4        ;R5=GOOD; R4=?
2768 012722 001401      BEQ      2$           ;BR IF OK
2769 012724 104003      HLT      3            ;COMPARISON ERROR
2770 012726 104400      2$:      SCOPE        ;SCOPE THIS TEST
```

```
***** TEST 45 *****
;*DV11 RESERVED REGISTER READ/WRITE TEST.
;*SET BIT2, VERIFY BIT2 WAS SET.
;*CLEAR BIT2, VERIFY BIT2 WAS CLEARED.
*****
```

: TEST 45

```
2780
2781 012730 012737 000045 001226 1ST45: MOV      #45,TSTNO
2782 012736 012737 013004 001216  MOV      #TST46,NEXT
2783 012744 013703 001404  MOV      RESV16,R3      ;SET REGISTER TO BE TESTED.
2784 012750 012705 000004  MOV      #BIT2,R5      ;SET 'EXPECTED'
2785 012754 010513  MOV      R5,(R3)      ;WRITE THE REGISTER.
2786 012756 011304  MOV      (R3),R4      ;READ THE REGISTER.
2787 012760 020504  CMP      R5,R4        ;R5=GOOD; R4=UNKNOWN.
2788 012762 001401  BEQ      1$           ;ARE THEY THE SAME?
2789 012764 104003  HLT      3            ;COMPARISON ERROR.
2790 012766 040513  1$:      BIC      R5,(R3)    ;CLEAR BIT2
2791 012770 011304  MOV      (R3),R4      ;READ THE REGISTER.
2792 012772 005005  CLR      R5           ;SET 'EXPECTED'
2793 012774 020504  CMP      R5,R4        ;R5=GOOD; R4=?
2794 012776 001401  BEQ      2$           ;BR IF OK
2795 013000 104003  HLT      3            ;COMPARISON ERROR
2796 013002 104400  2$:      SCOPE        ;SCOPE THIS TEST
```

```
***** TEST 46 *****
;*DV11 RESERVED REGISTER READ/WRITE TEST.
;*SET BIT3, VERIFY BIT3 WAS SET.
;*CLEAR BIT3, VERIFY BIT3 WAS CLEARED.
*****
```

: TEST 46

```
2806
2807 013004 012737 000046 001226 1ST46: MOV      #46,TSTNO
2808 013012 012737 013060 001216  MOV      #TST47,NEXT
2809 013020 013703 001404  MOV      RESV16,R3      ;SET REGISTER TO BE TESTED.
2810 013024 012705 000010  MOV      #BIT3,R5      ;SET 'EXPECTED'
2811 013030 010513  MOV      R5,(R3)      ;WRITE THE REGISTER.
2812 013032 011304  MOV      (R3),R4      ;READ THE REGISTER.
2813 013034 020504  CMP      R5,R4        ;R5=GOOD; R4=UNKNOWN.
2814 013036 001401  BEQ      1$           ;ARE THEY THE SAME?
```



```
2815 013040 104003          HLT      3          :COMPARISON ERROR.  
2816 013042 040513          1$: BIC      R5,(R3) :CLEAR BIT3  
2817 013044 011304          MOV      (R3),R4   :READ THE REGISTER.  
2818 013046 005005          CLR      R5        :SET 'EXPECTED'  
2819 013050 020504          CMP      R5,R4     :R5=GOOD; R4=?  
2820 013052 001401          BEQ      2$        :BR IF OK  
2821 013054 104003          HLT      3          :COMPARISON ERROR  
2822 013056 104400          2$: SCOPE         :SCOPE THIS TEST
```

```
2823  
2824  
2825 :***** TEST 47 *****  
2826 :*DV11 RESERVED REGISTER READ/WRITE TEST.  
2827 :*SET BIT4, VERIFY BIT4 WAS SET.  
2828 :*CLEAR BIT4, VERIFY BIT4 WAS CLEARED.  
2829 :*****
```

```
2830  
2831 : TEST 47  
2832 :-----  
2833 013060 012737 000047 001226 TST47: MOV      #47,TSTNO  
2834 013066 012737 013134 001216      MOV      #TST50,NEXT  
2835 013074 013703 001404          MOV      RESV16,R3   :SET REGISTER TO BE TESTED.  
2836 013100 012705 000020          MOV      #BIT4,R5    :SET 'EXPECTED'  
2837 013104 010513          MOV      R5,(R3)     :WRITE THE REGISTER.  
2838 013106 011304          MOV      (R3),R4     :READ THE REGISTER.  
2839 013110 020504          CMP      R5,R4       :R5=GOOD; R4=UNKNOWN.  
2840 013112 001401          BEQ      1$          :ARE THEY THE SAME?  
2841 013114 104003          HLT      3          :COMPARISON ERROR.  
2842 013116 040513          1$: BIC      R5,(R3) :CLEAR BIT4  
2843 013120 011304          MOV      (R3),R4     :READ THE REGISTER.  
2844 013122 005005          CLR      R5          :SET 'EXPECTED'  
2845 013124 020504          CMP      R5,R4       :R5=GOOD; R4=?  
2846 013126 001401          BEQ      2$          :BR IF OK  
2847 013130 104003          HLT      3          :COMPARISON ERROR  
2848 013132 104400          2$: SCOPE         :SCOPE THIS TEST
```

```
2849  
2850  
2851 :***** TEST 50 *****  
2852 :*DV11 RESERVED REGISTER READ/WRITE TEST.  
2853 :*SET BITS, VERIFY BITS WAS SET.  
2854 :*CLEAR BITS, VERIFY BITS WAS CLEARED.  
2855 :*****
```

```
2856  
2857 : TEST 50  
2858 :-----  
2859 013134 012737 000050 001226 TST50: MOV      #50,TSTNO  
2860 013142 012737 013210 001216      MOV      #TST51,NEXT  
2861 013150 013703 001404          MOV      RESV16,R3   :SET REGISTER TO BE TESTED.  
2862 013154 012705 000040          MOV      #BITS,R5    :SET 'EXPECTED'  
2863 013160 010513          MOV      R5,(R3)     :WRITE THE REGISTER.  
2864 013162 011304          MOV      (R3),R4     :READ THE REGISTER.  
2865 013164 020504          CMP      R5,R4       :R5=GOOD; R4=UNKNOWN.  
2866 013166 001401          BEQ      1$          :ARE THEY THE SAME?  
2867 013170 104003          HLT      3          :COMPARISON ERROR.  
2868 013172 040513          1$: BIC      R5,(R3) :CLEAR BITS  
2869 013174 011304          MOV      (R3),R4     :READ THE REGISTER.  
2870 013176 005005          CLR      R5          :SET 'EXPECTED'
```

```
2871 013200 020504      CMP      R5,R4      ;R5=GOOD; R4=?
2872 013202 001401      BEQ      2$         ;BR IF OK
2873 013204 104003      HLT      3          ;COMPARISON ERROR
2874 013206 104400      2$:      SCOPE     ;SCOPE THIS TEST
2875
2876
2877
```

```
:***** TEST 51 *****
:*DV11 RESERVED REGISTER READ/WRITE TEST.
:*SET BIT6, VERIFY BIT6 WAS SET.
:*CLEAR BIT6, VERIFY BIT6 WAS CLEARED.
:*****
```

: TEST 51

```
-----
2884 013210 012737 000051 001226 TST51: MOV      #51,TSTNO
2885 013216 012737 013264 001216      MOV      #TST52,NEXT
2886 013224 013703 001404      MOV      RESV16,R3      ;SET REGISTER TO BE TESTED.
2887 013230 012705 000100      MOV      #BIT6,R5      ;SET 'EXPECTED'
2888 013234 010513      MOV      R5,(R3)       ;WRITE THE REGISTER.
2889 013236 011304      MOV      (R3),R4       ;READ THE REGISTER.
2890 013240 020504      CMP      R5,R4         ;R5=GOOD; R4=UNKNOWN.
2891 013242 001401      BEQ      1$           ;ARE THEY THE SAME?
2892 013244 104003      HLT      3            ;COMPARISON ERROR.
2893 013246 040513      1$:      BIC      R5,(R3)     ;CLEAR BIT6
2894 013250 011304      MOV      (R3),R4       ;READ THE REGISTER.
2895 013252 005005      CLR      R5            ;SET 'EXPECTED'
2896 013254 020504      CMP      R5,R4         ;R5=GOOD; R4=?
2897 013256 001401      BEQ      2$           ;BR IF OK
2898 013260 104003      HLT      3            ;COMPARISON ERROR
2900 013262 104400      2$:      SCOPE     ;SCOPE THIS TEST
2901
2902
2903
```

```
:***** TEST 52 *****
:*DV11 RESERVED REGISTER READ/WRITE TEST.
:*SET BIT7, VERIFY BIT7 WAS SET.
:*CLEAR BIT7, VERIFY BIT7 WAS CLEARED.
:*****
```

: TEST 52

```
-----
2910 013264 012737 000052 001226 TST52: MOV      #52,TSTNO
2911 013272 012737 013340 001216      MOV      #TST53,NEXT
2912 013300 013703 001404      MOV      RESV16,R3      ;SET REGISTER TO BE TESTED.
2913 013304 012705 000200      MOV      #BIT7,R5      ;SET 'EXPECTED'
2914 013310 010513      MOV      R5,(R3)       ;WRITE THE REGISTER.
2915 013312 011304      MOV      (R3),R4       ;READ THE REGISTER.
2916 013314 020504      CMP      R5,R4         ;R5=GOOD; R4=UNKNOWN.
2917 013316 001401      BEQ      1$           ;ARE THEY THE SAME?
2918 013320 104003      HLT      3            ;COMPARISON ERROR.
2919 013322 040513      1$:      BIC      R5,(R3)     ;CLEAR BIT7
2920 013324 011304      MOV      (R3),R4       ;READ THE REGISTER.
2921 013326 005005      CLR      R5            ;SET 'EXPECTED'
2922 013330 020504      CMP      R5,R4         ;R5=GOOD; R4=?
2923 013332 001401      BEQ      2$           ;BR IF OK
2924 013334 104003      HLT      3            ;COMPARISON ERROR
2925 013336 104400      2$:      SCOPE     ;SCOPE THIS TEST
2926
```

2927  
 2928  
 2929  
 2930  
 2931  
 2932  
 2933  
 2934  
 2935  
 2936  
 2937  
 2938  
 2939  
 2940  
 2941  
 2942  
 2943  
 2944  
 2945  
 2946  
 2947  
 2948  
 2949  
 2950  
 2951  
 2952  
 2953  
 2954  
 2955  
 2956  
 2957  
 2958  
 2959  
 2960  
 2961  
 2962  
 2963  
 2964  
 2965  
 2966  
 2967  
 2968  
 2969  
 2970  
 2971  
 2972  
 2973

```

:***** TEST 53 *****
:*TEST OF THE BYTE OPERATIONS FOR THE DV11
:*SYSTEM CONTROL REG AND THE SECONDARY REG SEL.
:*THE TEST WILL CLEAR DVSCR AND THE WRITE (LOW BYTE)
:*BIT3; THEN VERIFY ONLY BIT3 IS SET; THEN THE
:*TEST WILL WRITE BIT 8(HIGH BYTE) AND VERIFY THAT
:*BIT8 AND BIT3 ARE SET. THE EXACT PROCEEDURE
:*WILL BE USED ON THE DVSRS REGISTER.
:*****
    
```

: TEST 53

-----

```

TST53:  MOV    #53,TSTNO
        MOV    #TST54,NEXT
        MOV    DVSCR,R3          ;SET DVSCR POINTER
        CLR    (R3)             ;MAKE SURE IT =0
        MOV    #BIT3,R5         ;LOAD EXPECTED RESULTS
        MOVB   R5,(R3)          ;"WRITE" BYTE (LOW) BIT3
        MOV    (R3),R4          ;READ (WORD) RESULT
        CMP    R5,R4            ;MAKE SURE ONLY BIT3 IS SET
        BEQ    1$              ;BR IF OK
        HLT    3                ;DVSRS WRONG
1$:     MOV    #BIT8+BIT3,R5     ;SET EXPECTED DATA
        MOVB   #BIT0,1(R3)      ;"WRITE" BYTE (HIGH) BIT0 [BIT8 OF WORD]
        MOV    (R3),R4          ;READ (WORD) RESULT
        CMP    R5,R4            ;OK?
        BEQ    2$              ;
        HLT    3                ;DVSRS WRONG
2$:     CLR    (R3)             ;LEAVE REGISTERED CLEARED
        MOV    DVSRS,R3         ;GET NEXT REGISTER FOR BYTE TEST.
        CLR    (R3)             ;MAKE SURE WORD IS =0.
        MOV    #BIT3,R5         ;SET EXPECTED
        MOVB   R5,(R3)          ;WRITE BYTE (LOW) BIT3
        MOV    (R3),R4          ;READ RESULT
        CMP    R5,R4            ;OK?
        BEQ    4$              ;
        HLT    3                ;DVSRS WRONG
4$:     MOV    #BIT8+BIT3,R5     ;SET EXPECTED
        MOVB   #BIT0,1(R3)      ;WRITE BYTE (HIGH) BIT0 [BIT8 OF WORD]
        MOV    (R3),R4          ;READ RESULT
        CMP    R5,R4            ;OK
        BEQ    5$              ;
        HLT    3                ;DVSRS FAILED
5$:     SCOPE                    ;SCOPE TEST
    
```

```

2974
2975
2976
2977
2978
2979
2980
2981
2982
2983 013474 012737 000054 001226
2984 013502 012737 013676 001216
2985 013510 012737 013526 001220
2986 013516 005000
2987 013520 005001
2988 013522 013702 001376
2989 013526 110077 165640
2990 013532 110177 165636
2991 013536 005004
2992 013540 010412
2993 013542 011203
2994 013544 020403
2995 013546 001401
2996 013550 104002
2997 013552 104401
2998 013554 005104
2999 013556 001370
3000 013560 012704 000001
3001 013564 012737 013572 001220
3002 013572 010412
3003 013574 011203
3004 013576 020403
3005 013600 001401
3006 013602 104002
3007 013604 104401
3008 013606 000241
3009 013610 006104
3010 013612 001367
3011 013614 012704 177776
3012 013620 012737 013626 001220
3013 013626 010412
3014 013630 011203
3015 013632 020403
3016 013634 001401
3017 013636 104002
3018 013640 104401
3019 013642 000261
3020 013644 006104
3021 013646 103767
3022 013650 005012
3023 013652 005201
3024 013654 022701 000020
3025 013660 001322
3026 013662 005001
3027 013664 005200
3028 013666 022700 000020
3029 013672 001315

:***** TEST 54 *****
:*SECONDARY REGISTER READ/WRITE TESTS
:*READ/WRITE TEST. READ AND WRITE DIFFERENT DATA
:*PATTERENS INTO THE SECONDARY REGISTERS VRFIFING
:*THAT WHAT WAS READ MATCHES WHAT WAS WRITTEN.
:*****

: TEST 54
-----
TST54: MOV #54,TSTNO
MOV #TST55,NEXT
MOV #1$,LOCK
CLR R0 ;SEL LINE NUMBER TO ZERO
CLR R1 ;SET SEC. REG TO ZERO ALSO
MOV DVSR,R2 ;SET ADDRESS POINTER INTO R2
1$: MOVB R0,@DVSR ;LOAD LINE NUMBER
MOVB R1,@DVSRSH ;LOAD SEC. REG.
CLR R4 ;ZERO DATA PATTERN
2$: MOV R4,(R2) ;LOAD DATA INTO DV11 REGISTER
MOV (R2),R3 ;READ BACK THE DATA
CMP R4,R3 ;WAS WHAT WAS WRITTEN READ BACK??
BEQ 64$ ;BR IF DATA OK
HLT 2 ;SECONDARY REGISTER READ WRITE ERROR.
64$: SCOP1 ;LOCK ON DATA,LINE,AND SEC REG? (SW09=1)
COM R4 ;MAKE DATA ALL 1'S
BNE 2$ ;RETURN AND WRITE IT INTO THE REGISTER
MOV #1,R4 ;GET READY FOR THE 'FLOATING 1'
3$: MOV #3$,LOCK ;SET IF SW09=1
MOV R4,(R2) ;LOAD DATA
MOV (R2),R3 ;READ DATA
CMP R4,R3 ;DATA OK??
BEQ 65$ ;BR IF OK.
HLT 2 ;SECONDARY REGISTER READ/WRITE ERROR
65$: SCOP1 ;SW09=1?
CLC ;ZERO CPU CARRY
ROL R4 ;FLOAT THE 1
BNE 3$ ;IF NOT DONE WRITE NEW PATTERN.
MOV #^(<1>,R4 ;GET READY FOR THE 'FLOATING 0'
4$: MOV #4$,LOCK ;SET IF SW09=1
MOV R4,(R2) ;WRITE DATA
MOV (R2),R3 ;READ DATA
CMP R4,R3 ;DATA OK??
BEQ 66$ ;BR IF OK!
HLT 2 ;SECONDARY REGISTER DATA COMPARE ERROR
66$: SCOP1 ;SW09=1??
SEC ;SET CPU CARRY
ROL R4 ;CHANGE DATA PATTERN
BCS 4$ ;IF NOT DONE ,GO BACK WITH NEW PATTERN
CLR (R2) ;ZERO THE REGISTER (ALL DONE WITH THIS ONE.
INC R1 ;UPDATE THE SEC REG POINTER
CMP #16.,R1 ;ALL SEC REGISTERS DONE?
BNE 1$ ; BR IF NO.
CLR R1 ;ZERO SEC REG POINTER
INC R0 ;UPDATE LINE NUMBER POINTER
CMP #16.,R0 ;ALL LINES DONE?
BNE 1$ ;BR IF NO
    
```

3030 013674 104400 SCOPE ;SCOPE THIS TEST

3031  
3032  
3033 :\*\*\*\*\* TEST 55 \*\*\*\*\*  
3034 :\*INDIVIDUAL LINE DUEL ADDRESS TESTS  
3035 :\*THIS TEST VERIFIES THAT WRITING ONE SECONDARY  
3036 :\*REGISTER FOR A SPECIFIC LINE DOES NOT ALTER  
3037 :\*ANY OTHER SECONDARY REGISTER FOR THAT LINE.  
3038 :\*\*\*\*\*  
3039

3040 ; TEST 55

3041 :-----  
3042 013676 012737 000055 001226 TST55: MOV #55,TSTNO  
3043 013704 012737 014036 001216 MOV #TST56,NEXT  
3044 013712 012737 013772 001220 MOV #2\$,LOCK  
3045 013720 005000 CLR R0 ;SELECT THE LINE NUMBER.  
3046 013722 005001 65\$: CLR R1 ;SET FOR SEC. REG. POINTER.  
3047 013724 005004 CLR R4 ;SET DATA  
3048 013726 013702 001376 MOV DVSR,R2 ;SET ACCESS REGISTER.  
3049 013732 110077 165434 MOV R0,@DVSR ;SELECT THE LINE NUMBER  
3050 013736 110177 165432 1\$: MOV R1,@DVSRSH ;SELECT THE SEC. REG.  
3051 013742 010412 MOV R4,(R2) ;WRITE SEC. REG.  
3052 013744 062704 010421 ADD #^B<0001000100010001>,R4  
3053 ;UPDATE DATA  
3054 013750 005201 INC R1 ;UPDATE SECONDARY REG. POINTER  
3055 013752 022701 000020 CMP #16.,R1 ;ALL SEC. REG. DONE?  
3056 013756 001367 BNE 1\$ ;BR IF NO  
3057 013760 005001 CLR R1 ;RESET SEC. REG. POINTER TO ZERO.  
3058 013762 005004 CLR R4 ;ZERO DATA COMPARE  
3059 013764 012737 013772 001220 MOV #2\$,LOCK ;SET FOR LOCK.  
3060 013772 110177 165376 2\$: MOV R1,@DVSRSH ;GET SEC. REG.  
3061 013776 011203 MOV (R2),R3 ;READ SEC. REG.  
3062 014000 020403 CMP R4,R3 ;R4=GOOD; R3=UNKNOWN  
3063 014002 001401 BEQ 3\$ ;BR IF ALL OK  
3064 014004 104002 HLT 2 ;SECONDARY REGISTER ADDRESSING ERROR  
3065 014006 104401 3\$: SCOP1 ;LOCK ON REG. (SW09=1)  
3066 014010 062704 010421 ADD #^B<0001000100010001>,R4  
3067 014014 005201 INC R1 ;UPDATE SEC REG POINTER  
3068 014016 022701 000020 CMP #16.,R1 ;ALL 16 LINES TESTED YET?  
3069 014022 001363 BNE 2\$ ;BR IF NO  
3070 014024 005200 INC R0 ;UPDATE LINE NO POINTER  
3071 014026 022700 000020 CMP #16.,R0 ;ALL LINES DONE??  
3072 014032 001333 BNE 65\$ ;BR IF NO  
3073 014034 104400 SCOPE ;SCOPE THE TEST

3074  
3075  
3076 :\*\*\*\*\* TEST 56 \*\*\*\*\*  
3077 :\*VERIFY NO LINE INTERACTION.  
3078 :\*THIS TEST VERIFIES THAT WRITING THE SECONDARY  
3079 :\*REGISTERS FOR ONE LINE DOES NOT INTERFER WITH  
3080 :\*THE SECONDARY REGISTERS OF ANOTHER LINE.  
3081 :\*\*\*\*\*  
3082

3083 ; TEST 56

3084 :-----  
3085 014036 012737 000056 001226 TST56: MOV #56,TSTNO

```

3086 014044 012737 000002 001222      MOV      #2,ICOUNT
3087 014052 012737 014434 001216      MOV      #TST57,NEXT
3088 014060 012737 014152 001220      MOV      #4$,LOCK
3089 014066 005000                      CLR      R0
3090 014070 005001                      CLR      R1
3091 014072 005004                      CLR      R4
3092 014074 013702 001376      MOV      DVSRA,R2
3093 014100 110077 165266      1$:     MOVB   R0,@DVSRS
3094 014104 110177 165264      2$:     MOVB   R1,@DVSRS
3095 014110 010412                      MOV      R4,(R2)
3096 014112 005201                      INC      R1
3097 014114 022701 000020      CMP      #16.,R1
3098 014120 001371                      BNE     2$
3099 014122 005001                      CLR      R1
3100 014124 062704 010421      ADD      #*B<0001000100010001>,R4
3101                      ;UPDATE DATA POINTER.
3102 014130 005200                      INC      R0
3103 014132 022700 000020      CMP      #16.,R0
3104 014136 001360                      BNF     1$
3105                      ;UPDATE LINE POINTER.
3106 014140 005000                      CLR      R0
3107 014142 005001                      CLR      R1
3108 014144 005004                      CLR      R4
3109 014146 110077 165220      3$:     MOVB   R0,@DVSRS
3110 014152 110177 165216      4$:     MOVB   R1,@DVSRS
3111 014156 011203                      MOV      (R2),R3
3112 014160 020403                      CMP      R4,R3
3113 014162 001401                      BEQ     5$
3114 014164 104002                      HLT     2
3115 014166 104401      5$:     SCOP1
3116 014170 005201                      INC      R1
3117 014172 022701 000020      CMP      #16.,R1
3118 014176 001365                      BNE     4$
3119 014200 062704 010421      ADD      #*B<0001000100010001>,R4
3120                      ;UPDATE DATA
3121 014204 005001                      CLR      R1
3122 014206 005200                      INC      R0
3123 014210 022700 000020      CMP      #16.,R0
3124 014214 001354                      BNE     3$
3125                      ;UPDATE LINE NUMBER POINTER.
3126                      ;ALL LINES DONE?
3127                      ;BR IF NO.
3128                      ;*PART 2
3129                      ;*FILL ALL RAMS WITH ALL 1'S AND THEN
3130                      ;*CLEAR JUST ONE BIT AT A TIME VERIFYING
3131                      ;*THAT ONLY THAT ONE BIT IS CLEAR AND THAT
3132                      ;*ALL OTHER RAMS STILL CONTAIN ALL 1'S.
3133                      ;*THERE SHOULD BE ONLY ONE BIT CLEARED AT
3134                      ;*ONE TIME.
3134 014216 005077 165162      CLR      @RESV16
3135 014222 005037 001220      CLR      LOCK
3136 014226 013702 001376      MAR17:  MOV      DVSRA,R2
3137 014232 005077 165134      CLR      @DVSRS
3138 014236 012712 177777      1$:     MOV      #*B<1111111111111111>,(R2)
3139                      ;PREPARE TO LOAD ALL 1'S INTO ALL RAM LOC.
3140 014242 002777 170361 165122      ADD      #*C<BIT11+BIT10+BIT9+BIT8+BIT3+BIT2+BIT1+BIT0>+BIT0,@DVSRS
3141                      ;UPDATE LINE AND SEC REG POINTERS.
    
```

```
3142 014250 001372          BNE      1$          ;BR IF NOT ALL DONE FILLING 1'S.
3143 014252 005000          CLR      R0          ;ZERO LINE # POINTER
3144 014254 005001          CLR      R1          ;ZERO SEC REG # POINTER
3145 014256 012704 177776    2$:      MOV      #^C<BIT0>,R4    ;SET INITIAL DATA
3146 014262 110077 165104    3$:      MOVVB   R0,@DVSRS      ;LOAD LINE #
3147 014266 110177 165102          MOVVB   R1,@DVSRSRSH    ;LOAD SEC REG #
3148 014272 010412          MOV      R4,(R2)      ;LOAD DATA
3149 014274 005077 165072          CLR      @DVSRS      ;ZERO POINTERS
3150 014300 022712 177777    100$:    CMP      #^B<1111111111111111>,(R2)
3151          ;VERIFY ONLY ONE LOC. HAS ONE BIT CLEARED
3152 014304 001417          BEQ      6$          ;BR IF LOC. OK
3153 014306 012777 177777 165070  MOV      #-1,@RESV16    ;SET 'LOC FOUND FLAG'.
3154 014314 011203          MOV      (R2),R3      ;SAVE DATA
3155 014316 120077 165050    CMPB    R0,@DVSRS      ;IS THIS THE RIGHT LINE?
3156 014322 001401          BEQ      4$          ;BR IF YES
3157 014324 104002          HLT      2           ;WRONG LINE HAS CLEARED BIT.
3158 014326 120177 165042    4$:      CMPB    R1,@DVSRSRSH    ;IS THIS THE RIGHT SEC REG?
3159 014332 001401          BEQ      5$          ;BR IF YES
3160 014334 104002          HLT      2           ;WRONG SEC REG HAS CLEARED BIT.
3161 014336 020403          5$:      CMP      R4,R3        ;IS THE ACTUAL DATA OK?
3162 014340 001401          BEQ      6$          ;BR IF YES
3163 014342 104002          HLT      2           ;ACTUAL DATA WAS WRONG.
3164 014344 062777 170361 165020 6$:      ADD      #^C<BIT11+BIT10+BIT9+BIT8+BIT3+BIT2+BIT1+BIT0>+BIT0,@DVSRS
3165 014352 001352          BNE      100$        ;BR IF NOT DONE.
3166 014354 005777 165024          TST     @RESV16      ;HAS A LOC BEEN FOUND?
3167 014360 001001          BNE      7$          ;BR IF YES
3168 014362 104000          HLT      0           ;NO LOC WAS FOUND WITH A ZERO BIT.
3169 014364 005077 165014    7$:      CLR      @RESV16      ;CLEAR 'LOC FOUND FLAG'.
3170 014370 000261          SEC          ;SHIFT IN A 1
3171 014372 006104          ROL      R4          ;CHANGE DATA PATTERN
3172 014374 103732          BCS      3$          ;DO IT ALL OVER AGAIN
3173 014376 110077 164770    MOVVB   R0,@DVSRS      ;LOAD LINE NO.
3174 014402 110177 164756    MOVVB   R1,@DVSRSRSH    ;LOAD SEC REG.
3175 014406 010412          MOV      R4,(R2)      ;PUT RAM LOC BACK TO ALL 1'S.
3176 014410 005201          INC      R1          ;UPDATE SEC REG #
3177 014412 022701 000020    CMP      #16.,R1      ;ALL SEC REG DONE?
3178 014416 001317          BNE      2$          ;BR IF NO
3179 014420 005001          CLR      R1          ;ZERO SEC REG POINTER
3180 014422 005200          INC      R0          ;UPDATE LINE POINTER
3181 014424 022700 000020    CMP      #16.,R0      ;ALL LINES DONE?
3182 014430 001312          BNE      2$          ;BR IF NO
3183 014432 104400          SCOPE          ;SCOPE THIS TEST.
```

```
3184
3185
3186          ;***** TEST 57 *****
3187          ;*MEMORY EXTENSION READ/WRITE TEST
3188          ;*VERIFY BITS 4 AND 5 OF EACH LINE
3189          ;*SECONDARY REGISTERS EXERCISED ARE:
3190          ;* 00 TX BUS ADDRESS (PRIMARY)
3191          ;* 02 TX BUS ADDRESS (SECONDARY)
3192          ;* 04 RX BUS ADDRESS
3193          ;* 10 TX TABLE BASE ADDRESS
3194          ;* 11 RX TABLE BASE ADDRESS
3195          ;*NOTE THAT ALL LINES (00-16) ARE EXERCISED.
3196          ;*****
3197
```

```

3198
3199
3200      : TEST 57
3201 014434 012737 000057 001226 TST57: MOV #57,TSTNO
3202 014442 012737 014630 001216      MOV #TST60,NEXT
3203 014450 012737 014510 001220      MOV #2$,LOCK
3204 014456 005000      CLR R0 ;R0=LINE NUMBER (START AT 0)
3205 014460 013702 001370      MOV DVLCR,R2 ;SET R2 =BASE ADDRESS(DVLCR)
3206 014464 012704 000060 1$: MOV #BIT5+BIT4,R4 ;R4=GOOD DATA (BOTH EA BITS SET AT START)
3207 014470 012705 000005      MOV #5,R5 ;R5 IS COUNTER OF SEC REGISTERS
3208 014474 012737 000004 001246      MOV #4,TEMP1 ;TEMP1 IS COUNTER OF COMB. OF EA BITS.
3209      ;EX: 11,10,01,00
3210 014502 012737 014622 001256      MOV #MEMEXT,TEMP5 ;TEMP5=SEC. REGISTER POINTER.
3211 014510 010477 164646 2$: MOV R4,@DVSCR ;LOAD DVSCR WITH EA BITS.
3212 014514 110077 164652      MOV R0,@DVSR ;SEL THE LINE NUMBER
3213 014520 117701 164532      MOV #TEMP5,R1 ;GET SEC REG.
3214 014524 110177 164644      MOV #R1,@DVSRSH ;SEL THE SEC. REGISTER
3215 014530 005077 164642      CLR @DVSR ;HIT THE SEC.REG. ACCESS REGISTER.
3216 014534 017703 164630      MOV @DVLCR,R3 ;SAVE THE DVLINE PARM. REG.
3217 014540 042703 177717      BIC #^C<BIT5+BIT4>,R3
3218      ;CLEAR ALL BUT BITS 5 AND 4.
3219 014544 020403      CMP R4,R3 ;ARE THE EA BITS GOOD
3220 014546 001401      BEQ 3$
3221 014550 104004      HLT 4
3222 014552 104401 3$: SCOP1 ;SW09=1?
3223 014554 005237 001256      INC TEMP5 ;POP POINTER
3224 014560 005305      DEC R5 ;ALL SEC REG DONE?
3225 014562 001352      BNE 2$ ;BR IF NO.
3226 014564 012737 014622 001256      MOV #MEMEXT,TEMP5 ;RESET POINTER
3227 014572 012705 000005      MOV #5,R5 ;RESET COUNTER
3228 014576 162704 000020      SUB #BIT4,R4 ;ADJUST FOR NEXT EA BIT PATTERN
3229 014602 005337 001246      DEC TEMP1 ;ALL PATERNS DONE?
3230 014606 001340      BNE 2$
3231 014610 005200      INC R0 ;UPDATE TO NEXT LINE
3232 014612 022700 000020      CMP #16.,R0 ;ALL LINES DONE
3233 014616 001322      BNE 1$ ;BR IF NO.
3234 014620 104400      SCOPE
3235      ;TABLE OF SECONDARY REGISTERS EXERCISERD....
3236 014622 000 MEMEXT: .BYTE 00
3237 014623 002      .BYTE 02
3238 014624 004      .BYTE 04
3239 014625 010      .BYTE 10
3240 014626 011      .BYTE 11
3241      .EVEN
3242
3243
3244      ;***** TEST 60 *****
3245      ;*INITIALIZATION TESTS
3246      ;*SET ALL POSSIBLE BITS IN ALL THE PRIMARY REGISTERS
3247      ;*AND VERIFY THAT ALL THE BITS ARE CLEARED
3248      ;*BY A BUS RESET
3249      ;:*****
3250
3251      : TEST 60
3252      :-----
3253 014630 012737 000060 001226 TST60: MOV #60,TSTNO
    
```



3254	014636	012737	015044	001216		MOV	#TST61,NEXT	
3255	014644	012737	000340	177776		MOV	#340,PS	:LOCK OUT INTERRUPTS.
3256	014652	013703	001362			MOV	DVSCR,R3	:SET REGISTER POINTER FOR LOADING
3257	014656	005077	164510			CLR	@DVSRS	:CLEAR LINE POINTER
3258	014662	005077	164510			CLR	@DVSRA	:CLEAR ACCESS REG.
3259	014666	012723	173777			MOV	#^C<BIT11>,(R3)+	
3260	014672	012702	000007			MOV	#7,R2	:SET ALL BITS BUT MSTCLR
3261	014676	012723	177777		1\$:	MOV	#-1,(R3)+	:LOAD ALL OTHER REGISTERS WITH ALL 1'S
3262	014702	005302				DEC	R2	:ALLREGISTERS LOADED?
3263	014704	001374				BNE	1\$	:BR IF NO
3264	014706	000005				RESET		:ISSUES A BUS INIT (RESET INSTR)
3265	014710	005200				INC	R0	:FLASH THE CPU LIGHTS!!!
3266	014712	013703	001362			MOV	DVSCR,R3	:SET REGISTER POINTER
3267	014716	005005				CLR	R5	:SET 'EXPECTED' FOR DVSCR
3268	014720	011304				MOV	(R3),R4	:READ THE DVSCR REG.
3269	014722	020504				CMP	R5,R4	:IS BIT8 ALONE SET?
3270	014724	001401				BEQ	2\$	:BR IF YES
3271	014726	104003				HLT	3	:DVSCR HAS WRONG DATA>
3272	014730	005723			2\$:	TST	(R3)+	:POP POINTER TO DVRIC
3273	014732	005005				CLR	R5	:SET EXPECTED TO ZERO
3274	014734	011304				MOV	(R3),R4	:DVRIC (EXPECT ALL 0'S)
3275	014736	001401				BEQ	3\$	:BR IF OK
3276	014740	104003				HLT	3	:DVRIC NO ALL 0'S
3277	014742	005723			3\$:	TST	(R3)+	:POP POINTER TO DVLCR REG
3278	014744	011304				MOV	(R3),R4	:DVLCR (READ DVLCR INTO R4)
3279	014746	042704	000063			BIC	#BIT5+BIT4+BIT1+BIT0,R4	
3280	014752	020504				CMP	R5,R4	:DISREGUARD BR TEST POINTS AND MEM EXT BITS.
3281	014754	001401				BEQ	4\$	:DVLCR OK?
3282	014756	104003				HLT	3	:DVLCR INCORECT (DISREGUARD BIT5,4,1,0)
3283	014760	005723			4\$:	TST	(R3)+	:POP POINTER TO DVSRS REG
3284	014762	011304				MOV	(R3),R4	:DVSRS (EXPECT ALL 0'S)
3285	014764	001401				BEQ	5\$	:BR IF OK
3286	014766	104003				HLT	3	:DVSRS REG NOT ALL ZEROS
3287	014770	005723			5\$:	TST	(R3)+	:POP POINTER TO DVSRA REG
3288	014772	011304				MOV	(R3),R4	:DVSRA (EXPECT ALL 0'S)
3289	014774	001401				BEQ	6\$	:BR IF GOOD
3290	014776	104003				HLT	3	:DVSRA NOT ALL 0'S
3291	015000	005723			6\$:	TST	(R3)+	:POP POINTER TO DVSFR
3292	015002	011304				MOV	(R3),R4	:DVSFR (EXPECT ALL 1'S (THATS RIGHT))
3293	015004	012705	177777			MOV	#177777,R5	:SET EXPECTED
3294	015010	020504				CMP	R5,R4	:EXPECETD =FOUND?
3295	015012	001401				BEQ	7\$	:BR IF YES
3296	015014	104003				HLT	3	:DVSFR NOT ALL 1'S
3297	015016	005723			7\$:	TST	(R3)+	:POP POINTER TO DVNSR REG
3298	015020	005713				TST	(R3)	:DVNSR S/B PLUS (15=0)
3299	015022	100001				BPL	64\$	
3300	015024	104000				HLT	0	
3301	015026	005723			64\$:	TST	(R3)+	:POP POINTER TO RESV16 REG
3302	015030	011304				MOV	(R3),R4	:RESV16 (EXPECT ALL 0'S)
3303	015032	005005				CLR	R5	:SET EXPECTED TO 0'S
3304	015034	020504				CMP	R5,R4	:WELL DOES IT =1'S?
3305	015036	001401				BEQ	8\$	:BR IF OK
3306	015040	104003				HLT	3	:RESV16 NOT ALL 0'S
3307	015042	104400			8\$:	SCOPE		:SCOPE THIS TEST;
3308								
3309								

```

3310 :***** TEST 61 *****
3311 :*INITIALIZATION TESTS
3312 :*SET ALL POSSIBLE BITS IN ALL THE PRIMARY REGISTERS
3313 :*AND VERIFY THAT ALL THE BITS ARE CLEARED
3314 :*BY A MASTER CLEAR
3315 :*****
3316
3317 : TEST 61
3318 -----
3319 015044 012737 000061 001226 TST61: MOV #61,TSTNO
3320 015052 012737 015262 001216 MOV #TST62,NEXT
3321 015060 012737 000340 177776 MOV #340,PS
3322 015066 013703 001362 MOV DVSCR,R3 ;LOCK OUT INTERRUPTS.
3323 015072 005077 164274 CLR @DVSRS ;SET REGISTER POINTER FOR LOADING
3324 015076 005077 164274 CLR @DVSRA ;CLEAR LINE POINTER
3325 015102 012723 173777 MOV #^C<BIT11>,(R3)+ ;CLEAR ACCESS REG.
3326 015106 012702 000007 MOV #7,R2 ;SET ALL BITS BUT MSTCLR
3327 015112 012723 177777 1< MOV #-1,(R3)+ ;LOAD ALL OTHER REGISTERS WITH ALL 1'S
3328 015116 005302 DEC R2 ;ALLREGISTERS LOADED?
3329 015120 001374 BNE 1$ ;BR IF NO
3330 015122 052777 004000 164232 BIS #MRESET,@DVSCR ;ISSUE A 'MASTER CLEAR'
3331 015130 013703 001362 MOV DVSCR,R3 ;SET REGISTER POINTER
3332 015134 005005 CLR R5 ;SET 'EXPECTED' FOR DVSCR
3333 015136 011304 MOV (R3),R4 ;READ THE DVSCR REG.
3334 015140 020504 CMP R5,R4 ;IS BIT8 ALONE SET?
3335 015142 001401 BEQ 2$ ;BR IF YES
3336 015144 104003 HLT 3 ;DVSCR HAS WRONG DATA>
3337 015146 005723 2$: TST (R3)+ ;POP POINTER TO DVRIC
3338 015150 005005 CLR R5 ;SET EXPECTED TO ZERO
3339 015152 011304 MOV (R3),R4 ;DVRIC (EXPECT ALL 0'S)
3340 015154 001401 BEQ 3$ ;BR IF OK
3341 015156 104003 HLT 3 ;DVRIC NO ALL 0'S
3342 015160 005723 3$: TST (R3)+ ;POP POINTER TO DVLCR REG
3343 015162 011304 MOV (R3),R4 ;DVLCR (READ DVLCR INTO R4)
3344 015164 042704 000063 BIC #BIT5+BIT4+BIT1+BIT0,R4
3345 015170 020504 CMP R5,R4 ;DISREGUARD BR TEST POINTS AND MEM EXT BITS.
3346 015172 001401 BEQ 4$ ;DVLCR OK?
3347 015174 104003 HLT 3 ;DVLCR INCORECT (DISREGUARD BITS,4,1,0)
3348 015176 005723 4$: TST (R3)+ ;POP POINTER TO DVSRS REG
3349 015200 011304 MOV (R3),R4 ;DVSRS (EXPECT ALL 0'S)
3350 015202 001401 BEQ 5$ ;BR IF OK
3351 015204 104003 HLT 3 ;DVSRS REG NOT ALL ZEROS
3352 015206 005723 5$: TST (R3)+ ;POP POINTER TO DVSRA REG
3353 015210 011304 MOV (R3),R4 ;DVSRA (EXPECT ALL 0'S)
3354 015212 001401 BEQ 6$ ;BR IF GOOD
3355 015214 104003 HLT 3 ;DVSRA NOT ALL 0'S
3356 015216 005723 6$: TST (R3)+ ;POP POINTER TO DVSFR
3357 015220 011304 MOV (R3),R4 ;DVSFR (EXPECT ALL 1'S (THAT'S RIGHT))
3358 015222 012705 177777 MOV #177777,R5 ;SET EXPECTED
3359 015226 020504 CMP R5,R4 ;EXPECETD =FOUND?
3360 015230 001401 BEQ 7$ ;BR IF YES
3361 015232 104003 HLT 3 ;DVSFR NOT ALL 1'S
3362 015234 005723 7$: TST (R3)+ ;POP POINTER TO DVNSR REG
3363 015236 005713 TST (R3) ;DVNSR S/B PLUS (15=0)
3364 015240 100001 BPL 64$
3365 015242 104000 HLT 0

```

```

3366 015244 005723      64$:  TST      (R3)+      ;POP POINTER TO RESV16 REG
3367 015246 011304      MOV      (R3),R4        ;RESV16 (EXPECT ALL 0'S)
3368 015250 005005      CLR      R5             ;SET EXPECTED TO 0'S
3369 015252 020504      CMP      R5,R4         ;WELL DOES IT =1'S?
3370 015254 001401      BEQ     8$             ;BR IF OK
3371 015256 104003      HLT     3              ;RESV16 NOT ALL 0'S
3372 015260 104400      8$:   SCOPE           ;SCOPE THIS TEST;
    
```

3373  
3374  
3375  
3376  
3377  
3378  
3379  
3380  
3381  
3382  
3383  
3384  
3385  
3386  
3387  
3388  
3389  
3390  
3391  
3392  
3393  
3394  
3395  
3396  
3397  
3398  
3399  
3400  
3401  
3402  
3403  
3404  
3405  
3406  
3407  
3408  
3409  
3410  
3411  
3412  
3413  
3414  
3415  
3416  
3417  
3418  
3419  
3420  
3421

```

:***** TEST 62 *****
:*ATTACK OF THE SPECIAL FUNCTIONS REGISTER.
:*BEGIN CHECK OF THE DVSFR.
:*SUMMARY OF PROC. INSTRUCTIONS
    
```

```

:*BIT14 BIT13 BIT12 INSTRUCTION
:* 0      0      0  BRANCH 'A'
:* 0      0      1  ALU OPERATION
:* 0      1      0  RAM OPERATION
:* 0      1      1  DATA TRANSFER
:* 1      0      0  NPR OPERATION
:* 1      0      1  SET/CLEAR OPERATION
:* 1      1      0  BCC CALCULATION
:* 1      1      1  BRANCH 'B'
    
```

```

:***** TEST 62 *****
:*VERIFY THAT 'ROM STEP'
:*IS SELF-CLEARING AND THAT
:*THE DATA IN THE DVSFR
:*IS CHANGED WHEN THE ROM IS STEPPED.
    
```

: TEST 62

```

1$T62:  MOV      #62,TSTNO
        MOV      #TST63,NEXT
        MSTCLR           ;CLEAR ALL THE DV11
        MOV      #BIT3,@DVSCR ;SET SOURCE SEL
        MOV      #S.C,R5 ;PUT INSTR INTO DVSFR
        MOV      R5,@DVSFR
        CMP      @DVSFR,R5 ;WAS THE DVSFR REALLY LOADED?
        BEQ     1$      ;BR IF YES
        HLT     ;BAD DVSFR
1$:     BIC      #BIT3,@DVSCR ;CLEAR SOURCE SEL.
        ROMCLK ;ISSUE A ROM CLOCK
        NOP      ;WAIST AN INTRUSTION TIME
        BIT      #BIT1,@DVSCR ;DID CLK BIT CLEAR BY IT SELF?
        BEQ     2$      ;BR IF CLK GONE
        HLT     ;BIT 1 OF DVSCR (ROM CLK) NOT ZERO
2$:     CMP      R5,@DVSFR ;HAS DATA IN DVSFR CHANGED?
        BNE     3$      ;BR IF YES
        HLT     ;DATA NOT CHANGED (DID CLK REALLY CLK??)
3$:     SCOPE           ;SCOPE THIS TEST.
    
```

:\*\*\*\*\* TEST 63 \*\*\*\*\*

X22  
X23  
X24  
X25  
X26  
X27  
X28  
X29  
X30  
X31  
X32  
X33  
X34  
X35  
X36  
X37  
X38  
X39  
X40  
X41  
X42  
X43  
X44  
X45  
X46  
X47  
X48  
X49  
X50  
X51  
X52  
X53  
X54  
X55  
X56  
X57  
X58  
X59  
X60  
X61  
X62  
X63  
X64  
X65  
X66  
X67  
X68  
X69  
X70  
X71  
X72  
X73  
X74  
X75  
X76  
X77

```

: *BASIC TEST OF DVSFR
: *TEST THAT 'BRANCH A' INSTRUCTION.
: *POINTS TESTED:
:
: *BIT11 BIT10 BIT09 BIT08 BR 'A' BR 'B' FUNCTION
: * 0      0      0      1      L      H      PLUS 3 VOLTS
: * 0      0      0      1      L,H     H,H     DVSCROB (=0,=1)
: * 0      1      1      1      H      H      NPR SILO NOT AVAIL.
: *      1      1      1      H      H      SILO FULL
: * 0      1      1      0      H      H      NXM
  
```

.....

: TEST 63

```

TST63:  MOV #63,TSTNO
        MOV #TST64,NEXT
        MSTCLR
        MOV DVSFR,R0
        MOV DVLCR,R3
        MOV #BIT8+BIT3,ADVSCR
18:     MOV #BIT8,(R0)
        MOV (R0),R2
        MOV #BIT1,R5
        MOV (R3),R4
        BIC #^C<BIT1+BIT0>,R4
        (MP R5,R4
        BEQ 28
        HLT 6
        MOV #BIT10+BIT8,(R0)
        MOV (R0),R2
        MOV #BIT1,R5
        MOV (R3),R4
        BIC #^C<BIT1+BIT0>,R4
        (MP R5,R4
        BEQ 38
        HLT 6
        BIC #BIT8,ADVSCR
        MOV (R0),R2
        MOV #BIT1+BIT0,R5
        MOV (R3),R4
        BIC #^C<BIT1+BIT0>,R4
        (MP R5,R4
        BEQ 48
        HLT 6
        MOV #BIT10+BIT9+BIT8,(R0)
        MOV (R0),R2
        MOV #BIT1+BIT0,R5
        MOV (R3),R4
        BIC #^C<BIT1+BIT0>,R4
        (MP R5,R4
        BEQ 58
  
```

001226  
001216  
  
163742  
18:  
  
28:  
  
38:  
  
48:  
  
58:

```

015364 012737 000063 001226
015372 012737 015630 001216
015400 104412
015402 013700 001400
015406 013703 001370
015412 012777 000410 163742
015420 012710 000400
015424 011002
015426 012705 000002
015432 011304
015434 042704 177774
015440 020504
015442 001401
015444 104006
015446 012710 002400 28:
015452 011002
015454 012705 000002
015460 011304
015462 042704 177774
015466 020504
015470 001401
015472 104006
015474 042777 000400 163660 38:
015502 011002
015504 012705 000003
015510 011304
015512 042704 177774
015516 020504
015520 001401
015522 104006
015524 012710 003400 48:
015530 011002
015532 012705 000003
015536 011304
015540 042704 177774
015544 020504
015546 001401
  
```

```
3478 015550 104006          HLT      6          :BR POINT WRONG
3479 015552 012710 007400 58:  MOV     #BIT11+BIT10+BIT9+BIT8,(R0)
3480 015556 011002          MOV     (R0),R2     :READ DV5FR FOR PRINTOUT
3481 015560 012705 000003  MOV     #BIT1+BIT0,R5 :SET EXPECTED RESULTS
3482 015564 011304          MOV     (R3),R4     :READ DVLCR INTO R4
3483 015566 042704 177774  BIC     #^C<BIT1+BIT0>,R4
3484                               :CLEAR UNWANTED BITS.
3485 015572 020504          CMP     R5,R4       :EXPECTED = FOUND??
3486 015574 001401          BEQ     68          :BR IF OK
3487 015576 104006          HLT     6          :BR POINT WRONG
3488 015600 012710 003000 68:  MOV     #BIT10+BIT9,(R0) :NXP
3489 015604 011002          MOV     (R0),R2     :READ DV5FR FOR PRINTOUT
3490 015606 012705 000003  MOV     #BIT1+BIT0,R5 :SET EXPECTED RESULTS
3491 015612 011304          MOV     (R3),R4     :READ DVLCR INTO R4
3492 015614 042704 177774  BIC     #^C<BIT1+BIT0>,R4
3493                               :CLEAR UNWANTED BITS.
3494 015620 020504          CMP     R5,R4       :EXPECTED = FOUND??
3495 015622 001401          BEQ     78          :BR IF OK
3496 015624 104006          HLT     6          :BR POINT WRONG
3497 015626 104400          78:  SCOPE          :SCOPE TEST
```

```
***** TEST 64 *****
:*TEST OF BRANCH B*
:*TEST THAT POINT 16 (GROUND)
:*MAKES LCR BIT1-1 AND BIT0 1.
*****
```

: TEST 64

```
3507 015630 012737 000064 001226 TST64: MOV     #64,TSTNO
3508 015636 012737 015706 001216  MOV     #TST65,NEXT
3509 015644 104412          MSTCLR          :CLEAR DV11
3510 015646 012777 000010 163506  MOV     #BIT3,@DV5CR :SET SOURCE SEL
3511 015654 012777 077400 163516  MOV     #BRB+BIT11+BIT10+BIT9+BIT8,@DV5FR
3512                               :BR-B 'GROUND'
3513 015662 017702 163512          MOV     @DV5FR,R2   :READ DV5FR INTO R2
3514 015666 012705 000003  MOV     #BIT1+BIT0,R5 :SET EXPECTED RESULTS
3515 015672 017704 163472  MOV     @DVLCR,R4    :READ REAL RESULTS
3516 015676 020504          CMP     R5,R4       :SAME??
3517 015700 001401          BEQ     18          :
3518 015702 104006          HLT     6          :BR TEST POINT WRONG
3519 015704 104400          18:  SCOPE          :SCOPE THIS TEST
```

```
***** TEST 65 *****
:*TEST OF BRANCH B
:*CHECKING 'DEFAULT' STATES OF THE DV11 SIGNALS.
:*BIT11 BIT10 BIT09 BIT08 FUNCTION
:* 1 0 0 0 DATA NOT AVAIL.
:* 1 0 0 1 REQUEST BUS
:* 1 0 1 0 MEMORY PARITY ERROR
:* 1 1 1 1 WRITE INHIBIT
*****
```

: TEST 65

```
3533 015706 012737 000065 001226 TST65: MOV     #65,TSTNO
```

```
3534 015714 012737 016074 001216      MOV      #TST66,NEXT
3535 015722 104412                      MSTCLR
3536 015724 013700 001400              MOV      DVSFR,R0
3537 015730 013703 001370              MOV      DVLCR,R3
3538 015734 012777 000010 163420      MOV      #BIT3,DVSCR
3539 015742 012710 074000 1$:      MOV      #BRB+BIT11,(R0)
3540 015746 011002                      MOV      (R0),R2      ;READ DVSFR FOR PRINTOUT
3541 015750 012705 000003              MOV      #BIT1+BIT0,R5 ;SET EXPECTED RESULTS
3542 015754 011304                      MOV      (R3),R4      ;READ DVLCR INTO R4
3543 015756 042704 177774              BIC      #^C<BIT1+BIT0>,R4
3544                                ;CLEAR UNWANTED BITS.
3545 015762 020504                      CMP      R4           ;EXPECTED = FOUND??
3546 015764 001401                      BEQ      2$          ;BR IF OK
3547 015766 104006                      HLT      6           ;BR POINT WRONG
3548 015770 012710 074400 2$:      MOV      #BRB+BIT11+BIT8,(R0)
3549 015774 011002                      MOV      (R0),R2      ;READ DVSFR FOR PRINTOUT
3550 015776 012705 000003              MOV      #BIT1+BIT0,R5 ;SET EXPECTED RESULTS
3551 016002 011304                      MOV      (R3),R4      ;READ DVLCR INTO R4
3552 016004 042704 177774              BIC      #^C<BIT1+BIT0>,R4
3553                                ;CLEAR UNWANTED BITS.
3554 016010 020504                      CMP      R5,R4       ;EXPECTED = FOUND??
3555 016012 001401                      BEQ      3$          ;BR IF OK
3556 016014 104006                      HLT      6           ;BR POINT WRONG
3557 016016 012710 075000 3$:      MOV      #BRB+BIT11+BIT9,(R0)
3558 016022 011002                      MOV      (R0),R2      ;READ DVSFR FOR PRINTOUT
3559 016024 012705 000003              MOV      #BIT1+BIT0,R5 ;SET EXPECTED RESULTS
3560 016030 011304                      MOV      (R3),R4      ;READ DVLCR INTO R4
3561 016032 042704 177774              BIC      #^C<BIT1+BIT0>,R4
3562                                ;CLEAR UNWANTED BITS.
3563 016036 020504                      CMP      R5,R4       ;EXPECTED = FOUND??
3564 016040 001401                      BEQ      4$          ;BR IF OK
3565 016042 104006                      HLT      6           ;BR POINT WRONG
3566 016044 012710 077000 4$:      MOV      #BRB+BIT11+BIT10+BIT9,(R0)
3567 016050 011002                      MOV      (R0),R2      ;READ DVSFR FOR PRINTOUT
3568 016052 012705 000003              MOV      #BIT1+BIT0,R5 ;SET EXPECTED RESULTS
3569 016056 011304                      MOV      (R3),R4      ;READ DVLCR INTO R4
3570 016060 042704 177774              BIC      #^C<BIT1+BIT0>,R4
3571                                ;CLEAR UNWANTED BITS.
3572 016064 020504                      CMP      R5,R4       ;EXPECTED = FOUND??
3573 016066 001401                      BEQ      5$          ;BR IF OK
3574 016070 104006                      HLT      6           ;BR POINT WRONG
3575 016072 104400 5$:      SCOPE
```

```
3576
3577
3578
3579
3580
3581
3582
3583
3584
3585
3586
3587
3588
3589
; TEST 66
;-----
:***** TEST 66 *****
: *BASIC TEST OF THE
: *SET/CLEAR INSTRUCTION.
: *TEST THAT THE SET/CLEAR CAN DO:
: *CLEAR DVSCR 08
: *SET DVSCR10
: *SET RECEIVER INTERRUPT (DVSCR07)
:*****
```

```

3590 016074 012737 000066 001226 TST66: MOV #66,TSTNO
3591 016102 012737 016266 001216 MOV #TST67,NEXT
3592 016110 012737 016140 001220 MOV #1$,LOCK
3593 016116 104412 MSTCLR ;CLEAR DV11
3594 016120 052777 000400 163234 BIS #BIT8,@DVSCR ;SET BIT8.
3595 016126 052777 000010 163226 BIS #BIT3,@DVSCR ;SET SOURCE SEL.
3596 016134 013700 001400 MOV DVSFR,R0 ;SET DVSFR POINTER ADDRESS IN R0
3597 016140 012710 050016 1$: MOV #S.C+BIT3+BIT2+BIT1,(R0)
3598 016144 012705 000010 MOV #BIT3,R5 ;DO SET/CLEAR -CLEAR BIT 8 OF DVSCR
3599 016150 011002 MOV (R0),R2 ;SAVE DVSFR CONTENTS FOR ERROR PRINT OUT IF NECESSARY
3600 016152 104415 ROMCLK ;CYCLE THE ROM
3601 016154 017704 163202 MOV @DVSCR,R4 ;READ DVSCR INTO 'FOUND LOC.
3602 016160 020504 CMP R5,R4 ;WAS THE ROM INSTR EXECUTED?
3603 016162 001401 BEQ 64$ ;BR IF DVSCR OK
3604 016164 104006 HLT 6 ;ROM FAILED TO EXECUTE
3605 016166 104401 64$: SCOP1 ;LOCK ON THIS SUB-TEST? SW09-1?
3606 016170 012737 016176 001220 MOV #3$,LOCK ;SET FOR RETURN IF SW09=1
3607 016176 052705 000200 3$: BIS #BIT7,R5 ;SET EXPECTED (SCR BIT 7-1)
3608 016202 012710 050013 MOV #S.C+BIT3+BIT1+BIT0,(R0)
3609 016206 011002 MOV (R0),R2 ;SAVE DVSFR CONTENTS FOR ERROR PRINT OUT IF NECESSARY
3610 016210 104415 ROMCLK ;CYCLE THE ROM
3611 016212 017704 163144 MOV @DVSCR,R4 ;READ DVSCR INTO 'FOUND LOC.
3612 016216 020504 CMP R5,R4 ;WAS THE ROM INSTR EXECUTED?
3613 016220 001401 BEQ 65$ ;BR IF DVSCR OK
3614 016222 104006 HLT 6 ;ROM FAILED TO EXECUTE
3615 016224 104401 65$: SCOP1 ;LOCK ON THIS SUB-TEST? SW09 1?
3616 016226 012737 016234 001220 MOV #4$,LOCK
3617 016234 052705 002000 4$: BIS #BIT10,R5 ;ALTER EXPECTED ADDRESS
3618 016240 012710 050012 MOV #S.C+BIT3+BIT1,(R0)
3619 016244 011002 MOV (R0),R2 ;DO A SET/CLEAR SET DVSCR BIT 10
3620 016246 104415 ROMCLK ;SAVE DVSFR CONTENTS FOR ERROR PRINT OUT IF NECESSARY
3621 016250 017704 163106 MOV @DVSCR,R4 ;CYCLE THE ROM
3622 016254 020504 CMP R5,R4 ;READ DVSCR INTO 'FOUND LOC.
3623 016256 001401 BEQ 66$ ;WAS THE ROM INSTR EXECUTED?
3624 016260 104006 HLT 6 ;BR IF DVSCR OK
3625 016262 104401 66$: SCOP1 ;ROM FAILED TO EXECUTE
3626 016264 104400 66$: SCOP1 ;LOCK ON THIS SUB-TEST? SW09 1?
3627 016264 104400 SCOPE ;SCOPE THIS TEST
3628
3629
3630
3631
3632
3633
3634
3635
3636
3637
3638
3639
3640
3641
3642
3643
3644
3645
  
```

```

:***** TEST 67 *****
:*BASIC TEST OF THE
:*SET/CLEAR INSTRUCTION.
:*TEST THAT THE SET/CLEAR CAN:
:*
:*BIT 14 BIT12 BIT03 BIT02 BIT01 BIT00 FUNCTION
:* 1 1 1 0 0 0 SET RICR 15
:* 1 1 1 0 0 1 SET RICR 14
:* 1 1 1 1 0 0 SET RICR 13
:* 1 1 1 1 0 1 SET RICR 12
:*
:*****
  
```

: TEST 67

```

3645 016266 012737 000067 001226 TST67: MOV #67,TSTNO
  
```

```
3646 016274 012737 016510 001216      MOV      #TST70,NEXT
3647 016302 104412                      MSTCLR
3648 016304 013700 001400              MOV      DVSFR,R0
3649 016310 012777 000010 163044      MOV      #BIT3,@DVSCR
3650 016316 012705 100000              MOV      #BIT15,R5
3651 016322 012710 050010              MOV      #S.C+BIT3,(R0)
3652
3653 016326 011002                      MOV      (R0),R2
3654 016330 104415                      ROMCLK
3655 016332 017704 163030              MOV      @DVRIC,R4
3656 016336 020504                      CMP      R5,R4
3657 016340 001401                      BEQ      64$
3658 016342 104006                      HLT      6
3659 016344 104401                      64$:   SCOPE
3660 016346 104412                      MSTCLR
3661 016350 012777 000010 163004      MOV      #BIT3,@DVSCR
3662 016356 012705 040000              MOV      #BIT14,R5
3663 016362 012710 050011              MOV      #S.C+BIT3+BIT0,(R0)
3664
3665 016366 011002                      MOV      (R0),R2
3666 016370 104415                      ROMCLK
3667 016372 017704 162770              MOV      @DVRIC,R4
3668 016376 020504                      CMP      R5,R4
3669 016400 001401                      BEQ      65$
3670 016402 104006                      HLT      6
3671 016404 104401                      65$:   SCOPE
3672 016406 104412                      MSTCLR
3673 016410 012777 000010 162744      MOV      #BIT3,@DVSCR
3674 016416 012705 020000              MOV      #BIT13,R5
3675 016422 012710 050014              MOV      #S.C+BIT3+BIT2,(R0)
3676
3677 016426 011002                      MOV      (R0),R2
3678 016430 104415                      ROMCLK
3679 016432 017704 162730              MOV      @DVRIC,R4
3680 016436 020504                      CMP      R5,R4
3681 016440 001401                      BEQ      66$
3682 016442 104006                      HLT      6
3683 016444 104401                      66$:   SCOPE
3684 016446 104412                      MSTCLR
3685 016450 012777 000010 162704      MOV      #BIT3,@DVSCR
3686 016456 012705 010000              MOV      #BIT12,R5
3687 016462 012710 050015              MOV      #S.C+BIT3+BIT2+BIT0,(R0)
3688
3689 016466 011002                      MOV      (R0),R2
3690 016470 104415                      ROMCLK
3691 016472 017704 162670              MOV      @DVRIC,R4
3692 016476 020504                      CMP      R5,R4
3693 016500 001401                      BEQ      67$
3694 016502 104006                      HLT      6
3695 016504 104401                      67$:   SCOPE
3696 016506 104400                      SCOPE
3697
3698
3699
3700
3701
```

\*\*\*\*\* TEST 70 \*\*\*\*\*  
:\*BASIC TEST OF DVSFR.  
:\*TEST OF "SET/CLEAR" AND



```

3702                                     :*'BRANCH A' AND 'BRANCH B' FUNCTIONS.
3703                                     :*****
3704
3705                                     : TEST 70
3706                                     :-----
3707 016510 012737 000070 001226 TST70: MOV #70,TSTNO
3708 016516 012737 017172 001216      MOV #TST71,NEXT
3709 016524 012737 016546 001220      MOV #1$,LOCK
3710 016532 104412                MSTCLR ;CLEAR DV11
3711 016534 012777 000010 162620      MOV #BIT3,ADVSCR ;SET SOURCE SELECT
3712 016542 013700 001400                MOV DVSFR,R0 ;SET DVSFR POINTER INTO R0
3713
3714                                     :*TEST SET/CLEAR FUNCTION
3715                                     :*FOR ALU BIT02
3716 016546 004237 017136 1$: JSR R2,10$ ;GOTO SUBROUTINE.
3717 016552 000024                BIT4+BIT2 ;POINT SET/CLEARED
3718 016554 006000                BIT11+BIT10 ;BR TEST POINT
3719 016556 000002                BIT1 ;EXPECTED RESULTS IN DVLCR
3720 016560 004237 017136      JSR R2,10$ ;GOSUB
3721 016564 000025                BIT4+BIT2+BIT10 ;POINT SET/CLEARED
3722 016566 006000                BIT11+BIT10 ;BR TEST POINT
3723 016570 000003                BIT1+BIT0 ;EXPECTED RESUTS IN DVLCR
3724 016572 104401                SCOP1 ;SW09=1??
3725
3726                                     :*TEST SET/CLEAR FUNCTION
3727                                     :*FOR RAM OUTPUT BIT00
3728 016574 012737 016602 001220      MOV #2$,LOCK ;SET RETURN IF SW09 1
3729 016602 004237 017136 2$: JSR R2,10$ ;GOTO THE SUBROUTINE
3730 016606 000047                BIT5+BIT2+BIT1+BIT0 ;POINT SET/CLEARED [SET]
3731 016610 070000                BRB ;BR TEST POINT
3732 016612 000001                BIT0 ;DVLCR EXPECTED
3733 016614 004237 017136      JSR R2,10$ ;GOTO SUB ROUTINE
3734 016620 000043                BIT5+BIT1+BIT0 ;POINT SET/CLEARED [CLEARED]
3735 016622 070000                BRB ;BR TEST POINT
3736 016624 000003                BIT1+BIT0 ;EXPECTED RESULTS
3737 016626 104401 65$: SCOP1 ;SWR 09=1?
3738
3739                                     :*TEST SET/CLEAR FUNCTION
3740                                     :*FOR RAM OUTPUT BIT01
3741 016630 012737 016636 001220      MOV #3$,LOCK ;SET RETURN IF SW09=1
3742 016636 004237 017136 3$: JSR R2,10$ ;GOTO THE SUBROUTINE
3743 016642 000045                BIT5+BIT2+BIT0 ;POINT SET/CLEARED [SET]
3744 016644 070400                BRB+BIT8 ;BR TEST POINT
3745 016646 000001                BIT0 ;DVLCR EXPECTED
3746 016650 004237 017136      JSR R2,10$ ;GOTO SUB ROUTINE
3747 016654 000041                BIT5+BIT0 ;POINT SET/CLEARED [CLEARED]
3748 016656 070400                BRB+BIT8 ;BR TEST POINT
3749 016660 000003                BIT1+BIT0 ;EXPECTED RESULTS
3750 016662 104401 67$: SCOP1 ;SWR 09=1?
3751
3752                                     :*TEST SET/CLEAR FUNCTION
3753                                     :*FOR RAM OUTPUT BIT02
3754 016664 012737 016672 001220      MOV #4$,LOCK ;SET RETURN IF SW09 1
3755 016672 004237 017136 4$: JSR R2,10$ ;GOTO THE SUBROUTINE
3756 016676 000046                BIT5+BIT2+BIT1 ;POINT SET/CLEARED [SET]
3757 016700 071000                BRB+BIT9 ;BR TEST POINT
    
```

```

3758 016702 000001          BIT0          ;DVLCR EXPECTED
3759 016704 004237 017136  JSR      R2,10$    ;GOTO SUB ROUTINE
3760 016710 000042          BITS+BIT1    ;POINT SET/CLEARED [CLEARED]
3761 016712 071000          BRB+BIT9    ;BR TEST POINT
3762 016714 000003          BIT1+BIT0   ;EXPECTED RESULTS
3763 016716 104401          SCOPI      ;SWR 09=1?
3764
3765
3766          ;*TEST SET/CLEAR FUNCTION
3767 016720 012737 016726 001220  MOV     #6$,LOCK   ;SET RETURN IF SW09=1
3768 016726 004237 017136 6$: JSR     R2,10$    ;GOTO THE SUBROUTINE
3769 016732 000044          BITS+BIT2    ;POINT SET/CLEARED [SET]
3770 016734 071400          BRB+BIT9+BIT8 ;BR TEST POINT
3771 016736 000001          BIT0        ;DVLCR EXPECTED
3772 016740 004237 017136  JSR     R2,10$    ;GOTO SUB ROUTINE
3773 016744 000040          BITS        ;POINT SET/CLEARED [CLEARED]
3774 016746 071400          BRB+BIT9+BIT8 ;BR TEST POINT
3775 016750 000003          BIT1+BIT0   ;EXPECTED RESULTS
3776 016752 104401          SCOPI      ;SWR 09=1?
3777
3778          ;*TEST SET/CLEAR FUNCTION
3779          ;*FOR RAM OUTPUT BIT03
3780 016754 012737 016762 001220  MOV     #7$,LOCK   ;SET RETURN IF SW09 1
3781 016762 004237 017136 7$: JSR     R2,10$    ;GOTO THE SUBROUTINE
3782 016766 000207          BIT7+BIT2+BIT1+BIT0 ;POINT SET/CLEARED [SET]
3783 016770 072000          BRB+BIT10   ;BR TEST POINT
3784 016772 000001          BIT0        ;DVLCR EXPECTED
3785 016774 004237 017136  JSR     R2,10$    ;GOTO SUB ROUTINE
3786 017000 000203          BIT7+BIT1+BIT0 ;POINT SET/CLEARED [CLEARED]
3787 017002 072000          BRB+BIT10   ;BR TEST POINT
3788 017004 000003          BIT1+BIT0   ;EXPECTED RESULTS
3789 017006 104401          SCOPI      ;SWR 09=1?
3790
3791          ;*TEST SET/CLEAR FUNCTION
3792          ;*FOR RAM OUTPUT BIT05
3793 017010 012737 017016 001220  MOV     #8$,LOCK   ;SET RETURN IF SW09 1
3794 017016 004237 017136 8$: JSR     R2,10$    ;GOTO THE SUBROUTINE
3795 017022 000205          BIT7+BIT2+BIT0   ;POINT SET/CLEARED [SET]
3796 017024 072400          BRB+BIT10+BIT8 ;BR TEST POINT
3797 017026 000001          BIT0        ;DVLCR EXPECTED
3798 017030 004237 017136  JSR     R2,10$    ;GOTO SUB ROUTINE
3799 017034 000201          BIT7+BIT0     ;POINT SET/CLEARED [CLEARED]
3800 017036 072400          BRB+BIT10+BIT8 ;BR TEST POINT
3801 017040 000003          BIT1+BIT0   ;EXPECTED RESULTS
3802 017042 104401          SCOPI      ;SWR 09=1?
3803
3804          ;*TEST SET/CLEAR FUNCTION
3805          ;*FOR RAM OUTPUT BIT06
3806 017044 012737 017052 001220  MOV     #9$,LOCK   ;SET RETURN IF SW09-1
3807 017052 004237 017136 9$: JSR     R2,10$    ;GOTO THE SUBROUTINE
3808 017056 000206          BIT7+BIT2+BIT1   ;POINT SET/CLEARED [SET]
3809 017060 073000          BRB+BIT10+BIT9 ;BR TEST POINT
3810 017062 000001          BIT0        ;DVLCR EXPECTED
3811 017064 004237 017136  JSR     R2,10$    ;GOTO SUB ROUTINE
3812 017070 000202          BIT7+BIT1     ;POINT SET/CLEARED [CLEARED]
3813 017072 073000          BRB+BIT10+BIT9 ;BR TEST POINT
    
```

```

3814 017074 000003          BIT1+BIT0          ;EXPECTED RESULTS
3815 017076 104401      77$: SCOPI          ;SWR 09=1?
3816
3817          ;*TEST SET/CLEAR FUNCTION
3818          ;*FOR RAM OUTPUT BIT07
3819 017100 012737 017106 00'220 101$: MOV #101$,LOCK ;SET RETURN IF SW09=1
3820 017106 004237 017136 JSR R2,10$ ;GOTO THE SUBROUTINE
3821 017112 000204          BIT7+BIT2          ;POINT SET/CLEARED [SET]
3822 017114 073400          BRB+BIT10+BIT9+BIT8 ;BR TEST POINT
3823 017116 000001          BIT0          ;DVLCR EXPECTED
3824 017120 004237 017136 JSR R2,10$ ;GOTO SUB ROUTINE
3825 017124 000200          BIT7          ;POINT SET/CLEARED [CLEARED]
3826 017126 073400          BRB+BIT10+BIT9+BIT8 ;BR TEST POINT
3827 017130 000003          BIT1+BIT0          ;EXPECTED RESULTS
3828 017132 104401      79$: SCOPI          ;SWR 09=1?
3829 017134 104400          SCOPE          ;SCOPE THE TEST
3830 017136 012710 050000 10$: MOV #S.C,(R0) ;SET/CLEAR INSTR
3831 017142 010201          MOV R2,R1 ;SAVE JSR PC ADDRESS
3832 017144 052210          BIS (R2)+,(R0) ;LOAD POINT SET/CLEARED
3833 017146 104415          ROMCLK ;CYCLE THE ROM
3834 017150 012210          MOV (R2)+,(R0) ;LOAD BR TEST POINT
3835 017152 011003          MOV (R0),R3 ;READ DVSFR INTO R3
3836 017154 012205          MOV (R2)+,R5 ;LOAD EXPECTED INTO R5
3837 017156 017704 162206 MOV @DVLCR,R4 ;READ DVLCR INTO FOUND LOC.
3838 017162 020504          CMP R5,R4 ;EXPECTED=FOUND?
3839 017164 001401          BEQ 11$ ;BR IF YES
3840 017166 104005          HLT 5 ;DVLCR WRONG BR RESULTS.
3841 017170 000202      11$: RTS R2 ;RETURN
3842
3843          ;***** TEST 71 *****
3844          ;*TEST OF 'RECEIVER CHARACTER SILO'
3845          ;*THRU THE USE OF THE DVSFR REG.
3846          ;*TEST THE FILLING THE SILO PRODUCES 'SILO FULL'
3847          ;*ON EXACTLY THE 128 LOAD.
3848          ;*SET/CLEAR IS USED TO STUFF SILO AND BRANCH A IS USED TO TEST SILO.
3849          ;*SET/CLEAR 'SILO IN' AND SET/CLEAR 'SILO OUT' ARE EXERCISED TOO.
3850          ;*****
3851
3852          ; TEST 71
3853          ;-----
3854 017172 012737 000071 001226 TST71: MOV #71,TSTNO
3855 017200 012737 017414 001216 MOV #TST72,NEXT
3856 017206 104412          MSTCLR ;CLEAR DV11
3857 017210 012700 000177          MOV #127,R0 ;SET R0 TO 1 LESS THAN FULL SILO
3858 017214 012777 000010 162140 MOV #BIT3,@DVSCR ;SET SOURCE SEL
3859 017222 012705 000003          MOV #BIT1+BIT0,R5 ;SET EXPECTED RESULTS INTO R5
3860 017226 012777 050021 162144 1$: MOV #S.C+BIT4+BIT0,@DVSFR
3861 017234 104415          ROMCLK ;S/C 'SILO IN'
3862 017236 012777 007400 162134 MOV #BIT11+BIT10+BIT9+BIT8,@DVSFR
3863          ;BR-A 'SILO FULL'?
3864 017244 017702 162130          MOV @DVSFR,R2 ;SAVE CONTENTS OF DVSFR FOR ERROR PRINTOUT
3865 017250 017704 162114          MOV @DVLCR,R4 ;READ DVLCR FOR RESULTS
3866 017254 020504          CMP R5,R4 ;ARE BR TEST POINTS CORRECT?
3867 017256 001401          BEQ 64$ ;BR IF YES
3868 017260 104006          HLT 6 ;BR TEST POINTS WRONG (BIT1 OR 0)
3869 017262 005300      64$: DEC R0 ;IS SILO FULL-1 YET?
    
```

```

3870 017264 001360          BNE 1$          ;BR IF NOT 127 TIMES YET
3871 017266 012777 050021 162104 2$: MOV #S.C+BIT4+BIT0,@DVSFR
3872                                ;S/C 'SILO IN'
3873 017274 104415          ROMCLK
3874 017276 000240          NOP
3875 017300 012777 007400 162072  MOV #BIT11+BIT10+BIT9+BIT8,@DVSFR
3876                                ;BR-A 'SILO FULL'?
3877 017306 017702 162066  MOV @DVSFR,R2      ;SAVE DVSFR
3878 017312 017704 162052  MOV @DVLCR,R4     ;READ BR TEST POINTS
3879 017316 042705 000001  BIC #BIT0,R5     ;ALTER EXPECTED RESULTS
3880 017322 020504          CMP R5,R4        ;BR TEST POINTS OK??
3881 017324 001401          BEQ 3$          ;BR IF YES
3882 017326 104006          HLT 6           ;BR TEST POINTS WRONG
3883 017330 012777 050020 162042 3$: MOV #S.C+BIT4,@DVSFR
3884 017336 104415          ROMCLK
3885 017340 000240          NOP
3886 017342 012777 007400 162030  MOV #BIT11+BIT10+BIT9+BIT8,@DVSFR
3887                                ;BR-A 'SILO FULL'?
3888 017350 005002          CLR R2          ;DELAY AT LEAST 32US
3889 017352 032777 000001 162010 4$: BIT #BIT0,@DVLCR  ;IS SILO *NOT FULL*??
3890 017360 001003          BNE 5$          ;BR IF OK.
3891 017362 062702 000001  ADD #1,R2        ;DELAY.....
3892 017366 001371          BNE 4$          ;GOTO 4$
3893 017370 017702 162004 5$: MOV @DVSFR,R2      ;SAVE DVSFR
3894 017374 017704 161770  MOV @DVLCR,R4     ;READ BR TEST POINTS
3895 017400 052705 000001  BIS #BIT0,R5     ;SET EXPECTED RESULTS
3896 017404 020504          CMP R5,R4        ;OK??
3897 017406 001401          BEQ 6$          ;YES
3898 017410 104006          HLT 6           ;SILO STILL FULL.
3899 017412 104400          6$: SCOPE      ;SCOPE TEST

```

```

3900
3901 ;***** TEST 72 *****
3902 ;*TEST THAT AFTER AN INIT
3903 ;*THAT 'RCV CHARACTER WAITING'
3904 ;*IS FALSE (HIGH) AND THEN VERIFY
3905 ;*THAT WHEN 'SILO IN' IS ASSERTED THAT
3906 ;*THAT 'RCVD CHARACTER WAITING' IS TRUE (LOW)
3907 ;*AND MAKES 'BRANCH A' TRUE.
3908 ;*****
3909

```

```

3910 ; TEST 72
3911 ;-----
3912 017414 012737 000072 001226 TST72: MOV #72,TSTNO
3913 017422 012737 017620 001216 MOV #TST73,NEXT
3914 017430 104412          MSTCLR          ;CLEAR DV11
3915 017432 012777 000010 161722  MOV #BIT3,@DVSCR ;SET SOURCE SEL
3916 017440 012705 000003  MOV #BIT1+BIT0,R5 ;SET EXPECTED RESULTS
3917 017444 012702 001400  MOV #BIT9+BIT8,R2 ;BR-A 'RCVD CHAR WAITING'?
3918 017450 010277 161724  MOV R2,@DVSFR    ;LOAD DV INSTR
3919 017454 017704 161710  MOV @DVLCR,R4    ;READ TEST POINTS
3920 017460 020504          CMP R5,R4        ;OK??
3921 017462 001401          BEQ 64$         ;YES
3922 017464 104006          HLT 6           ;TEST POINT RCV CHAR WAITING WRONG
3923 017466 012702 050021 64$: MOV #S.C+BIT4+BIT0,R2
3924                                ;S/C 'SILO IN'
3925 017472 010277 161702  MOV R2,@DVSFR    ;LOAD INSTR

```

```

3926 017476 104415 ROMCLK ;CLOCK
3927 017500 005004 CLR R4 ;PREPARE COUNTER
3928 017502 012702 001400 MOV #BIT9+BIT8,R2 ;BR-A RCV CHAR WAITING
3929 ;BR-A 'RCVD CHAR WAITING'?
3930 017506 010277 161666 MOV R2,@DVSFR ;LOAD INSTR
3931 017512 012705 000002 MOV #BIT1,R5 ;SET GOOD RESULTS
3932 017516 032777 000001 161644 1$: BIT #BIT0,@DVLCR ;TEST DV11 BR POINT
3933 017524 001403 BEQ 2$ ;BR IF OK
3934 017526 062704 000001 ADD #1,R4 ;DELAY
3935 017532 001371 BNE 1$ ;GOTO 1$
3936 017534 017704 161630 2$: MOV @DVLCR,R4 ;READ DV11 BR POINT
3937 017540 020504 CMP R5,R4 ;
3938 017542 001401 BEQ 3$ ;
3939 017544 104006 HLT 6 ;BR POINT RCV CHAR WAITING WRONG
3940 ;*TEST THAT SETTING DVSCRO7
3941 ;*INHIBITS RCV CHAR WAITING FROM APPEARING
3942 ;*TRUE; AND THAT CLEARING
3943 ;*DVSCRO7 MAKES IT APPEAR TRUE AGAIN.
3944
3945 017546 012705 000003 161602 3$: MOV #BIT1+BIT0,R5 ;LOAD EXPECTED
3946 017552 052777 001200 BIS #BIT9+BIT7,@DVSCR ;SET RECV INTER
3947 017560 017704 161604 MOV @DVLCR,R4 ;READ DV BR POINTS
3948 017564 020504 CMP R5,R4 ;
3949 017566 001401 BEQ 4$ ;
3950 017570 104006 HLT 6 ;BR TEST POINTS WRONG
3951 017572 042705 000001 161556 4$: BIC #BIT0,R5 ;RESET EXPECTED RESULTS
3952 017576 042777 000200 BIC #BIT7,@DVSCR ;CLEAR RECV INT
3953 017604 017704 161560 MOV @DVLCR,R4 ;READ BR POINTS
3954 017610 020504 CMP R5,R4 ;
3955 017612 001401 BEQ 5$ ;
3956 017614 104006 HLT 6 ;BR TEST POINTS WRONG
3957 017616 104400 5$: SCOPE ;SCOPE THIS TEST
3958
3959
3960 ;***** TEST 73 *****
3961 ;*BASIC TEST OF THE 'DATA TRANSFER INSTRUCTION'
3962 ;*BITS 07,06,05,04 OF DVSFR INDICATE THE SOURCE
3963 ;*BITS 03,02,01,00 OF DVSFR INDICATE THE DESTINATION.
3964 ;*****
3965
3966 ; TEST 73
3967 ;-----
3968 017620 012737 000073 001226 TST73: MOV #73,TSTNO
3969 017626 012737 020160 001216 MOV #TST74,NEXT
3970 017634 012737 017734 001220 MOV #1$,LOCK
3971 017642 104412 MST ;CLEAR DV11
3972 017644 012777 000010 161510 MOV #BIT3,@DVSCR ;SET SOURCE SEL
3973 017652 013700 001400 MOV /SFR,R0 ;SET DVSFR POINTER INTO R0
3974
3975 ;*TEST TO XFR SOURCE REGISTERS TO THE DVRIC
3976 ;*REGISTER VERIFYING THAT THE FOLLOWING REGISTERS
3977 ;*ARE CLEARED AND THAT THE XFR BUS IS CLEAR AFTER
3978 ;*A MSTCLR.
3979 ;*REGISTER FUNCTION
3980 ;* 0000 GROUND
3981 ;* 0001 GROUND
    
```

```

3982                ;* 0010          GROUND
3983                ;* 0011          GROUND
3984                ;* 0100          GROUND
3985                ;* 0101          MASTER SCAN 0-3/0-3
3986                ;* 0110          ALU RESULT 8-11/0-3
3987                ;* 0111          ALU RESULT 5-7/0-2
3988                ;* 1000          LOW BYTE=B REG 8-15 ; HIGH BYTE=GRND
3989                ;* 1001          LO BYTE=NPR OUT ; HI BYTE=CDC REG
3990                ;* 1010          RAM OUTPUT 0-2/8-10
3991                ;* 1011          RAM OUTPUT
3992                ;* 1101          NPR INPUT REGISTER
3993                ;* 1110          BCC REGISTER
3994                ;* 1111          ALU RESULT REGISTER
3995
3996 017656 005005    CLR      R5          ;SET EXPECTED TO 0
3997 017660 012702 030000  MOV     #XFR,R2      ;SET DATA XFR INSTR.
3998 017664 052702 000006  BIS     #BIT2+BIT1,R2 ;SET DESTINATION TO DVRIC REG.
3999 017670 005003    CLR      R3          ;ZERO SOURCE REG POINTER
4000 017672 042702 000360 65$:   BIC     #BIT7+BIT6+BIT5+BIT4,R2
4001 017676 050302    BIS     R3,R2        ;SET SOURCE REGISTER
4002 017700 010210    MOV     R2,(R0) ;LOAD SFR WITH XFR INSTR
4003 017702 104415    ROMCLK          ;EXECUTE INSTR
4004 017704 017704 161456  MOV     @DVRIC,R4    ;READ SOURCE REGISTER
4005 017710 001401    BEQ    66$          ;BR IF IT WAS ZERO
4006 017712 104006    HLT    6            ;SOURCE REGISTER IN SFR NOT ZERO
4007 017714 062703 000020 66$:   ADD     #BIT4,R3    ;UPDATE SOURCE REGISTER
4008 017720 022703 000300    CMP    #300,R3     ;DON'T DO SILO REGISTER!!
4009 017724 001773    BEQ    66$          ;GET NEXT REG IF THIS IS SILO.
4010 017726 032703 000360  BIT     #BIT7+BIT6+BIT5+BIT4,R3
4011 017732 001357    BNE    65$          ;BR IF MORE TO DO.
4012
4013                ;*TEST OF SET RAM OUTPUT BIT0
4014                ;*AND THE USE OF THE DATA XFER INSTR.
4015                ;*PLACE RAM BIT0 INTO THE DVRIC REG
4016 017734 012705 000400 1$:   MOV     #BIT8,R5
4017 017740 012703 030000  MOV     #XFR,R3 ;-DATA XFER-
4018 017744 052703 000246  BIS     #BIT7+BIT5+BIT2+BIT1,R3
4019 017750 004237 020100  JSR    R2,10$      ;S= RAM OUTPUT 0-2. D= DVRIC
4020 017754 000047    BIT5+BIT2+BIT1+BIT0
4021 017756 000043    BIT5+BIT1+BIT0
4022
4023                ;*TEST TO SET RAM OUTPUT DATA BIT3
4024                ;*AND THE USE OF THE 'DATA TRANSFER' INSTRUCTION
4025                ;*TO PLACE BIT3 INTO THE DVRIC REGISTER.
4026 017760 012737 017766 001220 3$:   MOV     #3$,LOCK    ;SET RETURN IF SW09=1
4027 017766 012705 000010  MOV     #BIT3,R5    ;SET EXPECTED DATA
4028 017772 012703 030000  MOV     #XFR,R3 ;-DATA XFER-
4029 017776 052703 000266  BIS     #BIT7+BIT5+BIT4+BIT2+BIT1,R3
4030 020002 004237 020100  JSR    R2,10$      ;S= RAM OUTPUT. D=DVRIC
4031 020006 000044    BIT5+BIT2
4032 020010 000040    BITS
4033
4034                ;*TEST TO SET RAM OUTPUT DATA BIT4
4035                ;*AND THE USE OF THE 'DATA TRANSFER' INSTRUCTION
4036                ;*TO PLACE BIT4 INTO THE DVRIC REGISTER.
4037 020012 012737 020020 001220  MOV     #4$,LOCK    ;SET RETURN IF SW09=1
    
```

```
4038 020020 012705 000020      4$:  MOV    #BIT4,R5      ;SET EXPECTED DATA
4039 020024 012703 030000      MOV    #XFR,R3      ;:-DATA XFER-
4040 020030 052703 000266      BIS    #BIT7+BIT5+BIT4+BIT2+BIT1,R3
4041 020034 004237 020100      JSR    R2,10$      ;S= RAM OUTPUT. D=DVRIC
4042 020040 000207      BIT7+BIT2+BIT1+BIT0
4043 020042 000203      BIT7+BIT1+BIT0
4044
4045      ;*TEST TO SET RAM OUTPUT DATA BIT7
4046      ;*AND THE USE OF THE 'DATA TRANSFER' INSTRUCTION
4047      ;*TO PLACE BIT7 INTO THE DVRIC REGISTER.
4048 020044 012737 020052 001220      6$:  MOV    #6$,LOCK      ;SET RETURN IF SW09=1
4049 020052 012705 000200      MOV    #BIT7,R5      ;SET EXPECTED DATA
4050 020056 012703 030000      MOV    #XFR,R3      ;:-DATA XFER-
4051 020062 052703 00026      BIS    #BIT7+BIT5+BIT4+BIT2+BIT1,R3
4052 020066 004237 020100      JSR    R2,10$      ;S= RAM OUTPUT. D=DVRIC
4053 020072 000204      BIT7+BIT2
4054 020074 000200      BIT7
4055 020076 104400      SCOPE
4056 020100 012710 050000      10$: MOV    #S.C,(R0) ;SET CLEAR INSTR
4057 020104 010201      MOV    R2,R1      ;JSR PC TO R1
4058 020106 052210      BIS    (R2)+,(R0) ;LOAD SET/CLEAR POINT
4059 020110 104415      ROMCLK ;EXECUTE
4060 020112 010310      MOV    R3,(R0) ;LOAD XFER
4061 020114 104415      ROMCLK ;EXECUTE
4062 020116 017704 161244      MOV    @DVRIC,R4 ;READ RESULTS
4063 020122 020504      CMP    R5,R4 ;OK??
4064 020124 001401      BEQ   11$ ;YES
4065 020126 104005      HLT   5 ;XFER FAILED
4066 020130 012710 050000      11$: MOV    #S.C,(R0) ;SET/CLEAR
4067 020134 052210      BIS    (R2)+,(R0) ;POINT
4068 020136 104415      ROMCLK ;EXECUTE
4069 020140 010310      MOV    R3,(R0) ;XFER
4070 020142 104415      ROMCLK ;EXECUTE
4071 020144 005005      CLR   R5 ;SET EXPECTED RESULTS
4072 020146 017704 161214      MOV    @DVRIC,R4 ;READ REAL RESULTS
4073 020152 001401      BEQ   12$ ;BR IF RESULT=0
4074 020154 104005      HLT   5 ;DVRICR NOT =0
4075 020156 000202      12$: RTS    R2 ;EXIT SUB
```

```
4076
4077
4078      ;***** TEST 74 *****
4079      ;*BASIC TEST OF THE 'ALU OPERATION' INSTRUCTION.
4080      ;*FIRST PART:ISSUE AN INIT AND MOVE
4081      ;*THE ALU RESULT REGISTER TO THE DVRIC
4082      ;*REGISTER VERIFYING THAT IT IS ZERO.
4083      ;*SECOND PART: DO A FUNCTION 'F=A'
4084      ;*THEN MOV 'F' TO THE DVRIC REGISTER VERIFYING IT TO
4085      ;*BE ZERO
4086      ;*THIRD PART: DO A FUNCTION 'F=A+B'; MOVING
4087      ;*'F' TO DVRIC AND MAKING SURE IT IS ZERO.
4088      ;*THUS THE FOLLOWING HAS BEEN TESTED:
4089      ;*ALU RESULT,'A' REG,AND 'B' REG ALL ZEROED ON INIT.
4090      ;*****
```

```
4091      ; TEST 74
4092      ;-----
4093
```

```

4094 020160 012737 000074 001226 TST74: MOV #74,TSTNO
4095 020166 012737 020300 001216 MOV #TST75,NEXT
4096 020174 104412 MSTCLR ;RESET DV11
4097 020176 012777 000010 161156 MOV #BIT3,@DVSCR ;SET SOURCE SELECT
4098 020204 013700 001400 MOV DVSFR,R0 ;SET DVSFR POINTER IN R0
4099 020210 012702 030366 MOV #XFR+BIT7+BIT6+BIT5+BIT4+BIT2+BIT1,R2
4100 020214 010210 MOV R2,(R0) ;XFR 'ALU RESULT REG.' TO DVNSR
4101 020216 104415 ROMCLK ;CLOCK INSTRUCTION
4102 020220 005005 CLR R5 ;ZERO 'EXPECTED' LOC
4103 020222 017704 161140 MOV @DVRIC,R4 ;READ 'ALU RESULT'
4104 020226 001401 BEQ 1$ ;S/B=0
4105 020230 104006 HLT 6 ;ALU RESULT NOT=0 ON INIT
4106 020232 012710 010037 1$: MOV #ALU+BIT4+BIT3+BIT2+BIT1+BIT0,(R0)
4107 020236 104415 ROMCLK ;DO 'ALU F=A'
4108 020240 010210 MOV R2,(R0) ;XFR 'ALU RESULT' TO DVNSR
4109 020242 104415 ROMCLK ;CLOCK INSTR
4110 020244 017704 161116 MOV @DVRIC,R4 ;READ RESULT
4111 020250 001401 BEQ 2$ ;S/B=0
4112 020252 104006 HLT 6 ;'A' REG NOT=0 ON INIT
4113 020254 012710 010026 2$: MOV #ALU+BIT4+BIT2+BIT1,(R0)
4114 020260 104415 ROMCLK ;ALU F=A+B
4115 020262 010210 MOV R2,(R0) ;XFR TO RIC
4116 020264 104415 ROMCLK ;CLOCK INSTR.
4117 020266 017704 161074 MOV @DVRIC,R4 ;READ RESULT
4118 020272 001401 BEQ 3$ ;S/B=0
4119 020274 104006 HLT 6 ;'B' REG NOT=0 ON INIT
4120 020276 104400 3$: SCOPE ;SCOPE TEST
    
```

```

4121
4122 :***** TEST 75 *****
4123 :*TEST OF ALU OPERATIONS.
4124 :*TEST OF ALL ALU OPERATIONS USED BY DV11.
4125 :*FUNCTIONS TESTED:(NOTE THAT 'F' IS ALU RESULT)
4126 :*DVSFR BITS:
4127 :*BIT12 BIT05 BIT04 BIT03 BIT02 BIT01 BIT00 FUNCTION
4128 :* 1 0 1 1 1 0 0 F -1
4129 :* 1 0 0 1 1 0 0 F-0
4130 :* 1 0 1 1 1 1 1 F A
4131 :* 1 0 0 0 1 0 1 F-B
4132 :* 1 1 1 1 1 1 1 F-A+1
4133 :* 1 0 1 0 1 1 0 F A+B
4134 :*****
    
```

```

4135 : TEST 75
4136 :-----
4137 TST75: MOV #75,TSTNO
4138 020300 012737 000075 001226 MOV #TST76,NEXT
4139 020306 012737 021210 001216 ;*FUNCTION TESTED
4140 ;*F=-1,RIC_F
4141 ;*
4142
4143 020314 012737 020322 001220 10$: MOV #10$,LOCK ;SET FOR SW09
4144 020322 104412 MSTCLR ;CLEAR DV11
4145 020324 012777 000010 161030 MOV #BIT3,@DVSCR ;SET SOURCE SEL
4146 020332 013700 001400 MOV DVSFR,R0 ;SET DVSFR POINTER IN R0
4147 020336 012702 030366 MOV #XFR+BIT7+BIT6+BIT5+BIT4+BIT2+BIT1,R2
4148 020342 012710 010034 MOV #ALU+BIT4+BIT3+BIT2,(R0)
4149 020346 104415 ROMCLK ;DO 'F=-1'
    
```



```

4150 020350 010210      MOV      R2,(R0)      ;XFR 'ALU RESULT 'O DVRIC'
4151 020352 104415      ROMCLK
4152 020354 017704 161006  MOV      @DVVIC,R4      ;READ RESULTS
4153 020360 012705 177777  MOV      #-1,R5      ;SET EXPECTED
4154 020364 020504      CMP      R5,R4      ;DID F=-1 WORK?
4155 020366 001401      BEQ      638      ;BR IF YES
4156 020370 104006      HLT      6      ;F=-1 APPEARED TO FAIL
4157      ;*TEST THAT DVVIC (NOW THAT ITS ALL 1'S)
4158      ;*CAN BE CLEARED BY A MSTCLR.
4159      ;*
4160 020372 104401 638:  SCOP1      ;SW09=1?
4161 020374 012737 020402 0C1220  MOV      #118,LOCK      ;SET RETURN IF SW09=1
4162 020402 104412 118:  MSTCLR      ;CLEAR DV11
4163 020404 005005      CLR      R5      ;SET EXPECTED RESULTS
4164 020406 017704 160754  MOV      @DVVIC,R4      ;READ DVVIC
4165 020412 001401      BEQ      648      ;BR IF=0
4166 020414 104006      HLT      6      ;DVVIC REG. NOT CLEARED ON INIT
4167      ;*NEXT SET OF FUNCTIONS:
4168      ;*F=0,RIC_F
4169      ;*
4170 020416 012777 000010 160735 648:  MOV      #BIT3,@DVSCR      ;SET SOURCE SEL
4171 020424 012710 010034  MOV      #ALU+BIT4+BIT3+BIT2,(R0)
4172 020430 104415      ROMCLK      ;'F=-1'
4173 020432 012710 010014  MOV      #ALU+BIT3+BIT2,(R0)
4174 020436 104415      ROMCLK      ;'F=0'
4175 020440 010210      MOV      R2,(R0)      ;XFR 'F' TO DVVIC
4176 020442 104415      ROMCLK
4177 020444 005005      CLR      R5      ;SET EXPECTED
4178 020446 017704 160714  MOV      @DVVIC,R4      ;READ RESULTS
4179 020452 001401      BEQ      658      ;BR IF=0
4180 020454 104006      HLT      6      ;'F=0' APPEARED TO FAIL
4181      ;*NEXT SET OF FUNCTIONS:
4182      ;*F=A+1,RIC_F,A_F,F=0,F=A
4183      ;*
4184 020456 104401 658:  SCOP1      ;SW09=1?
4185 020460 012737 020466 001220  MOV      #128,LOCK      ;SET RETURN
4186 020466 012705 000001 128:  MOV      #1,R5      ;SET GOOD RESULTS
4187 020472 012710 010077 18:  MOV      #ALU+BIT5+BIT4+BIT3+BIT2+BIT1+BIT0,(R0)
4188 020476 052777 000002 160656  BIS      #BIT1,@DVSCR      ;ISSUE ROM CLOCK
4189 020504 010210      MOV      R2,(R0)      ;XFR 'F' TO DVVIC
4190 020506 052777 000002 160646  BIS      #BIT1,@DVSCR      ;ISSUE ROM CLK
4191 020514 017704 160646  MOV      @DVVIC,R4      ;READ RESULTS
4192 020520 020504      CMP      R5,R4      ;FUNCTION WORK?
4193 020522 001401      BEQ      668      ;YES
4194 020524 104006      HLT      6      ;F=A+1 APPEARED TO FAIL
4195 020526 012710 030361 668:  MOV      #XFR+BIT7+BIT6+BIT5+BIT4+BIT0,(R0)
4196 020532 052777 000002 160622  BIS      #BIT1,@DVSCR      ;ISSUE ROM CLK
4197 020540 012710 010014  MOV      #ALU+BIT3+BIT2,(R0)
4198 020544 052777 000002 160610  BIS      #BIT1,@DVSCR      ;ROM CLK
4199 020552 010210      MOV      R2,(R0)      ;XFR RIC_F
4200 020554 052777 000002 160600  BIS      #BIT1,@DVSCR      ;ROM CLK
4201 020562 017704 160600  MOV      @DVVIC,R4      ;READ RESULT
4202 020566 001401      BEQ      678      ;BR IF GOOD
4203 020570 104000      HLT      0      ;F=0 FAILED
4204 020572 012710 010037 678:  MOV      #ALU+BIT4+BIT3+BIT2+BIT1+BIT0,(R0)
4205 020576 052777 000002 160556  BIS      #BIT1,@DVSCR      ;CLOCK 'F=A'

```

```

4206 020604 010210          MOV     R2,(R0)          ;XFR RIC_F
4207 020606 052777 000002 160546  BIS     #BIT1,@DVSCR    ;ROM CLK
4208 020614 017704 160546  MOV     @DVSCR,R4       ;READ RESULT
4209 020620 020504          CMP     R5,R4          ;BR IF OK
4210 020622 001401          BEQ    68$             ;
4211 020624 104006          HLT    6              ;F=A FAILED
4212 020626 005205 68$:   INC     R5              ;UPDATE DATA
4213 020630 001320          BNE    18             ;BR IF NOT ALL DONE.
4214 020632 104401          SCOP1                 ;SW09=1?
4215          ;*NEXT SET OF FUNCTIONS:
4216          ;*F=A+1,A_F,B_F,F=A+B,RIC_F
4217          ;*
4218 020634 012737 020642 001220  MOV     #13$,LOCK      ;SET RETURN
4219 020642 104412          MSTCLR                 ;RESET DV11
4220 020644 012777 000010 160510 13$:   MOV     #BIT3,@DVSCR   ;SET SOURCE SEL.
4221 020652 012705 000002          MOV     #2,R5         ;SET EXPECTED RESULTS
4222 020656 013701 001366          MOV     DVSCR,R1      ;SET POINTER
4223 020662 013703 001362          MOV     DVSCR,R3      ;SET POINTER
4224 020666 012710 010077          MOV     #ALU+BIT5+BIT4+BIT3+BIT2+BIT1+BIT0,(R0)
4225 020672 104415          ROMCLK                 ;F=A+1
4226 020674 012710 030361          MOV     #XFR+BIT7+BIT6+BIT5+BIT4+BIT0,(R0)
4227 020700 104415          ROMCLK                 ;XFR A ALU RESULT
4228 020702 012710 030362 2$:   MOV     #XFR+BIT7+BIT6+BIT5+BIT4+BIT1,(R0)
4229 020706 052713 000002          BIS     #BIT1,(R3)    ;XFR B ALU RESULT
4230 020712 012710 010026          MOV     #ALU+BIT4+BIT2+BIT1,(R0)
4231 020716 052713 000002          BIS     #BIT1,(R3)    ;CLOCK F=A+B
4232 020722 010210          MOV     R2,(R0)       ;XFR RIC_F
4233 020724 052713 000002          BIS     #BIT1,(R3)    ;
4234 020730 011104          MOV     (R1),R4       ;READ DVRIC
4235 020732 020504          CMP     R5,R4         ;FUNCTION WORK?
4236 020734 001401          BEQ    69$             ;BR IF YES
4237 020736 104006          HLT    6              ;F=A+B APPEARED TO FAIL
4238 020740 005205 69$:   INC     R5              ;INC DATA POINTER
4239 020742 022705 000002          CMP     #2,R5         ;ALL DONE?
4240 020746 001355          BNE    2$             ;BR IF NO
4241 020750 104412          MSTCLR                 ;RESET DV11
4242 020752 017704 160410          MOV     @DVSCR,R4     ;DVIC ZERO ON INIT?
4243 020756 001401          BEQ    70$             ;BR IF YES
4244 020760 104000          HLT    0              ;S/B=0
4245 020762 104401 70$:   SCOP1                 ;SW09=1?
4246          ;*NEXT SET OF FUNCTIONS:
4247          ;*F -1,B_F,F=0,F=B
4248          ;*
4249 020764 012737 020772 001220  MOV     #14$,LOCK      ;SET RETURN
4250 020772 104412          MSTCLR                 ;RESET DV11
4251 020774 012777 000010 160360 14$:   MOV     #BIT3,@DVSCR   ;SET SOURCE SEL.
4252 021002 012710 010034          MOV     #ALU+BIT4+BIT3+BIT2,(R0)
4253 021006 104415          ROMCLK                 ;F=-1
4254 021010 012710 030362          MOV     #XFR+BIT7+BIT6+BIT5+BIT4+BIT1,(R0)
4255 021014 104415          ROMCLK                 ;XFR B_F
4256 021016 012710 010014          MOV     #ALU+BIT3+BIT2,(R0) ;
4257 021022 104415          ROMCLK                 ;F=0
4258 021024 012710 010005          MOV     #ALU+BIT2+BIT0,(R0) ;
4259 021030 104415          ROMCLK                 ;F=B
4260 021032 010210          MOV     R2,(R0)       ;XFR RIC_F
4261 021034 104415          ROMCLK                 ;CLOCK

```

```

4262 021036 017704 160324      MOV    @DVRIC,R4      ;READ RESULTS
4263 021042 012705 177777      MOV    #-1,R5        ;SET EXPECTED
4264 021046 020504              CMP    R5,R4         ;DID F=B WORK?
4265 021050 001401              BEQ    71$           ;BR IF YES
4266 021052 104006              HLT   6              ;F=B FAILED
4267 021054 104401              71$: SCOP1          ;SW09=1?
4268                          ;*NEXT SET OF FUNCTIONS:
4269                          ;*CATCH 'SET/CLEAR' THAT WAS MISSED.
4270                          ;*F=-1,S/C[ALU01=0],RIC_F
4271                          ;*
4272 021056 012737 021064 001220 15$: MOV    #15$,LOCK      ;SET RETURN
4273 021064 104412              MSTCLR                ;RESET DV11
4274 021066 012777 000010 160266 MOV    #BIT3,@DVSCR   ;SET SOURCE SELECT
4275 021074 012710 010034      MOV    #ALU+BIT4+BIT3+BIT2,(R0)
4276 021100 104415              ROMCLK                ;F=-1
4277 021102 012710 050026      MOV    #S.C+BIT4+BIT2+BIT1,(R0)
4278 021106 104415              ROMCLK                ;S/C 'CLEAR ALU01'
4279 021110 010210              MOV    R2,(R0)        ;XFR RIC_F
4280 021112 104415              ROMCLK                ;
4281 021114 011104              MOV    (R1),R4        ;READ DVRIC
4282 021116 012705 177775      MOV    #177775,R5    ;SET EXPECTED
4283 021122 020504              CMP    R5,R4         ;DID S/C ALU01 WORK?
4284 021124 001401              BEQ    72$           ;BR IF YES
4285 021126 104006              HLT   6              ;S/C ALU01 FAILED.
4286 021130 104401              72$: SCOP1          ;SW09=1?
4287                          ;*NEXT SET OF FUNCTIONS:
4288                          ;*CATCH ANOTHER 'SET/CLEAR' THAT WAS MISSED.
4289                          ;*F=-1,S/C[ALU HIGH BYTE=0],RIC_F
4290                          ;*
4291 021132 012737 021140 001220 16$: MOV    #16$,LOCK      ;SET RETURN
4292 021140 104412              MSTCLR                ;RESET DV11
4293 021142 012777 000010 160212 MOV    #BIT3,@DVSCR   ;SET SOURCE SEL
4294 021150 012710 010034      MOV    #ALU+BIT4+BIT3+BIT2,(R0)
4295 021154 104415              ROMCLK                ;F=-1
4296 021156 012710 050027      MOV    #S.C+BIT4+BIT2+BIT1+BIT0,(R0)
4297 021162 104415              ROMCLK                ;S/C 'CLEAR ALU HIGH BYTE'
4298 021164 010210              MOV    R2,(R0)        ;XFR RIC_F
4299 021166 104415              ROMCLK                ;
4300 021170 011104              MOV    (R1),R4        ;READ RIC
4301 021172 012705 000377      MOV    #377,R5       ;SET EXPECTED
4302 021176 020504              CMP    R5,R4         ;DID S/C WORK?
4303 021200 001401              BEQ    73$           ;BR IF YES
4304 021202 104006              HLT   6              ;S/C ALU HIGH BYTE FAILED.
4305 021204 104401              73$: SCOP1          ;SW09=1?
4306 021206 104400              SCOPE                 ;SCOPE TEST

```

```

4307
4308
4309 ;***** TEST 76 *****
4310 ;*MASTER SCANNER TEST.
4311 ;*VERIFY FIRST THAT THE MASTER SCANNER
4312 ;*IS CLEARED BY INIT.
4313 ;*VERIFY SECONDLY THAT THE MASTER SCANNER
4314 ;*CAN BE INCREMENTED FROM 0 THRU 17 BACK TO 0.
4315 ;*****
4316
4317

```

; TEST 76

```

4318
4319 021210 012737 000076 001226 TST76: MOV #76,TSTNO
4320 021216 012737 021336 001226 MOV #TST77,NEXT
4321 021224 104412 MSTCLR :RESET DV11
4322 021226 012777 000010 160126 MOV #BIT3,@DVSCR :SET SOURCE SEL
4323 021234 013700 001400 MOV DVSFR,R0 :SET DVSFR POINTER
4324 021240 012702 030126 MOV #XFR+BIT6+BIT4+BIT2+BIT1,R2
4325 021244 010210 MOV R2,(R0) :XFR 'MASTER SCAN 0-3' TO DVRIC
4326 021246 104415 ROMCLK
4327 021250 005005 CLR R5 :SET EXPECTED
4328 021252 017704 160110 MOV @DVRIC,R4 :READ RESULTS
4329 021256 001401 BEQ 1$ :BR IF=0
4330 021260 104006 HLT 6 :MSCAN NOT=0 ON INIT
4331 021262 005205 1$: INC R5 :UPDATE POINTER
4332 021264 012703 000025 MOV #25,R3 :SET COUNT TO 25
4333 021270 012710 050102 2$: MOV #S.C+BIT6+BIT1,(R0) ;S/C 'ADVANCE MSCAN'
4334 021274 104415 ROMCLK :CLOCK
4335 021276 012710 050102 MOV #S.C+BIT6+BIT1,(R0) ;S/C 'ADVANCE MSCAN'
4336 021302 104415 ROMCLK :CLOCK
4337 021304 010210 MOV R2,(R0) :XFR RIC_MSCAN
4338 021306 104415 ROMCLK :CLOCK
4339 021310 017704 160052 MOV @DVRIC,R4 :READ RESULTS
4340 021314 020504 CMP R5,R4 :MSCAN INCREMENTED?
4341 021316 001401 BEQ 3$ :BR IF YES
4342 021320 104006 HLT 6 :MSCAN WRONG
4343 021322 005205 3$: INC R5 :UPDATE
4344 021324 042705 177760 BIC #^C<17>,R5 :CLEAN
4345 021330 005303 DEC R3 :COUNT DONE?
4346 021332 001356 BNE 2$ :BR IF NO
4347 021334 104400 SCOPE :SCOPE
4348
4349
4350
4351
4352
4353
4354
4355
4356
4357
4358

```

```

:***** TEST 77 *****
:*BASIC TESTS OF THE 'RAM OPERATION' INSTRUCTION.
:*VERIFY THE READ PORTION OF THE RAM OPERATION.
:*LOAD ALL SECONDARY REGISTERS OF ALL LINES
:*WITH DIFFERENT NUMBERS AND VERIFY THAT THE RAM OPERATION
:*CAN READ THE CORRECT SEC. REG. INTO THE DVRIC REG.
:*****

```

: TEST 77

```

4359 021336 012737 000077 001226 TST77: MOV #77,TSTNO
4360 021344 012737 021562 001216 MOV #TST100,NEXT
4361 021352 104412 MSTCLR :RESET DV11
4362 021354 005000 CLR R0
4363 021356 005001 CLR R1
4364 021360 013702 001372 MOV DVSRS,R2
4365 021364 013703 001374 MOV DVSRSR,R3
4366 021370 013705 001376 MOV DVSRA,R5
4367 021374 012704 000001 MOV #1,R4
4368 021400 110012 1$: MOV R0,(R2) :LOAD LINE NUMBER
4369 021402 110113 MOV R1,(R3) :LOAD SEC. REG. POINTER
4370 021404 010415 MOV R4,(R5) :LOAD DATA
4371 021406 122024 (MPB (R0)+,(R4)+ :UPDATE LINE AND DATA
4372 021410 022700 000020 (MP #16.,R0 :ALL LINES DONE?
4373 021414 001371 BNE 1$ :BR IF NO

```

```

4374 021416 005000 CLR R0 ;ZERO LINE POINTER
4375 021420 005201 INC R1 ;UPDATE SEC REG POINTER.
4376 021422 022701 000020 CMP #16.,R1 ;ALL SEC REG POINTERS DONE?
4377 021426 001364 BNE 1$ ;BR IF NO
4378 021430 005077 157736 2$: CLR @DVSRS ;ZERO POINTERS
4379 021434 012705 000001 MOV #1,R5 ;SET GOOD DATA
4380 021440 012777 000010 157714 MOV #BIT3,@DVSCR ;SET SOURCE SEL
4381 021446 012703 020000 MOV #RAM,R3 ;LOAD RAM INSTR.
4382 021452 013700 001400 MOV DVSR,R0 ;SET POINTER
4383 021456 012702 030266 MOV #XFR+BIT7+BIT5+BIT4+BIT2+BIT1,R2
4384 021462 010310 3$: MOV R3,(R0) ;DO RAM READ
4385 021464 104415 ROMCLK ;EXECUTE
4386 021466 010210 MOV R2,(R0) ;XFR RIC_RAM OUTPUT
4387 021470 104415 ROMCLK ;CLOCK
4388 021472 017704 157670 MOV @DVRIC,R4 ;READ RESULT
4389 021476 020504 CMP R5,R4 ;GOOD?
4390 021500 001401 BEQ 4$ ;BR IF YES
4391 021502 104006 HLT 6 ;RAM READ FAILED.
4392 021504 062705 000020 4$: ADD #20,R5 ;UPDATE DATA
4393 021510 005203 INC R3 ;UPDATE POINTER
4394 021512 122703 000020 CMPB #16.,R3 ;ALL DONE
4395 021516 001361 BNE 3$ ;BR IF NO
4396 021520 042705 177760 BIC #^C<17>,R5 ;CLEAR JUNK
4397 021524 012703 020000 MOV #RAM,R3 ;SET RAM INSTR
4398 021530 010046 MOV R0,-(SP) ;SAVE R0 ON STACK
4399 021532 004537 031676 PERFORM ,SETSCAN ;UPDATE MSCANNER
4400 021536 000001 1 ;
4401 021540 012600 MOV (SP)+,R0 ;RESTORE R0
4402 021542 005205 INC R5 ;UPDATE DATA
4403 021544 012710 030124 MOV #XFR+BIT6+BIT4+BIT2,(R0) ;XFR RAR_MSCAN 0-3
4404 021550 104415 ROMCLK ;EXECUTE
4405 021552 022705 000020 CMP #16.,R5 ;ALL DONE?
4406 021556 001341 BNE 3$ ;BR IF NO
4407 021560 104400 SCOPE ;SCOPE TEST.

```

```

4408
4409 ;***** TEST 100 *****
4410 ;*TEST OF BRANCH A TEST POINTS
4411 ;*THAT WERE PREVIOUSLY SKIPPED BECAUSE
4412 ;*OF THE SIGNALS NEEDED TO TEST THE
4413 ;*FUNCTIONS:
4414 ;*BIT11 BIT10 BIT09 BIT08 FUNCTION
4415 ;* 0 0 0 0 ALU 15=1,0
4416 ;* 1 0 0 0 ALU 13=1,0
4417 ;* 1 0 0 0 -12=1,0
4418 ;* 1 0 1 0 ALU 00=1,0
4419 ;* 1 0 1 1 ALU 01=1,0
4420 ;* 1 1 0 0 ALU 02=1,0
4421 ;* 1 1 0 1 ALU 03=1,0
4422 ;* 1 1 1 0 ALU 04=1,0
4423 ;*****
4424

```

```

4425 ; TEST 100
4426 -----
4427 021562 012737 000100 001226 TST100: MOV #100,TSTNO
4428 021570 012737 023164 001216 MOV #TST101,NEXT
4429 021576 005077 157570 CLR @DVSRS

```

```
4430 ;*BRANCH 'A' TEST OF ALU 15
4431 ;*
4432 021602 104412 MSTCLR ;RESET DV11
4433 021604 012777 000010 157550 MOV #BIT3,@DVSCR ;SET SOURCE SEL
4434 021612 012777 100000 157556 MOV #BIT15,@VSR A ;LOAD DATA
4435 021620 012777 020000 157552 MOV #RAM,@DV SFR ;DO A 'RAM READ'
4436 021626 104415 ROMCLK ;EXECUTE
4437 021630 012777 030261 157542 MOV #XFR+BIT7+BIT5+BIT4+BIT0,@DV SFR
4438 021636 104415 ROMCLK ;XFR RAM OUTPUT TO 'A' REG
4439 021640 012777 010037 157532 MOV #ALU+BIT4+BIT3+BIT2+BIT1+BIT0,@DV SFR
4440 021646 104415 ROMCLK ;F=A
4441 021650 012777 000000 157522 MOV #0000,@DV SFR ;LOAD BRANCH POINT
4442 021656 017704 157506 MOV @DVLCR,R4 ;READ BRANCH TEST POINT
4443 021662 042704 177774 BIC #^C<BIT1+BIT0>,R4 ;CLEAR JUNK
4444 021666 012705 000002 MOV #BIT1,R5 ;SET EXPECTED
4445 021672 020504 CMP R5,R4 ;BR POINTS CORRECT?
4446 021674 001401 BEQ 64$ ;BR IF YES
4447 021676 104006 HLT 6 ;BRANCH POINTS WRONG
4448 021700 104412 64$: MSTCLR
4449 021702 012777 000010 157452 MOV #BIT3,@DVSCR ;SET SOURCE SEL.
4450 021710 012777 000000 157462 MOV #0000,@DV SFR ;RESET DV11
4451 021716 017704 157446 MOV @DVLCR,R4 ;LOAD BRANCH. POINT TEST
4452 021722 042704 177774 BIC #^C<BIT1+BIT0>,R4 ;READ BR POINTS
4453 021726 012705 000003 MOV #BIT1+BIT0,R5 ;CLEAR JUNK
4454 021732 020504 CMP R5,R4 ;SET EXPECTED
4455 021734 001401 BEQ 65$ ;BR POINT OK?
4456 021736 104006 HLT 6 ;BR IF YES
4457 021740 65$: ;BR POINTS WRONG
4458 ;*BRANCH 'A' TEST OF ALU 13-
4459 ;*
4460 021740 104412 MSTCLR ;RESET DV11
4461 021742 012777 000010 157412 MOV #BIT3,@DVSCR ;SET SOURCE SEL
4462 021750 012777 020000 157420 MOV #BIT13,@DV SRA ;LOAD DATA
4463 021756 012777 020000 157414 MOV #RAM,@DV SFR ;DO A 'RAM READ'
4464 021764 104415 ROMCLK ;EXECUTE
4465 021766 012777 030261 157404 MOV #XFR+BIT7+BIT5+BIT4+BIT0,@DV SFR
4466 021774 104415 ROMCLK ;XFR RAM OUTPUT TO 'A' REG
4467 021776 012777 010037 157374 MOV #ALU+BIT4+BIT3+BIT2+BIT1+BIT0,@DV SFR
4468 022004 104415 ROMCLK ;F=A
4469 022006 012777 004000 157364 MOV #BIT11,@DV SFR ;LOAD BRANCH POINT
4470 022014 017704 157350 MOV @DVLCR,R4 ;READ BRANCH TEST POINT
4471 022020 042704 177774 BIC #^C<BIT1+BIT0>,R4 ;CLEAR JUNK
4472 022024 012705 060002 MOV #BIT1,R5 ;SET EXPECTED
4473 022030 020504 CMP R5,R4 ;BR POINTS CORRECT?
4474 022032 001401 BEQ 66$ ;BR IF YES
4475 022034 104006 HLT 6 ;BRANCH POINTS WRONG
4476 022036 104412 66$: MSTCLR
4477 022040 012777 000010 157314 MOV #BIT3,@DVSCR ;SET SOURCE SEL.
4478 022046 012777 004000 157324 MOV #BIT11,@DV SFR ;RESET DV11
4479 022054 017704 157310 MOV @DVLCR,R4 ;LOAD BRANCH. POINT TEST
4480 022060 042704 177774 BIC #^C<BIT1+BIT0>,R4 ;READ BR POINTS
4481 022064 012705 000003 MOV #BIT1+BIT0,R5 ;CLEAR JUNK
4482 022070 020504 CMP R5,R4 ;SET EXPECTED
4483 022072 001401 BEQ 67$ ;BR POINT OK?
4484 022074 104006 HLT 6 ;BR IF YES
4485 022076 67$: ;BR POINTS WRONG
```

```

4486 ;*BRANCH 'A' TEST OF ALU -12
4487 ;*
4488 022076 104412 MSTCLR ;RESET DV11
4489 022100 012777 000010 157254 MOV #BIT3,@DVSCR ;SET SOURCE SEL
4490 022100 012777 010000 157262 MOV #BIT12,@DVSRA ;LOAD DATA
4491 022114 012777 020000 157256 MOV #RAM,@DVSFR ;DO A 'RAM READ'
4492 022122 104415 ROMCLK ;EXECUTE
4493 022124 012777 030261 157246 MOV #XFR+BIT7+BIT5+BIT4+BIT0,@DVSFR
4494 022132 104415 ROMCLK ;XFR RAM OUTPUT TO 'A' REG
4495 022134 012777 010037 157236 MOV #ALU+BIT4+BIT3+BIT2+BIT1+BIT0,@DVSFR
4496 022142 104415 ROMCLK ;F=A
4497 022144 012777 004000 157226 MOV #BIT11,@DVSFR ;LOAD BRANCH POINT
4498 022152 017704 157212 MOV @DVLCR,R4 ;READ BRANCH TEST POINT
4499 022156 042704 177774 BIC #^C<BIT1+BIT0>,R4 ;CLEAR JUNK
4500 022162 012705 000002 MOV #BIT1,R5 ;SET EXPECTED
4501 022166 020504 CMP R5,R4 ;BR POINTS CORRECT?
4502 022170 001401 BEQ 68$ ;BR IF YES
4503 022172 104006 HLT 6 ;BRANCH POINTS WRONG
4504 022174 104412
4505 022176 012777 000010 157156 68$: MSTCLR
4506 022204 012777 004000 157166 MOV #BIT3,@DVSCR ;SET SOURCE SEL.
4507 022212 017704 157152 MOV #BIT11,@DVSFR ;RESET DV11
4508 022216 042704 177774 MOV @DVLCR,R4 ;LOAD BRANCH. POINT TEST
4509 022222 012705 000003 BIC #^C<BIT1+BIT0>,R4 ;READ BR POINTS
4510 022226 020504 MOV #BIT1+BIT0,R5 ;CLEAR JUNK
4511 022230 001401 CMP R5,R4 ;SET EXPECTED
4512 022232 104006 BEQ 69$ ;BR POINT OK?
4513 022234 HLT 6 ;BR IF YES
4514 69$: ;BR POINTS WRONG
4515 ;*BRANCH 'A' TEST OF ALU 00
4516 ;*
4516 022234 104412 MSTCLR ;RESET DV11
4517 022236 012777 000010 157116 MOV #BIT3,@DVSCR ;SET SOURCE SEL
4518 022244 012777 000001 157124 MOV #BIT0,@DVSRA ;LOAD DATA
4519 022252 012777 020000 157120 MOV #RAM,@DVSFR ;DO A 'RAM READ'
4520 022260 104415 ROMCLK ;EXECUTE
4521 022262 012777 030261 157110 MOV #XFR+BIT7+BIT5+BIT4+BIT0,@DVSFR
4522 022270 104415 ROMCLK ;XFR RAM OUTPUT TO 'A' REG
4523 022272 012777 010037 157100 MOV #ALU+BIT4+BIT3+BIT2+BIT1+BIT0,@DVSFR
4524 022300 104415 ROMCLK ;F=A
4525 022302 012777 005000 157070 MOV #BIT11+BIT9,@DVSFR ;LOAD BRANCH POINT
4526 022310 017704 157054 MOV @DVLCR,R4 ;READ BRANCH TEST POINT
4527 022314 042704 177774 BIC #^C<BIT1+BIT0>,R4 ;CLEAR JUNK
4528 022320 012705 000002 MOV #BIT1,R5 ;SET EXPECTED
4529 022324 020504 CMP R5,R4 ;BR POINTS CORRECT?
4530 022326 001401 SEQ 70$ ;BR IF YES
4531 022330 104006 HLT 6 ;BRANCH POINTS WRONG
4532 022332 104412 70$: MSTCLR
4533 022334 012777 000010 157020 MOV #BIT3,@DVSCR ;SET SOURCE SEL.
4534 022342 012777 005000 157030 MOV #BIT11+BIT9,@DVSFR ;RESET DV11
4535 022350 017704 157014 MOV @DVLCR,R4 ;LOAD BRANCH. POINT TEST
4536 022354 042704 177774 BIC #^C<BIT1+BIT0>,R4 ;READ BR POINTS
4537 022360 012705 000003 MOV #BIT1+BIT0,R5 ;CLEAR JUNK
4538 022364 020504 CMP R5,R4 ;SET EXPECTED
4539 022366 001401 BEQ 71$ ;BR POINT OK?
4540 022370 104006 HLT 6 ;BR IF YES
4541 022372 71$: ;BR POINTS WRONG
    
```

```

4542      . *BRANCH 'A' TEST OF ALU 01
4543      : *
4544 022372 104412 MSTCLR          : RESET DV11
4545 022374 012777 000010 156760 MOV #BIT3,@DVSCR : SET SOURCE SEL
4546 022402 012777 000002 156766 MOV #BIT1,@DVSRA : LOAD DATA
4547 022410 012777 020000 156762 MOV #RAM,@DVSFR  : DO A 'RAM READ'
4548 022416 104415 ROMCLK          : EXECUTE
4549 022420 012777 030261 156752 MOV #XFR+BIT7+BIT5+BIT4+BIT0,@DVSFR : XFR RAM OUTPUT TO 'A' REG
4550 022426 104415 ROMCLK          : F=A
4551 022430 012777 010037 156742 MOV #ALU+BIT4+BIT3+BIT2+BIT1+BIT0,@DVSFR
4552 022436 104415 ROMCLK          : F=A
4553 022440 012777 005400 156732 MOV #BIT11+BIT9+BIT8,@DVSFR : LOAD BRANCH POINT
4554 022446 017704 156716 MOV @DVLCR,R4 : READ BRANCH TEST POINT
4555 022452 042704 177774 BIC #^C<BIT1+BIT0>,R4 : CLEAR JUNK
4556 022456 012705 000002 MOV #BIT1,R5 : SET EXPECTED
4557 022462 020504 CMP R5,R4 : BR POINTS CORRECT?
4558 022464 001401 BEQ 72$ : BR IF YES
4559 022466 104006 HLT 6 : BRANCH POINTS WRONG
4560 022470 104412 72$: MSTCLR
4561 022472 012777 000010 156662 MOV #BIT3,@DVSCR : SET SOURCE SEL.
4562 022500 012777 005400 156672 MOV #BIT11+BIT9+BIT8,@DVSFR : RESET DV11
4563 022506 017704 1566$ MOV @DVLCR,R4 : LOAD BRANCH. POINT TEST
4564 022512 042704 1777$ BIC #^C<BIT1+BIT0>,R4 : READ BR POINTS
4565 022516 012705 000003 MOV #BIT1+BIT0,R5 : CLEAR JUNK
4566 022522 020504 CMP R5,R4 : SET EXPECTED
4567 022524 001401 BEQ 73$ : BR POINT OK?
4568 022526 104006 HLT 6 : BR IF YES
4569 022530 73$: : BR POINTS WRONG
4570      . *BRANCH 'A' TEST OF ALU 02
4571      : *
4572 022530 104412 MSTCLR          : RESET DV11
4573 022532 012777 000010 156622 MOV #BIT3,@DVSCR : SET SOURCE SEL
4574 022540 012777 000004 156630 MOV #BIT2,@DVSRA : LOAD DATA
4575 022546 012777 020000 156624 MOV #RAM,@DVSFR  : DO A 'RAM READ'
4576 022554 104415 ROMCLK          : EXECUTE
4577 022556 012777 030261 156614 MOV #XFR+BIT7+BIT5+BIT4+BIT0,@DVSFR : XFR RAM OUTPUT TO 'A' REG
4578 022564 104415 ROMCLK          : F=A
4579 022566 012777 010037 156604 MOV #ALU+BIT4+BIT3+BIT2+BIT1+BIT0,@DVSFR
4580 022574 104415 ROMCLK          : F=A
4581 022576 012777 006000 156574 MOV #BIT11+BIT10,@DVSFR : LOAD BRANCH POINT
4582 022604 017704 156560 MOV @DVLCR,R4 : READ BRANCH TEST POINT
4583 022610 042704 177774 BIC #^C<BIT1+BIT0>,R4 : CLEAR JUNK
4584 022614 012705 000002 MOV #BIT1,R5 : SET EXPECTED
4585 022620 020504 CMP R5,R4 : BR POINTS CORRECT?
4586 022622 001401 BEQ 74$ : BR IF YES
4587 022624 104006 HLT 6 : BRANCH POINTS WRONG
4588 022626 104412 74$: MSTCLR
4589 022630 012777 000010 156524 MOV #BIT3,@DVSCR : SET SOURCE SEL.
4590 022636 012777 006000 156534 MOV #BIT11+BIT10,@DVSFR : RESET DV11
4591 022644 017704 156520 MOV @DVLCR,R4 : LOAD BRANCH. POINT TEST
4592 022650 042704 177774 BIC #^C<BIT1+BIT0>,R4 : READ BR POINTS
4593 022654 012705 000003 MOV #BIT1+BIT0,R5 : CLEAR JUNK
4594 022660 020504 CMP R5,R4 : SET EXPECTED
4595 022662 001401 BEQ 75$ : BR POINT OK?
4596 022664 104006 HLT 6 : BR IF YES
4597 022666 75$: : BR POINTS WRONG
    
```



4598						:*BRANCH 'A' TEST OF ALU 03
4599						:*
4600	022666	104412				MSTCLR ;RESET DV11
4601	022670	012777	000010	156464		MOV #BIT3,@DVSCR ;SET SOURCE SEL
4602	022676	012777	000010	156472		MOV #BIT3,@DVSRA ;LOAD DATA
4603	022704	012777	020000	156466		MOV #RAM,@DVSFR ;DO A 'RAM READ'
4604	022712	104415				ROMCLK ;EXECUTE
4605	022714	012777	030261	156456		MOV #XFR+BIT7+BIT5+BIT4+BIT0,@DVSFR
4606	022722	104415				ROMCLK ;XFR RAM OUTPUT TO 'A' REG
4607	022724	012777	010037	156446		MOV #ALU+BIT4+BIT3+BIT2+BIT1+BIT0,@DVSFR
4608	022732	104415				ROMCLK ;F=A
4609	022734	012777	006400	156436		MOV #BIT11+BIT10+BIT8,@DVSFR ;LOAD BRANCH POINT
4610	022742	017704	156422			MOV @DVLCR,R4 ;READ BRANCH TEST POINT
4611	022746	042704	177774			BIC #^C<BIT1+BIT0>,R4 ;CLEAR JUNK
4612	022752	012705	000002			MOV #BIT1,R5 ;SET EXPECTED
4613	022756	020504				CMP R5,R4 ;BR POINTS CORRECT?
4614	022760	001401				BEQ 76\$ ;BR IF YES
4615	022762	104006				HLT 6 ;BRANCH POINTS WRONG
4616	022764	104412			76\$:	MSTCLR
4617	022766	012777	000010	156366		MOV #BIT3,@DVSCR ;SET SOURCE SEL.
4618	022774	012777	006400	156376		MOV #BIT11+BIT10+BIT8,@DVSFR ;RESET DV11
4619	023002	017704	156362			MOV @DVLCR,R4 ;LOAD BRANCH. POINT TEST
4620	023006	042704	177774			BIC #^C<BIT1+BIT0>,R4 ;READ BR POINTS
4621	023012	012705	000003			MOV #BIT1+BIT0,R5 ;CLEAR JUNK
4622	023016	020504				CMP R5,R4 ;SET EXPECTED
4623	023020	001401				BEQ 77\$ ;BR POINT OK?
4624	023022	104006				HLT 6 ;BR IF YES
4625	023024				77\$:	HLT 6 ;BR POINTS WRONG
4626						:*BRANCH 'A' TEST OF ALU 04
4627						:*
4628	023024	104412				MSTCLR ;RESET DV11
4629	023026	012777	000010	156326		MOV #BIT3,@DVSCR ;SET SOURCE SEL
4630	023034	012777	000020	156334		MOV #BIT4,@DVSRA ;LOAD DATA
4631	023042	012777	020000	156330		MOV #RAM,@DVSFR ;DO A 'RAM READ'
4632	023050	104415				ROMCLK ;EXECUTE
4633	023052	012777	030261	156320		MOV #XFR+BIT7+BIT5+BIT4+BIT0,@DVSFR
4634	023060	104415				ROMCLK ;XFR RAM OUTPUT TO 'A' REG
4635	023062	012777	010037	156310		MOV #ALU+BIT4+BIT3+BIT2+BIT1+BIT0,@DVSFR
4636	023070	104415				ROMCLK ;F=A
4637	023072	012777	007000	156300		MOV #BIT11+BIT10+BIT9,@DVSFR ;LOAD BRANCH POINT
4638	023100	017704	156264			MOV @DVLCR,R4 ;READ BRANCH TEST POINT
4639	023104	042704	177774			BIC #^C<BIT1+BIT0>,R4 ;CLEAR JUNK
4640	023110	012705	000002			MOV #BIT1,R5 ;SET EXPECTED
4641	023114	020504				CMP R5,R4 ;BR POINTS CORRECT?
4642	023116	001401				BEQ 78\$ ;BR IF YES
4643	023120	104006				HLT 6 ;BRANCH POINTS WRONG
4644	023122	104412			78\$:	MSTCLR
4645	023124	012777	000010	156230		MOV #BIT3,@DVSCR ;SET SOURCE SEL.
4646	023132	012777	007000	156240		MOV #BIT11+BIT10+BIT9,@DVSFR ;RESET DV11
4647	023140	017704	156224			MOV @DVLCR,R4 ;LOAD BRANCH. POINT TEST
4648	023144	042704	177774			BIC #^C<BIT1+BIT0>,R4 ;READ BR POINTS
4649	023150	012705	000003			MOV #BIT1+BIT0,R5 ;CLEAR JUNK
4650	023154	020504				CMP R5,R4 ;SET EXPECTED
4651	023156	001401				BEQ 79\$ ;BR POINT OK?
4652	023160	104006				HLT 6 ;BR IF YES
4653	023162				79\$:	HLT 6 ;BR POINTS WRONG

4654 023162 104400

SCOPE

4655  
4656  
4657  
4658  
4659  
4660  
4661  
4662  
4663  
4664  
4665  
4666  
4667  
4668  
4669  
4670  
4671  
4672  
4673  
4674  
4675  
4676  
4677  
4678  
4679  
4680  
4681  
4682  
4683  
4684  
4685  
4686  
4687  
4688  
4689  
4690  
4691  
4692  
4693  
4694  
4695  
4696  
4697  
4698  
4699  
4700  
4701  
4702  
4703  
4704  
4705  
4706  
4707  
4708  
4709

```
***** TEST 101 *****  
: *TEST OF BRANCH 'B' 'RAM OUTPUT 0-14-0'.  
: *TEST TO A RAM READ AND 'FLOAT' A '1' FROM  
: *RAM 0 TO 14 ; EXPECTING 'RAM 014=0' TO BE FALSE.  
: *THEN THE '1' IS SHIFTED INTO BIT15 AND  
: *'RAM 0-14=0' SHOULD BE FALSE.  
: *THIS ALSO TEST 'BRB' [RAM OUTPUT BIT15] TRUE.  
: *****
```

: TEST 101

```
-----  
TST101: MOV #101,TSTNO  
MOV #TST102,NEXT  
MSTCLR ;RESET DV11  
MOV #1,R3 ;SET DATA  
MOV #BRB+BIT11+BIT10,R2 ;BRB 'RAM OUTPUT 0-14-0'?  
MOV #BIT3,@DVSFR ;SET SOURCE SEL.  
MOV R2,@DVSFR ;LOAD BRB TEST  
MOV @DVLCR,R4 ;READ TEST POINTS  
MOV #BIT0,R5 ;SET EXPECTED  
BIC #^C<BIT1+BIT0>,R4 ;CLEAR JUNK  
CMP R5,R4 ;TRUE?  
BEQ .+4 ;BR IF YES  
HLT 6 ;RAM OUTPUT 0-14 NOT TRUE AFTER INIT  
MOV #BIT1+BIT0,R5 ;SET EXPECTED  
1$: MOV R3,@DVSRA ;LOAD DATA  
MOV #RAM,@DVSFR ;ISSUE RAM READ INSTR  
ROMCLK ;  
MOV R2,@DVSFR ;BRB TEST  
MOV @DVLCR,R4 ;READ BR TEST POINTS  
BIC #^C<BIT1+BIT0>,R4 ;CLEAR JUNK  
TST R3 ;IS 15=1 IN DATA?  
BPL 2$ ;BR IF NO  
BIC #BIT1,R5 ;IF 15=1 - 0-14=TRUE  
2$: CMP R5,R4 ;BRB TEST POINT OK?  
BEQ 3$ ;BR IF YES  
HLT 6 ;BAD TEST POINT  
3$: CLC ;CLEAR CARRY  
ROL R3 ;MOV BIT TO NEXT RAM POSITION  
1$ BNE 1$ ;HAVE ALL BITS BEEN TESTED?  
MOV #BIT0,R5 ;SET EXPECTED RESULTS  
MOV #BRB+BIT11+BIT9+BIT8,@DVSFR ;BRB 'RAM OUTPUT 15H'  
MOV @DVLCR,R4 ;BRB 'RAM OUTPUT 15H'  
BIC #^C<BIT1+BIT0>,R4 ;  
CMP R5,R4 ;BRANCH RESULTS OK?  
BEQ 4$ ;BR IF YES  
MOV @DVSFR,R2 ;SAVE DVSFR FOR TYPEOUT  
HLT 6 ;'RAM OUTPUT 15H' S/B TRUE  
4$: SCOPE ;SCOPE TESTS
```

```
***** TEST 102 *****  
: *TEST OF THE RAM WRITE OPERATION.
```

```

4710                                     : *WRITE ALL SECONDARY REGISTERS FOR ALL LINES
4711                                     : *WITH DIFFERENT DATA E USING THE ROM
4712                                     : *AND VERIFY THE DATA BY THE UNIBUS.
4713                                     : *****
4714
4715                                     : TEST 102
4716                                     : -----
4717 023364 012737 000102 001226 TST102: MOV #102,TSTNO
4718 023372 012737 024000 001216 MOV #TST103,NEXT
4719 023400 104412 MSTCLR ;CLEAR ALL DV11 REGISTERS
4720 023402 012777 000010 155752 MOV #BIT3,@DVSCR ;SET SOURCE SEC
4721 023410 013700 001400 MOV DVSFR,R0 ;SET DVSFR POINTER IN R0
4722 023414 012702 010077 MOV #ALU+BIT5+BIT4+BIT3+BIT2+BIT1+BIT0,R2
4723 ;FUNCTION 'F=A+1'
4724 023420 012703 020760 MOV #RAM+BIT8+BIT7+BIT6+BIT5+BIT4,R3
4725 ;RAM WRITE FROM ALU RESULT.
4726 023424 012704 030361 MOV #XFR+BIT7+BIT6+BIT5+BIT4+BIT0,R4
4727 ;MOVE ALU RESULT TO 'A' REG.
4728 023430 005005 CLR R5 ;CLEAR LINE NUMBER COUNTER
4729 023432 005001 CLR R1 ;ZERO SEC REG POINTER
4730 023434 110577 155732 1$: MOV#B R5,@DVSR5 ;LOAD LINE
4731 023440 110177 155730 2$: MOV#B R1,@DVSRSH ;LOAD SEC REG.
4732 023444 012777 177777 155724 MOV #-1,@DVSR A ;SET 'FOOT PRINT'
4733 023452 042703 000017 BIC #17,R3 ;CLEAR LINER
4734 023456 050103 BIS R1,R3 ;SET LINE
4735 023460 010310 MOV R3,(R0) ;DO 'RAM WRITE'
4736 023462 104415 ROMCLK ;
4737 023464 012777 077000 155706 MOV #BRB+BIT11+BIT10+BIT9,@DVSRF
4738 023472 010546 MOV R5,-(SP) ;SAVERS
4739 023474 010446 MOV R4,-(SP) ;SAVE R4
4740 023476 010246 MOV R2,-(SP) ;SAVE R2
4741 023500 012705 000001 MOV #BIT0,R5 ;EXPECTED
4742 023504 017704 155660 MOV @DVLCR,R4 ;READ BR. RESULT
4743 023510 042704 177774 BIC #^C<BIT1+BIT0>,R4 ;STRIP JUNK
4744 023514 020504 CMP R5,R4 ;WRITE INHIBIT TRUE?
4745 023516 001401 BEQ .+4 ;
4746 023520 104006 HLT 6 ;WRITE INHIBIT FAILED
4747 023522 022777 177777 155646 CMP #-1,@DVSR A ;WAS WRITE
4748 023530 001401 BEQ .+4 ;REALLY
4749 023532 104000 HLT 0 ;INHIBITED?
4750 023534 012602 MOV (SP)+,R2 ;RESTORE
4751 023536 012604 MOV (SP)+,R4 ;REGISTERS
4752 023540 012605 MOV (SP)+,R5 ;
4753 023542 012710 020000 MOV #RAM,(R0) ;PLACE RAM INSTR IN SFR
4754 023546 050110 BIS R1,(R0) ;PLACE SEC REG POINTER IN SFR
4755 023550 104415 ROMCLK ;EXECUTE INSTR.
4756 023552 010210 MOV R2,(R0) ;'F=A+1'
4757 023554 104415 ROMCLK ;EXECUTE
4758 023556 042703 000017 BIC #17,R3 ;CLEAR SEC REG POINTER
4759 023562 050103 BIS R1,R3 ;SET SEC REG POINTER
4760 023564 010310 MOV R3,(R0) ;RAM WRITE FROM ALU RESULT
4761 023566 104415 ROMCLK ;EXECUTE
4762 023570 012777 077000 155602 MOV #BRB+BIT11+BIT10+BIT9,@DVSRF
4763 023576 010546 MOV R5,-(SP) ;TEST WRITE
4764 023600 010446 MOV R4,-(SP) ;INHIBIT
4765 023602 010246 MOV R2,-(SP) ;FALSE.
    
```

```
4766 023604 017702 155570      MOV      @DVSFR,R2
4767 023610 012705 000003      MOV      #BIT1+BIT0,R5      :EXPECTED
4768 023614 017704 155550      MOV      @DVLCR,R4         :READ RESULTS
4769 023620 042704 177774      BIC      #^C<BIT1+BIT0>.R4 :STRIP JUNK
4770 023624 020504           CMP      R5,R4             :GOOD?
4771 023626 001401           BEQ      .+4               :
4772 023630 040006           HLT      6                 :WRITE INHIBIT S/B FALSE
4773 023632 012602           MOV      (SP)+,R2          :RESTORE
4774 023634 012604           MOV      (SP)+,R4          :REGISTERS
4775 023636 012605           MOV      (SP)+,R5          :
4776 023640 010410           MOV      R4,(R0)           :MOVE ALU RESULT TO A REG (UPDATED DATA)
4777 023642 104415           ROMCLK                    :EXECUTE
4778 023644 005201           INC      R1                 :UPDATE SEC REG POINTER
4779 023646 022701 000020           CMP      #16.,R1           :ALL REGISTERS DONE?
4780 023652 001270           BNE     2$                 :BR IF NO
4781 023654 010046           MOV      R0,-(SP)          :SAVE R0
4782 023656 004537 031676           PERFORM ,SETSCAN          :UPDATE MSCAN
4783 023662 000001           1                          : (LINE NO. UPDATE)
4784 023664 012600           MOV      (SP)+,R0          :RESTORE R0
4785 023666 012710 030124           MOV      #XFR+BIT6+BIT4+BIT2,(R0)
4786 023672 104415           ROMCLK                    :XFR MSCAN TO RAR 0-3 CLOCK
4787 023674 005205           INC      R5                 :UPDATE LINE NUMBER COUNTER.
4788 023676 022705 000020           CMP      #16.,R5           :ALL LINES DONE?
4789 023702 001253           BNE     1$                 :BR IF NO
4790 023704 005000           3$: CLR      R0              :CLEAR SEC REG POINTER
4791 023706 005037 001246           CLR      TEMP1             :CLEAR LINE NUMBER POINTER
4792 023712 013702 001372           MOV      DVSRS,R2          :SET SRC POINTER (LINE SEL)
4793 023716 013703 001374           MOV      DVSRSR,R3         :SET HIGH BYTE POINTER (SEC REG SEL)
4794 023722 013701 001376           MOV      DVSRA,R1          :SET SRA POINTER (ACCESS REG)
4795 023726 012705 000001           MOV      #1,R5             :SET EXPECTED DATA
4796 023732 110013           4$: MOV      R0,(R3)         :LOAD SEC REG SEL
4797 023734 113712 001246           MOV      TEMP1,(R2)        :LOAD LINE NO.
4798 023740 011104           MOV      (R1),R4           :READ RAM RESULT
4799 023742 020504           CMP      R5,R4             :WAS RAM 'WRITTEN' OCCRECTLY?
4800 023744 001401           BEQ     5$                 :BR IF RAM DATA OK
4801 023746 104006           HLT     6                   :RAM WAS 'WRITTEN' INCORRECTLY
4802 023750 122025           5$: CMP      (R0)+,(R5)+    :UPDATE SEC REG POINTER AND DATA EXPECTED
4803 023752 022700 000020           CMP      #16.,R0           :ALL SEC REGISTERS DONE?
4804 023756 001365           BNE     4$                 :BR IF NO
4805 023760 005000           CLR      R0                :ZERO SEC REG POINTER
4806 023762 005237 001246           INC      TEMP1             :UPDATE LINE NUMBER POINTER
4807 023766 023727 001246 000020           CMP      TEMP1,#16.        :ALL LINES DONE?
4808 023774 001356           BNE     4$                 :BR IF NO
4809 023776 104400           SCOPE                      :SCOPE TEST
```

```
4810
4811 :***** TEST 103 *****
4812 :*BASIC TEST FOR THE 'BCC OPERATION'
4813 :*POLYNOMIAL SELECTION TABLE:
4814 :*RAM OUTPUT BIT04 BIT03      POLY
4815 :*          0      0          LRC 8
4816 :*          0      1          CRC 16
4817 :*          1      1          CRC CCITT
4818 :*****
4819
4820 :***** TEST 103 *****
4821 :*TEST OF LRC 8.
```

4822  
4823  
4824  
4825  
4826  
4827  
4828  
4829  
4830  
4831  
4832  
4833  
4834  
4835  
4836  
4837  
4838  
4839  
4840  
4841  
4842  
4843  
4844  
4845  
4846  
4847  
4848  
4849  
4850  
4851  
4852  
4853  
4854  
4855  
4856  
4857  
4858  
4859  
4860  
4861  
4862  
4863  
4864  
4865  
4866  
4867  
4868  
4869  
4870  
4871  
4872  
4873  
4874  
4875  
4876  
4877

024000 012737 000103 001226  
024006 012737 024254 001216  
024014 104412  
024016 012777 000010 155336  
024024 013700 001400  
024030 012702 030346  
024034 012710 060000  
024040 104415  
024042 010210  
024044 104415  
024046 005005  
024050 017704 155312  
024054 001401  
024056 104006  
024060 012710 010034  
024064 104415  
024066 012710 030362  
024072 104415  
024074 012710 060000  
024100 104415  
024102 010210  
024104 104415  
024106 012705 000377  
024112 017704 155250  
024116 020504  
024120 001401  
024122 104006  
024124 104412  
024126 012777 000010 155226  
024134 012710 010034  
024140 104415  
024142 012710 030361  
024146 104415  
024150 012710 060000  
024154 104415  
024156 010210  
024160 104415  
024162 017704 155200  
024166 020504  
024170 001401  
024172 104006  
024174 104412  
024176 012777 000010 155156  
024204 012710 010034  
024210 104415

```
;*PATTERNS ARE:
;*A REG B REG BCC (EXPECTED)
;* 0'S 0'S 0'S
;* 0'S 1'S 1'S
;* 1'S 0'S 1'S
;* 1'S 1'S 0'S
:*****

: TEST 103
-----
TST103: MOV #103,TSTNO
MOV #TST104,NEXT
MSTCLR ;RESET DV11
MOV #BIT3,@DVSCR ;SET SOURCE SEL
MOV DVSFR,R0 ;SET DVSFR POINTER
MOV #XFR+BIT7+BIT6+BIT5+BIT2+BIT1,R2
MOV #BCC,(R0) ;DO A 'BCC' INSTRUCTION USING 'LRC8'
ROMCLK ;EXECUTE
MOV R2,(R0) ;XFR BCC REG TO DVRIC
ROMCLK ;EXECUTE
CLR R5 ;SET EXPECTED TO 0
MOV @DVRIC,R4 ;READ RESULTS OF BCC REG.
BEQ 1$ ;BR IF =0
HLT 6 ;BCC REG NOT =0
1$: MOV #ALU+BIT4+BIT3+BIT2,(R0)
ROMCLK ;ALU OPR 'F=-1'
MOV #XFR+BIT7+BIT6+BIT5+BIT4+BIT1,(R0)
ROMCLK ;XFR ALU RESULT TO B REGISTER
MOV #BCC,(R0) ;DO A 'BCC' OPR
ROMCLK ;EXECUTE
MOV R2,(R0) ;XFR BCC REG TO DVRIC
ROMCLK ;EXECUTE
MOV #377,R5 ;SET EXPECTED RESULTS -ALL 1'S [LOW BYTE]
MOV @DVRIC,R4 ;READ RESULTS
CMP R5,R4 ;DID BCC OPR WORK
BEQ 2$ ;BR IF OK
HLT 6 ;EITHER 'BCC' OPR FAILED OR BIT(S) DROPPED IN LOW BYTE 0
2$: MSTCLR ;RESET INTERNAL REG (BCC,ALU,B)
MOV #BIT3,@DVSCR ;SET SOURCE SEL.
MOV #ALU+BIT4+BIT3+BIT2,(R0)
ROMCLK ;ALU 'F=-1'
MOV #XFR+BIT7+BIT6+BIT5+BIT4+BIT0,(R0)
ROMCLK ;XFR ALU RESULT TO 'A' REGISTER
MOV #BCC,(R0) ;BCC OPERATION
ROMCLK ;EXECUTE
MOV R2,(R0) ;XFR BCC TO DVRIC
ROMCLK ;EXECUTE
MOV @DVRIC,R4 ;READ RESULTS
CMP R5,R4 ;BCC REG S/B =377
BEQ 3$ ;BR IF OK
HLT 6 ;BCC REG NOT =377
3$: MSTCLR ;READ OF DATA FROM 'A' REG LEG FAILED
MOV #BIT3,@DVSCR ;CLEAR INTERNAL REGISTERS
MOV #ALU+BIT4+BIT3+BIT2,(R0) ;SET SOURCE SEL
ROMCLK ;ALU 'F=-1'
```

```

4878 024212 012710 030361      MOV      #XFR+BIT7+BIT6+BIT5+BIT4+BIT0,(R0)
4879 024216 104415              ROMCLK      ;XFR ALU RESULT TO 'A' REG
4880 024220 012710 030362      MOV      #XFR+BIT7+BIT6+BIT5+BIT4+BIT1,(R0)
4881 024224 104415              ROMCLK      ;XFR ALU RESULT TO 'B' REG
4882 024226 012710 060000      MOV      #BCC,(R0) ;SO BCC OPR
4883 024232 104415              ROMCLK      ;
4884 024234 010210              MOV      R2,(R0) ;XFR BCC TO DVRIC
4885 024236 104415              ROMCLK      ;ROMCLK
4886 024240 005005              CLR      R5 ;EXPECT 0'S
4887 024242 017704 155120      MOV      @DVRIC,R4 ;READ BCC RESULT
4888 024246 001401              BEQ      4$ ;BR IF =0
4889 024250 104006              HLT      6 ;BCC 'LRC' XOR TEST FAILED
4890 024252 104400      4$:      SCOPE ;SCOPE TEST

```

```

:***** TEST 104 *****
:*TEST OF POLYNOMIAL 'CRC 16'
:*TEST THAT BITS 9-13 OF THE 'B' REG APPEAR
:*IN BITS 1-5 OF THE BCC REG.
:*****

```

: TEST 104

```

4891
4892
4893
4894
4895
4896
4897
4898
4899
4900 024254 012737 000104 001226  TST104: MOV      #104,TSTNO
4901 024262 012737 024446 001216      MOV      #TST105,NEXT
4902 024270 104412              MSTCLR      ;RESET DV11
4903 024272 012777 000010 155062      MOV      #BIT3,@DVSCR ;SET SOURCE SEL
4904 024300 013700 001400              MOV      DVSFR,R0 ;SET DVSFR POINTER
4905 024304 012777 001000 155064      MOV      #BIT9,@DVSRA ;LOAD 'DATA'
4906 024312 012710 020000      MOV      #RAM,(R0) ;RAM READ
4907 024316 104415              ROMCLK      ;EXECUTE
4908 024320 012710 030261      MOV      #XFR+BIT7+BIT5+BIT4+BIT0,(R0)
4909 024324 104415              ROMCLK      ;XFR RAM OUTPUT TO 'A' REG.
4910 024326 012710 030262      MOV      #XFR+BIT7+BIT5+BIT4+BIT1,(R0)
4911 024332 104415              ROMCLK      ;XFR RAM OUTPUT TO 'B' REG.
4912 024334 012777 000010 155034      MOV      #BIT3,@DVSRA ;SELECT POLYNOMIAL 'CRC16'
4913 024342 012710 020000      MOV      #RAM,(R0) ;READ DATA
4914 024346 104415              ROMCLK      ;
4915 024350 012705 001000      1$:      MOV      #BIT9,R5 ;SET EXPECTED RESULTS
4916 024354 012710 060000      2$:      MOV      #BCC,(R0) ;BCC OPR 'CRC16'
4917 024360 104415              ROMCLK      ;EXECUTE
4918 024362 012710 030346      MOV      #XFR+BIT7+BIT6+BIT5+BIT2+BIT1,(R0)
4919 024366 104415              ROMCLK      ;XFR BCC REG TO DVRIC
4920 024370 017704 154772      MOV      @DVRIC,R4 ;READ RESULTS
4921 024374 012703 000010      MOV      #8.,R3 ;PREPARE
4922 024400 000241      3$:      CLC ;TO
4923 024402 006104              ROL      R4 ;POSITION
4924 024404 005303              DEC      R3 ;RESULT
4925 024406 001374              BNE      3$ ;OF BCC OPERATION
4926 024410 020504              CMP      R5,R4 ;DID CRC16 WORK?
4927 024412 001401              BEQ      4$ ;BR IF YES
4928 024414 104006              HLT      6 ;INCORRECT BCC RESULTS
4929 024416 012710 010026      4$:      MOV      #ALU+BIT4+BIT2+BIT1,(R0)
4930 024422 104415              ROMCLK      ;ALU 'F=A+B'
4931 024424 012710 030362      MOV      #XFR+BIT7+BIT6+BIT5+BIT4+BIT1,(R0)
4932 024430 104415              ROMCLK      ;XFR ALU RESULT TO 'B' REG
4933 024432 062705 001000      ADD      #BIT9,R5 ;UPDATE DATA COMPARE

```

4934	024436	032705	040000	BIT	#BIT14,R5	:ALL DATA DONE?
4935	024442	001744		BFO	2\$	:BR IF NO
4936	024444	104400		SCOPE		:SCOPE TEST

4937  
4938  
4939  
4940  
4941  
4942  
4943  
4944  
4945  
4946  
4947  
4948

```

:***** TEST 105 *****
:*TEST OF THE BCC OPERATION USING
:*USING CRC16 FOR THE POLYNOMIAL
:*SPECIFIC DATA PATTERNS ARE USED TO
:*ISOLATE FAULTS AS SOON AS POSSIBLE
:*****

```

: TEST 105

4949	024446	012737	000105	001226	TST105.	MOV	#105,TSTNO	
4950	024454	012737	026102	001216		MOV	#TST106,NEXT	
4951	024462	012737	120001	032202		MOV	#CRC16,XPOLY	:SET POLYNOMIAL
4952	024470	013700	001400			MOV	DVSFR,R0	:SET REGISTER POINTER
4953	024474	012702	030346			MOV	#XFR+BIT7+BIT6+BIT5+BIT2+BIT1,R2	:SET FOR XFER FROM BCC TO DVRIC
4954								
4955	024500	012737	024506	001220		MOV	#1\$,LOCK	:SET IF SW09=1
4956	024506	104412			1\$:	MSTCLR		:ISSUE A MSTCLR
4957	024510	012777	000010	154644		MOV	#BIT3,@DVSCR	:SET SOURCE SEL
4958	024516	005037	032206			CLR	CALBCC	:ZERO CALCULATED BCC (SOFTWARE)
4959	024522	004537	032030			JSR	R5,SIMBCC	:GO AND CALCULATE A NEW BCC
4960	024526	000010				8.		:SHIFTS PERFORMED
4961	024530	000000				0		:DATA CHAR
4962	024532	000000				0		:PREVIOUS BCC
4963	024534	004337	031754			JSR	R3,L.DATA	:GO AND LOAD DATA INTO THE DV11
4964	024540	000000				0		:DATA TO THE A REGISTER
4965	024542	000000				0		:DATA TO THE B REGISTER
4966	024544	000010				10		:POLY SELT. (RAMOUTPUT)
4967	024546	012710	060000			MOV	#BCC,(R0)	
4968	024552	104415				ROMCLK		:DO A BCC CALCULATION (HRDW)
4969	024554	010210				MOV	R2,(R0)	:DO A DATA XFR FROM BCC REG TO DVRIC
4970	024556	104415				ROMCLK		
4971	024560	013705	032206			MOV	CALBCC,R5	:PUT SOFTWARE BCC INTO EXPECTED
4972	024564	017704	154576			MOV	@DVRIC,R4	:PUT HRDW BCC INTO RECEIVERD
4973	024570	020504				CMP	R5,R4	:DOES EXPECTED = FOUND??
4974	024572	001401				BEQ	64\$	:BR IF OK!
4975	024574	104006				HLT	6	:BCC CALCULATION ERROR
4976	024576	104401			64\$:	SCOP1		:SW09=1?
4977	024600	012737	024606	001220		MOV	#2\$,LOCK	:SET IF SW09=1
4978	024606	104412			2\$:	MSTCLR		:ISSUE A MSTCLR
4979	024610	012777	000010	154544		MOV	#BIT3,@DVSCR	:SET SOURCE SEL
4980	024616	005037	032206			CLR	CALBCC	:ZERO CALCULATED BCC (SOFTWARE)
4981	024622	004537	032030			JSR	R5,SIMBCC	:GO AND CALCULATE A NEW BCC
4982	024626	000010				8.		:SHIFTS PERFORMED
4983	024630	000252				#B<10101010>		:DATA CHAR
4984	024632	040000				BIT14		:PREVIOUS BCC
4985	024634	004337	031754			JSR	R3,L.DATA	:GO AND LOAD DATA INTO THE DV11
4986	024640	000252				#B<10101010>		:DATA TO THE A REGISTER
4987	024642	040000				BIT14		:DATA TO THE B REGISTER
4988	024644	000010				10		:POLY SELT. (RAMOUTPUT)
4989	024646	012710	060000			MOV	#BCC,(R0)	





5046	025116	005037	032206			CLR	CALBCC		:ZERO CALCULATED BCC (SOFTWARE)
5047	025122	004537	032030			JSR	R5,SIMBCC		:GO AND CALCULATE A NEW BCC
5048	025126	000010				8.			:SHIFTS PERFORMED
5049	025130	000001				^B<00000001>			:DATA CHAR
5050	025132	000000				0			:PREVIOUS BCC
5051	025134	004337	031754			JSR	R3,L.DATA		:GO AND LOAD DATA INTO THE DV11
5052	025140	000001				^B<00000001>			:DATA TO THE A REGISTER
5053	025142	000000				0			:DATA TO THE B REGISTER
5054	025144	000010				10			:POLY SELT. (RAMOUTPUT)
5055	025146	012710	060000			MOV	#BCC,(R0)		
5056	025152	104415				ROMCLK			:DO A BCC CALCULATION (HRDW)
5057	025154	010210				MOV	R2,(R0)		:DO A DATAFR FROM BCC REG TO DVRIC
5058	025156	104415				ROMCLK			:
5059	025160	013705	032206			MOV	CALBCC,R5		:PUT SOFTWARE BCC INTO EXPECTED
5060	025164	017704	154176			MOV	@DVRIC,R4		:PUT HRDW BCC INTO RECEIVERD
5061	025170	020504				CMP	R5,R4		:DOES EXPECTED = FOUND??
5062	025172	001401				BEQ	68\$		:BR IF OK!
5063	025174	104006				HLT	6		:BCC CALCULATION ERROR
5064	025176	104401			68\$:	SCOP1			:SW09=1?
5065	025200	012737	025206	001220		MOV	#6\$,LOCK		:SET IF SW09=1
5066	025206	104412			6\$:	MSTCLR			:ISSUE A MSTCLR
5067	025210	012777	000010	154144		MOV	#BIT3,@DVSCK		:SET SOURCE SEL
5068	025216	005037	032206			CLR	CALBCC		:ZERO CALCULATED BCC (SOFTWARE)
5069	025222	004537	032030			JSR	R5,SIMBCC		:GO AND CALCULATE A NEW BCC
5070	025226	000010				8.			:SHIFTS PERFORMED
5071	025230	000004				^B<00000100>			:DATA CHAR
5072	025232	000000				0			:PREVIOUS BCC
5073	025234	004337	031754			JSR	R3,L.DATA		:GO AND LOAD DATA INTO THE DV11
5074	025240	000004				^B<00000100>			:DATA TO THE A REGISTER
5075	025242	000000				0			:DATA TO THE B REGISTER
5076	025244	000010				10			:POLY SELT. (RAMOUTPUT)
5077	025246	012710	060000			MOV	#BCC,(R0)		
5078	025252	104415				ROMCLK			:DO A BCC CALCULATION (HRDW)
5079	025254	010210				MOV	R2,(R0)		:DO A DATAFR FROM BCC REG TO DVRIC
5080	025256	104415				ROMCLK			:
5081	025260	013705	032206			MOV	CALBCC,R5		:PUT SOFTWARE BCC INTO EXPECTED
5082	025264	017704	154076			MOV	@DVRIC,R4		:PUT HRDW BCC INTO RECEIVERD
5083	025270	020504				CMP	R5,R4		:DOES EXPECTED = FOUND??
5084	025272	001401				BEQ	69\$		:BR IF OK!
5085	025274	104006				HLT	6		:BCC CALCULATION ERROR
5086	025276	104401			69\$:	SCOP1			:SW09=1?
5087	025300	012737	025306	001220		MOV	#7\$,LOCK		:SET IF SW09=1
5088	025306	104412			7\$:	MSTCLR			:ISSUE A MSTCLR
5089	025310	012777	000010	154044		MOV	#BIT3,@DVSCK		:SET SOURCE SEL
5090	025316	005037	032206			CLR	CALBCC		:ZERO CALCULATED BCC (SOFTWARE)
5091	025322	004537	032030			JSR	R5,SIMBCC		:GO AND CALCULATE A NEW BCC
5092	025326	000010				8.			:SHIFTS PERFORMED
5093	025330	000020				^B<00010000>			:DATA CHAR
5094	025332	000000				0			:PREVIOUS BCC
5095	025334	004337	031754			JSR	R3,L.DATA		:GO AND LOAD DATA INTO THE DV11
5096	025340	000020				^B<00010000>			:DATA TO THE A REGISTER
5097	025342	000000				0			:DATA TO THE B REGISTER
5098	025344	000010				10			:POLY SELT. (RAMOUTPUT)
5099	025346	012710	060000			MOV	#BCC,(R0)		
5100	025352	104415				ROMCLK			:DO A BCC CALCULATION (HRDW)
5101	025354	010210				MOV	R2,(R0)		:DO A DATAFR FROM BCC REG TO DVRIC





```

5214 026064 017704 153276      MOV    @DVRIC,R4      ;PUT HRDW BCC INTO RECEIVERD
5215 026070 020504              CMP    R5,R4         ;DOES EXPECTED - FOUND??
5216 026072 001401              BEQ    75$           ;BR IF OK.
5217 026074 104006              HLT    6             ;BCC CALCULATION ERROR
5218 026076 104401              75$: SCOPE1         ;SW09-1?
5219 026100 104400              SCOPE
    
```

```

5220
5221
5222      ;***** TEST 106 *****
5223      ;*TEST TO RUN A BINARY COUNT (000-377)PATTERN
5224      ;*THROUGH THE BCC GENERATION LOGIC.
5225      ;*THE POLYNOMIAL USED WILL BE LRC8 .
5226      ;*THE BCC REGISTER WILL BE BUILT UP AFTER
5227      ;*EACH CHARACTER -*NOT ZEROED OUT*-
5228      ;*****
    
```

```

5230      ; TEST 106
5231      ;-----
5232 026102 012737 000106 001226 1$T106: MOV    #106,TSTNO
5233 026110 012737 026260 001216      MOV    #TST107,NEXT
5234 026116 012737 000200 032202      MOV    #LRC8,XPOLY      ;LOAD POLYNOMIAL FOR SOFTWARE CAL.
5235 026124 012702 030346              MOV    #XFR+BIT7+BIT6+BIT5+BIT2+BIT1,R2
5236 026130 013700 001400              MOV    DVSFR,R0         ;LOAD DATA XFER FROM BCC TO DVRIC
5237 026134 005001              CLR    R1               ;SET DATA POINTER TO 0
5238 026136 005037 032206              CLR    CALBCC          ;ZERO SOFTWARE BCC
5239 026142 104412              1$: MSTCLR            ;CLEAR THE DV11
5240 026144 012777 000010 153210      MOV    #BIT3,@DVSCR    ;SET SOURCE SEL
5241 026152 010137 026204              MOV    R1,2$          ;LOAD SOFTWARE CHAR
5242 026156 010137 026214              MOV    R1,4$          ;LOAD HRDW CHAR
5243 026162 013737 032206 026206      MOV    CALBCC,3$      ;PLACE PREVIOUS BCC FOR SOFTWARE
5244 026170 013737 032206 026216      MOV    CALBCC,5$      ;PLACE PREVIOUS BCC FOR HRDW
5245 026176 004537 032030              JSR    R5,SIMBCC      ;HAVE SOFTWARE GET THE RIGHT BCC
5246 026202 000010              8.                   ;EIGHT SHIFTS
5247 026204 000001              2$: .BLKW 1           ;DATA
5248 026206 000001              3$: .BLKW 1           ;PREVIOUS BCC
5249 026210 004337 031754              JSR    R3,L.DATA     ;LOAD DV11 REGISTERS
5250 026214 000001              4$: .BLKW 1           ;TO BE PLACED INTO THE 'A' REG
5251 026216 000001              5$: .BLKW 1           ;TO BE PLACED INTO THE 'B' REG
5252 026220 000000              0.                   ;TO BE LEFT IN THE RAM OUTPUT REG
5253 026222 012710 060000              MOV    #BCC,(R0)
5254 026226 104415              ROMCLK              ;DO A BCC OPERATION
5255 026230 010210              MOV    R2,(R0)       ;DO A DATA XFER OPR.
5256 026232 104415              ROMCLK              ;
5257 026234 013705 032206              MOV    CALBCC,R5     ;GET GOOD BCC
5258 026240 017704 153122              MOV    @DVRIC,R4     ;GET ??? BCC
5259 026244 020504              CMP    R5,R4         ;ARE THEY THE SAME:
5260 026246 001401              BEQ    6$            ;BR IF YES
5261 026250 104006              HLT    6             ;BCC ERROR
5262 026252 105201              6$: INCB    R1        ;UPDATE DATA CHAR
5263 026254 001332              BNE    1$           ;BR IF NOT ALL DATA DONE.
5264 026256 104400              SCOPE                ;SCOPE THIS TEST
    
```

```

5265
5266
5267      ;***** TEST 107 *****
5268      ;*TEST TO RUN A BINARY COUNT (000-377)PATTERN
5269      ;*THROUGH THE BCC GENERATION LOGIC.
    
```

5270  
 5271  
 5272  
 5273  
 5274  
 5275  
 5276  
 5277 026260 012737 000107 001226  
 5278 026266 012737 026436 001216  
 5279 026274 012737 120001 032202  
 5280 026302 012702 030346  
 5281 026306 013700 001400  
 5282 026312 005001  
 5283 026314 005037 032206  
 5284 026320 104412  
 5285 026322 012777 000010 153032  
 5286 026330 010137 026362  
 5287 026334 010137 026372  
 5288 026340 013737 032206 026364  
 5289 026346 013737 032206 026374  
 5290 026354 004537 032030  
 5291 026360 000010  
 5292 026362 000001  
 5293 026364 000001  
 5294 026366 004337 031754  
 5295 026372 000001  
 5296 026374 000001  
 5297 026376 000010  
 5298 026400 012710 060000  
 5299 026404 104415  
 5300 026406 010210  
 5301 026410 104415  
 5302 026412 013705 032206  
 5303 026416 017704 152744  
 5304 026422 020504  
 5305 026424 001401  
 5306 026426 104006  
 5307 026430 105201  
 5308 026432 001332  
 5309 026434 104400

:\*THE POLYNOMIAL USED WILL BE CRC\*6 .  
 :\*THE BCC REGISTER WILL BE BUILT UP AFTER  
 :\*EACH CHARACTER -\*NOT ZEROED OUT\*  
 :\*\*\*\*\*

: TEST 107

```

TST107: MOV #107,TSTNO
        MOV #TST110,NEXT
        MOV #CRC16,XPOLY ;LOAD POLYNOMIAL FOR SOFTWARE CAL.
        MOV #XFR+BIT7+BIT6+BIT5+BIT2+BIT1,R2
        MOV DVSFR,R0 ;LOAD DATA XFER FROM BCC TO DVRIC
        CLR R1 ;SET DATA POINTER TO 0
        CLR CALBCC ;ZERO SOFTWARE BCC
1$: MSTCLR ;CLEAR THE DV11
        MOV #BIT3,@DVSCR ;SET SOURCE SEL
        MOV R1,2$ ;LOAD SOFTWARE CHAR
        MOV R1,4$ ;LOAD HRDW CHAR
        MOV CALBCC,3$ ;PLACE PREVIOUS BCC FOR SOFTWARE
        MOV CALBCC,5$ ;PLACE PREVIOUS BCC FOR HRDW
        JSR R5,SIMBCC ;HAVE SOFTWARE GET THE RIGHT BCC
        8. ;EIGHT SHIFTS
2$: .BLKW 1 ;DATA
3$: .BLKW 1 ;PREVIOUS BCC
        JSR R3,L.DATA ;LOAD DV11 REGISTERS
4$: .BLKW 1 ;TO BE PLACED INTO THE 'A' REG
5$: .BLKW 1 ;TO BE PLACED INTO THE 'B' REG
        10 ;TO BE LEFT IN THE RAM OUTPUT REG
        MOV #BCC,(R0)
        ROMCLK ;DO A BCC OPERATION
        MOV R2,(R0) ;DO A DATA XFER OPR.
        ROMCLK
        MOV CALBCC,R5 ;GET GOOD BCC
        MOV @DVRIC,R4 ;GET ??? BCC
        CMP R5,R4 ;ARE THEY THE SAME?
        BEQ 6$ ;BR IF YES
        HLT 6 ;BCC ERROR
6$: INCB R1 ;UPDATE DATA CHAR
        BNE 1$ ;BR IF NOT ALL DATA DONE.
        SCOPE ;SCOPE THIS TEST
    
```

:\*\*\*\*\* TEST 110 \*\*\*\*\*  
 :\*TEST TO RUN A BINARY COUNT (000-377)PATTERN  
 :\*THROUGH THE BCC GENERATION LOGIC.  
 :\*THE POLYNOMIAL USED WILL BE CRC.CCITT .  
 :\*THE BCC REGISTER WILL BE BUILT UP AFTER  
 :\*EACH CHARACTER -\*NOT ZEROED OUT\*  
 :\*\*\*\*\*

: TEST 110

5320  
 5321  
 5322 026436 012737 000110 001226  
 5323 026444 012737 026614 001216  
 5324 026452 012737 102010 032202  
 5325 026460 012702 030346

```

TST110: MOV #110,TSTNO
        MOV #TST111,NEXT
        MOV #CRC.CCITT,XPOLY ;LOAD POLYNOMIAL FOR SOFTWARE CAL.
        MOV #XFR+BIT7+BIT6+BIT5+BIT2+BIT1,R2
    
```

5326	026464	013700	001400		MOV	DVSFR,R0	:LOAD DATA XFER FROM BCC TO DV11
5327	026470	005001			CLR	R1	:SET DATA POINTER TO 0
5328	026472	005037	032206		CLR	CALBCC	:ZERO SOFTWARE BCC
5329	026476	104412		1\$:	MSTCLR		:CLEAR THE DV11
5330	026500	012777	000010	152654	MOV	#BIT3,@DVSCR	:SET SOURCE SEL
5331	026506	010137	026540		MOV	R1,2\$	:LOAD SOFTWARE CHAR
5332	026512	010137	026550		MOV	R1,4\$	:LOAD HRDW CHAR
5333	026516	013737	032206	026542	MOV	CALBCC,3\$	:PLACE PREVIOUS BCC FOR SOFTWARE
5334	026524	013737	032206	026552	MOV	CALBCC,5\$	:PLACE PREVIOUS BCC FOR HRDW
5335	026532	004537	032030		JSR	R5,SIMBCC	:HAVE SOFTWARE GET THE RIGHT BCC
5336	026536	000010			8.		:EIGHT SHIFTS
5337	026540	000001		2\$:	.BLKW 1		:DATA
5338	026542	000001		3\$:	.BLKW 1		:PREVIOUS BCC
5339	026544	004337	031754		JSR	R3,L.DATA	:LOAD DV11 REGISTERS
5340	026550	000001		4\$:	.BLKW 1		:TO BE PLACED INTO THE 'A' REG
5341	026552	000001		5\$:	.BLKW 1		:TO BE PLACED INTO THE 'B' REG
5342	026554	000030			30		:TO BE LEFT IN THE RAM OUTPUT REG
5343	026556	012710	060000		MOV	#BCC,(R0)	
5344	026562	104415			ROMCLK		:DO A BCC OPERATION
5345	026564	010210			MOV	R2,(R0)	:DO A DATA XFER OPR.
5346	026566	104415			ROMCLK		:
5347	026570	013705	032206		MOV	CALBCC,R5	:GET GOOD BCC
5348	026574	017704	152566		MOV	@DV11,R4	:GET ??? BCC
5349	026600	020504			CMP	R5,R4	:ARE THEY THE SAME?
5350	026602	001401			BEQ	6\$	:BR IF YES
5351	026604	104006			HLT	6	:BCC ERROR
5352	026606	105201		6\$:	INCB	R1	:UPDATE DATA CHAR
5353	026610	001332			BNE	1\$	:BR IF NOT ALL DATA DONE.
5354	026612	104400			SCOPE		:SCOPE THIS TEST
5355							

```

5356
5357
5358
5359
5360
5361
5362
5363
5364 026614 012737 000111 001226
5365 026622 012737 026764 001216
5366 026630 012737 000340 177776
5367 026636 104412
5368 026640 004537 032210
5369 026644 032232
5370 026646 032236
5371 026650 340 340
5372 026652 005037 177776
5373 026656 012777 001200 152476
5374 026664 000240
5375 026666 012777 001100 152466
5376 026674 000240
5377 026676 005077 152460
5378 026702 004537 032210
5379 026706 026746
5380 026710 026760
5381 026712 340 340
5382 026714 052777 001300 152440
5383 026722 000240
5384 026724 104007
5385 026726 004537 032210
5386 026732 032232
5387 026734 032236
5388 026736 340 340
5389 026740 005077 152416
5390 026744 104400
5391 026746 012706 001200
5392 026752 005077 152404
5393 026756 000763
5394 026760 104012
5395 026762 000771
5396
5397
5398
5399
5400
5401
5402
5403
5404 026764 012737 000112 001226
5405 026772 012737 027134 001216
5406 027000 012737 000340 177776
5407 027006 104412
5408 027010 004537 032210
5409 027014 032232
5410 027016 032236
5411 027020 340 340

```

```

:***** TEST 111 *****
:*TEST THAT SETTING BIT9!BIT7 AND BIT9.BIT6
:*RECV IE AND RECV INTR PRODUCE AN INTERUPT ON VECTOR 'A'
:*****

```

: TEST 111

```

-----
TST111: MOV #111,TSTNO
MOV #TST112,NEXT
MOV #340,PS ;LOCK OUT CPU INTERUPTS
MSTCLR ;ISSUE DVRESET
JSR R5,SETVEC ;GO SET VECTOR 'A' AND 'B'
NO.ATRAP ;'A'
NO.BTRAP ;'B'
.BYTE 340,340 ;PRIO. AT 7
CLR PS ;ZERO CPU PRIO.
MOV #BIT9!BIT7,@DVSCR ;SET AN INTERUPT RELATIVE BIT.
NOP ;WAIST TIME
MOV #BIT9!BIT6,@DVSCR ;SET THE ALTERNATE RELATIVE BIT.
NOP ;WAIST
CLR @DVSCR ;ZERO REG
JSR R5,SETVEC ;GO RESET VECTORS 'A' AND 'B'
2$ ;'A'
3$ ;'B'
.BYTE 340,340 ;PRIO. AT 7
BIS #BIT9!BIT7!BIT9!BIT6,@DVSCR ;SET BOTH INTERUPT RELATIVE BITS.
NOP ;SETTING OF THESE BITS FAILED TO PRODUCE AN INTERUPT
HLT 7
1$: JSR R5,SETVEC ;RESET VECTORS
NO.ATRAP ;'A'
NO.BTRAP ;'B'
.BYTE 340,340
CLR @DVSCR ;DSABLE DV11
SCOPE ;SCOPE THIS TEST.
2$: MOV #STACK,SP ;RESET STACK
CLR @DVSCR ;DSABLE DV11
BR 1$ ;RETURN
3$: HLT 12 ;VECTOR HERE WAS WRONG SIDE
BR 2$

```

```

:***** TEST 112 *****
:*TEST THAT SETTING BIT12 AND BIT10
:*STORE IE AND NPR STAT OVFLOW PRODUCE AN INTERUPT ON VECTOR 'B'
:*****

```

: TEST 112

```

-----
TST112: MOV #112,TSTNO
MOV #TST113,NEXT
MOV #340,PS ;LOCK OUT CPU INTERUPTS
MSTCLR ;ISSUE DVRESET
JSR R5,SETVEC ;GO SET VECTOR 'A' AND 'B'
NO.ATRAP ;'A'
NO.BTRAP ;'B'
.BYTE 340,340 ;PRIO. AT 7

```

```
5412 027022 005037 177776 CLR PS ;ZERO CPU PRIO.
5413 027026 012777 010000 152326 MOV #BIT12,@DVSCR ;SET AN INTERRUPT RELATIVE BIT.
5414 027034 000240 NOP ;WAIST TIME
5415 027036 012777 002000 152316 MOV #BIT10,@DVSCR ;SET THE ALTERNATE RELATIVE BIT.
5416 027044 000240 NOP ;WAIST
5417 027046 005077 152310 CLR @DVSCR ;ZERO REG
5418 027052 004537 032210 JSR R5,SETVEC ;GO RESET VECTORS 'A' AND 'B'
5419 027056 027130 3$ ;'A'
5420 027060 027116 2$ ;'B'
5421 027062 340 340 .BYTE 340,340 ;PRIO. AT 7
5422 027064 052777 012000 152270 BIS #BIT12.BIT10,@DVSCR
5423 027072 000240 NOP ;SET BOTH INTERRUPT RELATIVE BITS.
5424 027074 104010 HLT 10 ;SETTING OF THESE BITS FAILED TO PRODUCE AN INTERRUPT
5425 027076 004537 032210 1$: JSR R5,SETVEC ;RESET VECTORS
5426 027102 032232 NO.ATRAP ;'A'
5427 027104 032236 NO.BTRAP ;'B'
5428 027106 340 340 .BYTE 340,340
5429 027110 005077 152246 CLR @DVSCR ;DSABLE DV11
5430 027114 104400 SCOPE ;SCOPE THIS TEST.
5431 027116 012706 001200 2$: MOV #STACK,SP ;RESET STACK
5432 027122 005077 152234 CLR @DVSCR ;DSABLE DV11
5433 027126 000763 BR 1$ ;RETURN
5434 027130 104011 3$: HLT 11 ;VECTOR HERE WAS WRONG SIDE
5435 027132 000771 BR 2$
```

```
:***** TEST 113 *****
:*TEST THAT SETTING BIT15!BIT9 AND BIT13!BIT9
:*NPR STAT INTR AND NPR STAT IE PRODUCE AN INTERRUPT ON VECTOR 'B'
:*****
```

: TEST 113

```
5444 027134 012737 000113 001226 TST113: MOV #113,TSTNO
5445 027142 012737 027312 001216 MOV #TST114,NEXT
5446 027150 012737 000340 177776 MOV #340,PS ;LOCK OUT CPU INTERRUPTS
5447 027156 104412 MSTCLR ;ISSUE DVRESET
5448 027160 004537 032210 JSR R5,SETVEC ;GO SET VECTOR 'A' AND 'B'
5449 027164 032232 NO.ATRAP ;'A'
5450 027166 032236 NO.BTRAP ;'B'
5451 027170 340 340 .BYTE 340,340 ;PRIO. AT 7
5452 027172 005037 177776 CLR PS ;ZERO CPU PRIO.
5453 027176 012777 101000 152156 MOV #BIT15!BIT9,@DVSCR ;SET AN INTERRUPT RELATIVE BIT.
5454 027204 000240 NOP ;WAIST TIME
5455 027206 012777 021000 152146 MOV #BIT13!BIT9,@DVSCR ;SET THE ALTERNATE RELATIVE BIT.
5456 027214 000240 NOP ;WAIST
5457 027216 005077 152140 CLR @DVSCR ;ZERO REG
5458 027222 012777 001000 152132 MOV #BIT9,@DVSCR ;SET SYS MAINT ENABLE
5459 027230 004537 032210 JSR R5,SETVEC ;GO RESET VECTORS 'A' AND 'B'
5460 027234 027306 3$ ;'A'
5461 027236 027274 2$ ;'B'
5462 027240 340 340 .BYTE 340,340 ;PRIO. AT 7
5463 027242 052777 121000 152112 BIS #BIT15!BIT9.BIT13!BIT9,@DVSCR
5464 027250 000240 NOP ;SET BOTH INTERRUPT RELATIVE BITS.
5465 027252 104010 HLT 10 ;SETTING OF THESE BITS FAILED TO PRODUCE AN INTFRUPT
5466 027254 004537 032210 1$: JSR R5,SETVEC ;RESET VECTORS
5467 027260 032232 NO.ATRAP ;'A'
```



```
5468 027262 032236 NO.BTRAP ;'B'  
5469 027264 340 340 .BYTE 340,340  
5470 027266 005077 152070 CLR @DVSCR ;DSABLE DV11  
5471 027272 104400 SCOPE ;SCOPE THIS TEST.  
5472 027274 012706 001200 2$: MOV #STACK,SP ;RESET STACK  
5473 027300 005077 152056 CLR @DVSCR ;DSABLE DV11  
5474 027304 000763 BR 1$ ;RETURN  
5475 027306 104011 3$: HLT 11 ;VECTOR HERE WAS WRONG SIDE  
5476 027310 000771 BR 2$
```

```
5477  
5478 :***** TEST 114 *****  
5479 :*TEST TO VERIFY THAT VECTOR 'A'  
5480 :*OCCURES BEFOR VECTOR 'B' EVEN  
5481 :*WHEN VECTOR 'B' IS ENABLED BEFORE  
5482 :*VECTOR 'A'.  
5483 :*****  
5484
```

```
5485 : TEST 114  
5486 :-----  
5487 027312 012737 000114 001226 TST114: MOV #114,TSTNO  
5488 027320 012737 027452 001216 MOV #TST115,NEXT  
5489 027326 012737 000340 177776 MOV #340,PS ;SET CPU TO PRIO 7  
5490 027334 104412 MSTCLR ;RESET DV11  
5491 027336 005003 CLR R3 ;SET COUNTER TO 0  
5492 027340 012700 000002 MOV #2,R0 ;SET TO GET 2 INTERRUPTS  
5493 027344 004537 032210 JSR R5,SETVEC ;SET DV11 VECTORS  
5494 027350 027420 2$ ;RECEIVER INTERRUPTS TO 2$  
5495 027352 027442 3$ ;TRANSMITTER INTERRUPTS TO 3$  
5496 027354 340 340 .BYTE 340,340 ;SET PRIORITY TO 7 ON INTERRUPT  
5497 027356 012777 001000 151776 MOV #BIT9,@DVSCR ;SET SYSTEM MAINT  
5498 027364 052777 120000 151770 BIS #BIT15+BIT13,@DVSCR  
5499 027372 052777 001300 151762 BIS #BIT9+BIT7+BIT6,@DVSCR  
5500 027400 005037 177776 CLR PS ;SET B INTERRUPT ;A INTERRUPT ;CLEAR CPU STATUS  
5501 027404 105203 INCB R3 ;HANG WAITING FOR INTERRUPTS  
5502 027406 001376 BNE -2 ;  
5503 027410 005700 TST R0 ;DID TWO INTERRUPTS OCCUR?  
5504 027412 001401 BEQ 1$ ;BR IF YES  
5505 027414 104000 HLT 0 ;EITHER NOT ENOUGH(2) OR TOO MANY (72) INTERRUPTS  
5506 027416 104400 1$: SCOPE ;SCOPE TEST  
5507 027420 005300 2$: DEC R0 ;RXISR  
5508 027422 022700 000001 CMP #1,R0 ;IF RX GOT HERE 1ST R0-1 S/B 1  
5509 027426 001401 BEQ 64$ ;BR IF OK  
5510 027430 104011 HLT 11 ;RX NOT 1ST  
5511 027432 042777 000300 151722 64$: BIC #BIT7+BIT6,@DVSCR ;DISQUALIFY INTERRUPT REQST  
5512 027440 000002 RTI ;RETURN  
5513 027442 005300 3$: DEC R0 ;TX ISR  
5514 027444 001401 BEQ 4$ ;IF SCOND INTERRUPT R0 SHOULD NOW 0  
5515 027446 104012 HLT 12 ;TX INTERRUPT OUT OF ORDER  
5516 027450 000002 4$: RTI ;RETURN
```

```
5517  
5518 :***** TEST 115 *****  
5519 :*PRIORITY INTERUPT TESTS.  
5520 :*SET PS TO PRIORITY 7 AND VERIFY  
5521 :*THAT THE DV11 DOESN'T INTERUPT.  
5522 :*****  
5523
```

```
5524  
5525 : TEST 115  
5526 :-----  
5527 027452 012737 000115 001226 TST115: MOV #115,TSTNO  
5528 027460 012737 027542 001216 MOV #TST116,NEXT  
5529 027466 012737 000340 177776 MOV #340,PS ;LOCK OUT INTERUPTS  
5530 027474 104412 MSTCLR ;CLEAN DV11  
5531 027476 004537 032210 JSR R5,SETVEC ;PREPARE VECTORS  
5532 027502 032232 NO.ATRAP ;'A'  
5533 027504 032236 NO.BTRAP ;'B'  
5534 027506 340 340 .BYTE 340,340 ;PRIO. AT 7  
5535 027510 012777 001300 151644 MOV #BIT9+BIT7+BIT6,@DVSCR  
5536 027516 012737 000340 177776 MOV #340,PS ;SET CPU PRIO AND ENABLE VECT. 'A'  
5537 027524 000240 NOP ;WAIST  
5538 027526 005077 151630 1$: CLR @DVSCR ;DSABLE DV11  
5539 027532 104400 SCOPE ;SCOPE THIS TEST  
5540 027534 012716 027526 2$: MOV #1$, (SP) ;SET FOR RETURN  
5541 027540 000002 RTI ;  
5542  
5543
```

```
:***** TEST 116 *****  
:*PRIORITY INTERRUPT TESTS.  
:*SET PS TO PRIORITY 6 AND VERIFY  
:*THAT THE DV11 DOESN'T INTERRUPT.  
:*****
```

```
5549  
5550 : TEST 116  
5551 :-----  
5552 027542 012737 000116 001226 TST116: MOV #116,TSTNO  
5553 027550 012737 027632 001216 MOV #TST117,NEXT  
5554 027556 012737 000340 177776 MOV #340,PS ;LOCK OUT INTERUPTS  
5555 027564 104412 MSTCLR ;CLEAN DV11  
5556 027566 004537 032210 JSR R5,SETVEC ;PREPARE VECTORS  
5557 027572 032232 NO.ATRAP ;'A'  
5558 027574 032236 NO.BTRAP ;'B'  
5559 027576 340 340 .BYTE 340,340 ;PRIO. AT 7  
5560 027600 012777 001300 151554 MOV #BIT9+BIT7+BIT6,@DVSCR  
5561 027606 012737 000300 177776 MOV #300,PS ;SET CPU PRIO AND ENABLE VECT. 'A'  
5562 027614 000240 NOP ;WAIST  
5563 027616 005077 151540 1$: CLR @DVSCR ;DSABLE DV11  
5564 027622 104400 SCOPE ;SCOPE THIS TEST  
5565 027624 012716 027616 2$: MOV #1$, (SP) ;SET FOR RETURN  
5566 027630 000002 RTI ;  
5567  
5568
```

```
:***** TEST 117 *****  
:*PRIORITY INTERRUPT TESTS.  
:*SET PS TO PRIORITY 5 AND VERIFY  
:*THAT THE DV11 DOESN'T INTERRUPT.  
:*****
```

```
5569  
5570 : TEST 117  
5571 :-----  
5572 027632 012737 000117 001226 TST117: MOV #117,TSTNO  
5573 027640 012737 027722 001216 MOV #TST120,NEXT  
5574 027646 012737 000340 177776 MOV #340,PS ;LOCK OUT INTERUPTS  
5575  
5576
```

```

5580 027654 104412          MSTCLR          ;CLEAN DV11
5581 027656 004537 032210 JSR      R5,SETVEC ;PREPARE VECTORS
5582 027662 032232          NO.ATRAPH      ;'A'
5583 027664 032236          NO.BTRAPH      ;'B'
5584 027666      340      340 .BYTE 340,340 ;PRIO. AT 7
5585 027670 012777 001300 151464 MOV      #BIT9+BIT7+BIT6,@DVSCR
5586 027676 012737 000240 177776 MOV      #240,PS ;SET CPU PRIO AND ENABLE VECT. 'A'
5587 027704 000240          NOP              ;WAIST
5588 027706 005077 151450 1$: CLR      @DVSCR ;DSABLE DV11
5589 027712 104400          SCOPE          ;SCOPE THIS TEST
5590 027714 012716 027706 2$: MOV      #1$, (SP) ;SET FOR RETURN
5591 027720 000002          RTI              ;
5592
5593
5594
5595
5596
5597
5598
5599

```

```

:***** TEST 120 *****
:*PRIORITY INTERRUPT TESTS.
:*SET PS TO PRIORITY 4 AND VERIFY
:*THAT THE DV11 DOES INTERRUPT.
:*****

```

: TEST 120

```

5600
5601
5602 027722 012737 000120 001226 TST120: MOV      #120,TSTNO
5603 027730 012737 030014 001216 MOV      #TST121,NEXT
5604 027736 012737 000340 177776 MOV      #340,PS ;LOCK OUT INTERRUPTS
5605 027744 104412          MSTCLR          ;CLEAN DV11
5606 027746 004537 032210 JSR      R5,SETVEC ;PREPARE VECTORS
5607 027752 030006          2$              ;'A'
5608 027754 032236          NO.BTRAPH      ;'B'
5609 027756      340      340 .BYTE 340,340 ;PRIO. AT 7
5610 027760 012777 001300 151374 MOV      #BIT9+BIT7+BIT6,@DVSCR
5611 027766 012737 000200 177776 MOV      #200,PS ;SET CPU PRIO AND ENABLE VECT. 'A'
5612 027774 000240          NOP              ;WAIST
5613 027776 104007          HLT      7 ;DV FAILED TO INTERRUPT
5614 030000 005077 151356 1$: CLR      @DVSCR ;DSABLE DV11
5615 030004 104400          SCOPE          ;SCOPE THIS TEST
5616 030006 012716 030000 2$: MOV      #1$, (SP) ;SET FOR RETURN
5617 030012 000002          RTI              ;
5618
5619
5620
5621
5622
5623
5624
5625
5626
5627

```

```

:***** TEST 121 *****
:*TEST THAT BIT15(NPR STATUS INTR) WILL
:*SET WHEN AN ENTRY IS MADE INTO THE
:*NPR STATUS REGISTER.
:*ALSO VERIFY THAT READING THE DVNSR CLEARS DVSCR BIT15.
:*****

```

: TEST 121

```

5628
5629 030014 012737 000121 001226 TST121: MOV      #121,TSTNO
5630 030022 012737 030116 001216 MOV      #TST122,NEXT
5631 030030 104412          MSTCLR          ;RESET DV11
5632 030032 005777 151324 TST      @DVSCR ;MAKE SURE 15 0
5633 030036 100001          BPL      64$ ;BR IF OK
5634 030040 104000          HLT      0 ;15 S/B-0
5635 030042 012777 000010 151312 64$: MOV      #BIT3,@DVSCR ;SET SOURCE SEC

```

```

5636 030050 012777 030367 151322      MOV    #XFR+BIT7+BIT6+BIT5+BIT4+BIT2+BIT1+BIT0,@DVSFR
5637 030056 005000                      CLR    R0                          :XFR ALU RESULT TO DVNSR
5638 030060 104415                      ROMCLK                      :EXECUTE
5639 030062 005777 151274      1$:   TST    @DVSCR                      :15=1?
5640 030066 100404                      BMI    2$                      :BR IF YES
5641 030070 104414                      DELAY                      :WASTE TIME
5642 030072 005200                      INC    R0                          :DO DELAY AND WAIT
5643 030074 001372                      BNE    1$                      :
5644 030076 104000                      HLT    0                          :15 NEVER SET
5645 030100 005777 151276      2$:   TST    @DVNSR                      :READ NSR
5646 030104 005777 151252      TST    @DVSCR                      :DID 15 CLEAR IN SCR?
5647 030110 100001                      BPL    3$                      :BR IF YES
5648 030112 104000                      HLT    0                          :DVSCR IS NOT CLEARED BY READING NSR
5649 030114 104400      3$:   SCOPE                          :SCOPE TEST

```

```

5650
5651
5652      :***** TEST 122 *****
5653      :*TEST TO WRITE PATTERNS THROUGH
5654      :*THE NPR STATUS REGISTER.
5655      :*BITS WRITTEN: 11,10,09,08,03,02,01,00
5656      :*(WHEN BIT 15 OF DVSCR IS SET SO SHOULD BIT 15 OF DVNSR)
5657      :*****

```

: TEST 122

```

5658
5659      :-----
5660      :TST122:
5661 030116 012737 000122 001226      MOV    #122,TSTNO
5662 030124 012737 030316 001216      MOV    #TST123,NEXT
5663 030132 012737 030174 001220      MOV    #1$,LOCK
5664 030140 104412                      MSTCLR                      :RESET DV11
5665 030142 012777 000010 151212      MOV    #BIT3,@DVSCR          :SET SOURCE SEL
5666 030150 005000                      CLR    R0                          :ZERO LINE NUMBER POINTER
5667 030152 005037 001250                      CLR    TEMP2                      :ZERO DATA
5668 030156 013703 001402                      MOV    DVNSR,R3                  :SET POINTER
5669 030162 013702 001400                      MOV    DVSFR,R2
5670 030166 012737 100000 001246      MOV    #BIT15,TEMP1           :SET CORRESPONDING BIT TO BE FOUND IN NSR.
5671 030174 012712 030267      1$:   MOV    #XFR+BIT7+BIT5+BIT4+BIT2+BIT1+BIT0,(R2)
5672 030200 104415                      ROMCLK                      :XFR RAM OUTPUT TO NSR
5673 030202 005777 151154                      TST    @DVSCR                      :WAIT FOR
5674 030206 100375                      BPL    -4                          :
5675 030210 013705 001246                      MOV    TEMP1,R5                  :GET GOOD 15=1 DATA
5676 030214 011304                      MOV    (R3),R4                  :READ NSR
5677 030216 020504                      CMP    R5,R4                      :OK?
5678 030220 001401                      BEQ    2$                          :BR IF GOOD
5679 030222 104013                      HLT    13                          :DVNSR HAS WRONG DATA
5680 030224 104401      2$:   SCOPE                          :LOCK ON DATA?
5681 030226 004537 031676      PERFORM ,SETSCAN              :UPDATE MSCANNER
5682 030232 000001                      1                                :BY ONE
5683 030234 005237 001246                      INC    TEMP1                      :UPDATE DATA
5684 030240 032700 000020                      BIT    #BIT4,R0                  :ALL DONE?
5685 030244 001753                      BEQ    1$                          :BR IF NO.
5686 030246 042737 077777 001246      BIC    #^C<BIT15>,TEMP1       :CLEAR ALL BUT ENTRY PRESENT
5687 030254 005000                      CLR    R0                          :ZERO LINE POINTER
5688 030256 005001                      CLR    R1                          :ZERO MSCANNER, POINTER
5689 030260 005237 001250                      INC    TEMP2                      :UPDATE HIGH BYTE DATA
5690 030264 153737 001250 001247      BJSB   TEMP2,TEMP1+1          :PLACE IN GOOD LOCATION
5691 030272 013712 001250                      MOV    TEMP2,(R2)                :RAM ADDRESS

```

5692 030276 052712 020000  
 5693 030302 104415  
 5694 030304 032737 000020 001250  
 5695 030312 001730  
 5696 030314 104400

BIS #RAM,(R2) ;RAM READ  
 ROMCLK ;  
 BIT #BIT4,TEMP2 ;ALL DONE?  
 BEQ 1\$ ;BR IF NO  
 SCOPE ;SCOPE TEST.

\*\*\*\*\* FIRST PLANNED ATTEMPT \*\*\*\*\*  
 \*\*\*\*\* TO EXECUTE NPR. \*\*\*\*\*

\*\*\*\*\* TEST 123 \*\*\*\*\*  
 ;\*BASIC TEST OF THE NPR OPERATION INSTRUCTION.  
 ;\*TEST THAT THE DV11 CAN 'READ' FROM CORE LOCATION  
 ;\*VIA THE NPR LOGIC.  
 ;\*LOCATION 'NPRLOC' WILL HAVE A BINARY COUNT PATTERN  
 ;\*PLACED INTO IT AND READ INTO THE DV11 AND XFERED  
 ;\*INTO THE DVRIC REGISTER.  
 ;\*NOTE: THIS TEST USES AN EVEN ADDRESS FOR THE NPR OPERATION  
 ;\*\*\*\*\*

; TEST 123

5714 030316 012737 000123 001226  
 5715 030324 012737 030456 001216  
 5716 030332 012737 030414 001220  
 5717 030340 104412  
 5718 030342 012777 000010 151012  
 5719 030350 013703 001366  
 5720 030354 013702 001400  
 5721 030360 012777 033352 151010  
 5722 030366 012712 020000  
 5723 030372 104415  
 5724 030374 012712 030263  
 5725 030400 104415  
 5726 030402 005037 033352  
 5727 030406 005005  
 5728 030410 005077 150762  
 5729 030414 012712 040000  
 5730 030420 104415  
 5731 030422 012712 030226  
 5732 030426 104415  
 5733 030430 011304  
 5734 030432 013705 033352  
 5735 030436 020504  
 5736 030440 001401  
 5737 030442 104013  
 5738 030444 104.01  
 5739 030446 105237 033352  
 5740 030452 001360  
 5741 030454 104400

```

TST123: MOV #123,TSTNO
        MOV #TST124,NEXT
        MOV #1$,LOCK
        MSTCLR ;RESET DV11
        MOV #BIT3,@DVSCR ;SET SOURCE SEL.
        MOV DVRIC,R3 ;SET POINTERS
        MOV DVSFR,R2 ;
        MOV #NPRLOC,@DVSRA
        MOV #RAM,(R2) ;RAM READ
        ROMCLK ;EXECUTE
        MOV #XFR+BIT7+BIT5+BIT4+BIT1+BIT0,(R2)
        ROMCLK ;DO XFR TO NPR ADD.
        CLR NPRLOC ;CLEAR NPR LOCATION
        CLR R5 ;CLEAR GOOD
        CLR @DVSRA ;CLEAR DATA PORT
1$: MOV #NPR,(R2) ;DO THE NPR
        ROMCLK ;***NOW***
        MOV #XFR+BIT7+BIT4+BIT2+BIT1,(R2)
        ROMCLK ;XFR NPR OUTPUT TO DVRIC
        MOV (R3),R4 ;READ RIC
        MOV NPRLOC,R5 ;READ GOOD DATA
        CMP R5,R4 ;OK?
        BEQ 2$ ;BR IF YES
        HLT 13 ;NPR FUNCTION FAILED
2$: SCOP1 ;LOOP?
        INCB NPRLOC ;UPDATE DATA
        BNE 1$ ;BR IF MORE
        SCOPE ;SCOPE TEST
    
```

\*\*\*\*\* TEST 124 \*\*\*\*\*  
 ;\*BASIC TEST OF THE NPR OPERATION INSTRUCTION.  
 ;\*TEST THAT THE DV11 CAN 'READ' FROM CORE LOCATION

5742  
 5743  
 5744  
 5745  
 5746  
 5747

5748 : \*VIA THE NPR LOGIC.  
5749 : \*LOCATION 'NPRLOC' WILL HAVE A BINARY COUNT PATTERN  
5750 : \*PLACED INTO IT AND READ INTO THE DV11 AND XFERED  
5751 : \*INTO THE DVRIC REGISTER.  
5752 : \*NOTE: THIS TEST USES AN ODD ADDRESS FOR THE NPR OPERATION.  
5753 : \*\*\*\*\*  
5754

: TEST 124

5755 :-----  
5757 030456 012737 000124 001226 TST124: MOV #124,TSTNO  
5758 030464 012737 030620 001216 MOV #TST125,NEXT  
5759 030472 012737 030554 001220 MOV #1\$,LOCK  
5760 030500 104412 MSTCLR ;RESET DV11  
5761 030502 012777 000010 150652 MOV #BIT3,@DVSCR ;SET SOURCE SEL.  
5762 030510 013703 001366 MOV DVRIC,R3 ;SET POINTERS  
5763 030514 013702 001400 MOV DVSFR,R2 ;  
5764 030520 012777 033353 150650 MOV #NPRLOC+1,@DVSRA ;  
5765 030526 012712 020000 MOV #RAM,(R2) ;RAM READ  
5766 030532 104415 ROMCLK ;EXECUTE  
5767 030534 012712 030263 MOV #XFR+BIT7+BIT5+BIT4+BIT1+BIT0,(R2) ;DO XFR TO NPR ADD.  
5768 030540 104415 ROMCLK ;CLEAR NPR LOCATION  
5769 030542 005037 033352 CLR NPRLOC ;CLEAR GOOD  
5770 030546 005005 CLR R5 ;CLEAR DATA PORT  
5771 030550 005077 150622 CLR @DVSRA ;DO THE NPR  
5772 030554 012712 040000 1\$: MOV #NPR,(R2) ;\*\*\*NOW\*\*\*  
5773 030560 104415 ROMCLK ;XFR NPR OUTPUT TO DVRIC  
5774 030562 012712 030226 MOV #XFR+BIT7+BIT4+BIT2+BIT1,(R2) ;READ RIC  
5775 030566 104415 ROMCLK ;GET GOOD DATA  
5776 030570 011304 MOV (R3),R4 ;OK?  
5777 030572 013705 033352 MOV NPRLOC,R5 ;BR IF YES  
5778 030576 000305 SWAB R5 ;NPR FUNCTION FAILED  
5779 030600 020504 CMP R5,R4 ;LOOP?  
5780 030602 001401 BEQ 2\$ ;  
5781 030604 104013 HLT 13 ;  
5782 030606 104401 2\$: SCOPI ;UPDATE DATA  
5783 030610 105237 033353 INCB NPRLOC+1 ;BR IF MORE  
5784 030614 001357 BNE 1\$ ;SCOPE TEST  
5785 030616 104400 SCOPE ;  
5786  
5787  
5788

5789 : \*\*\*\*\* TEST 125 \*\*\*\*\*  
5790 : \*BASIC TEST OF THE NPR OPERATION INSTRUCTION.  
5791 : \*TEST THAT THE DV11 CAN 'WRITTEN' INTO CORE LOCATION  
5792 : \*VIA THE NPR LOGIC.  
5793 : \*LOCATION 'NPRLOC' WILL HAVE A BINARY COUNT PATTERN  
5794 : \*WRITTEN INTO IT BY THE DV11 NPR LOGIC.  
5795 : \*NOTE: THIS TEST USES AN EVEN ADDRESS FOR THE NPR OPERATION  
5796 : \*\*\*\*\*  
5797

: TEST 125

5798 :-----  
5799 :  
5800 030620 012737 000125 001226 TST125: MOV #125,TSTNO  
5801 030626 012737 031002 001216 MOV #TST126,NEXT  
5802 030634 012737 030716 001220 MOV #1\$,LOCK  
5803 030642 104412 MSTCLR ;RESET DV11

```
5804 030644 012777 000010 150510 MOV #BIT3,@DVSCR ;SET SOURCE SEL.
5805 030652 013703 001366 MOV DVRIC,R3 ;SET POINTERS
5806 030656 013702 001400 MOV DVSFR,R2 ;
5807 030662 012777 033352 150506 MOV #NPRLOC,@DVSRA ;
5808 030670 012712 020000 MOV #RAM,(R2) ;RAM READ
5809 030674 104415 ROMCLK ;EXECUTE
5810 030676 012712 031663 MOV #XFR+BIT9+BIT8+BIT7+BIT5+BIT4+BIT1+BIT0,(R2)
5811 030702 104415 ROMCLK ;DO XFR TO NPR ADD.
5812 030704 005037 033352 CLR NPRLOC ;CLEAR NPR LOCATION
5813 030710 005005 CLR R5 ;CLEAR GOOD
5814 030712 005077 150460 CLR @DVSRA ;CLEAR DATA PORT
5815 030716 005037 033352 1$: CLR NPRLOC ;CLEAR NPR LOC
5816 030722 012712 020000 MOV #RAM,(R2) ;DO RAM READ
5817 030726 104415 ROMCLK ;
5818 030730 012712 030265 MOV #XFR+BIT7+BIT5+BIT4+BIT2+BIT0,(R2)
5819 030734 104415 ROMCLK ;XFR RAM OUTPUT TO NPR INPUT DATA
5820 030736 012712 040000 MOV #NPR,(R2) ;DO THE NPR
5821 030742 104415 ROMCLK ;***NOW***
5822 030744 017705 150426 MOV @DVSRA,R5 ;READ GOOD DATA
5823 030750 013704 033352 MOV NPRLOC,R4 ;READ ??? DATA
5824 030754 020504 CMP R5,R4 ;OK?
5825 030756 001401 BEQ 2$ ;BR IF YES
5826 030760 104013 HLT 13 ;NPR FUNCTION FAILED
5827 030762 104401 2$: SCOP* ;LOOP?
5828 030764 005277 150406 INC @DVSRA ;UPDATE DATA
5829 030770 032777 000400 150400 BIT #BIT8,@DVSRA ;ALL DONE?
5830 030776 001747 BEQ 1$ ;BR IF NO
5831 031000 104400 SCOPE ;SCOPE TEST
```

```
5832
5833
5834
5835 ;***** TEST 126 *****
5836 ;*BASIC TEST OF THE NPR OPERATION INSTRUCTION.
5837 ;*TEST THAT THE DV11 CAN 'WRITTEN' INTO CORE LOCATION
5838 ;*VIA THE NPR LOGIC.
5839 ;*LOCATION 'NPRLOC' WILL HAVE A BINARY COUNT PATTERN
5840 ;*WRITTEN INTO IT BY THE DV11 NPR LOGIC.
5841 ;*NOTE: THIS TEST USES AN ODD ADDRESS FOR THE NPR OPERATION.
5842 ;*****
5843
```

```
5844 ; TEST 126
5845 ;-----
5846 031002 012737 000126 001226 TST126: MOV #126,TSTNO
5847 031010 012737 031166 001216 MOV #TST127,NEXT
5848 031016 012737 031100 001220 MOV #1$,LOCK
5849 031024 104412 MSTCLR ;RESET DV11
5850 031026 012777 000010 150326 MOV #BIT3,@DVSCR ;SET SOURCE SEL.
5851 031034 013703 001366 MOV DVRIC,R3 ;SET POINTERS
5852 031040 013702 001400 MOV DVSFR,R2 ;
5853 031044 012777 033353 150324 MOV #NPRLOC+1,@DVSRA ;
5854 031052 012712 020000 MOV #RAM,(R2) ;RAM READ
5855 031056 104415 ROMCLK ;EXECUTE
5856 031060 012712 031663 MOV #XFR+BIT9+BIT8+BIT7+BIT5+BIT4+BIT1+BIT0,(R2)
5857 031064 104415 ROMCLK ;DO XFR TO NPR ADD.
5858 031066 005037 033352 CLR NPRLOC ;CLEAR NPR LOCATION
5859 031072 005005 CLR R5 ;CLEAR GOOD
```

5860	031074	005077	150276		CLR	@DVSRA	:CLEAR DATA POR'
5861	031100	005037	033352	18:	CLR	NPRLOC	:CLEAR NPR LOC
5862	031104	012712	020000		MOV	#RAM,(R2)	:DO RAM READ
5863	031110	104415			ROMCLK		
5864	031112	012712	030265		MOV	#XFR+BIT7+BIT5+BIT4+BIT2+BIT0,(R2)	
5865	031116	104415			ROMCLK		:XFR RAM OUTPUT TO NPR INPUT DATA
5866	031120	012712	040000		MOV	#NPR,(R2)	:DO THE NPR
5867	031124	104415			ROMCLK		:***NOW***
5868	031126	017705	150244		MOV	@DVSRA,R5	:READ GOOD DATA
5869	031132	013704	033352		MOV	NPRLOC,R4	:GET DATA
5870	031136	000304			SWAB	R4	
5871	031140	020504			CMP	R5,R4	:OK?
5872	031142	001401			BEQ	28	:BR IF YES
5873	031144	104013			HLT	13	:NPR FUNCTION FAILED
5874	031146	104401		28:	SCOPE		:LOOP?
5875	031150	005277	150222		INC	@DVSRA	:UPDATE DATA
5876	031154	032777	000400	150214	BIT	#BIT8,@DVSRA	:ALL DONE?
5877	031162	001746			BEQ	18	:BR IF NO
5878	031164	104400			SCOPE		:SCOPE TEST

5880  
5881  
5882 :\*\*\*\*\* TEST 127 \*\*\*\*\*  
5883 :\*BASIC TEST OF THE NPR OPERATION INSTRUCTION.  
5884 :\*TEST THAT THE DV11 CAN DO AN NPR  
5885 :\*TO A NON-EXISTANT MEMORY.  
5886 :\*TEST THAT BRANCH 'A' -NXM H- IS SET AFTER  
5887 :\*THE NPR. THEN DO A 'SET/CLEAR' CLEAR NXM  
5888 :\*AND VERIFY THAT IT IS CLEARED.  
5889 :\*\*\*\*\*

5890  
5891 : TEST 127  
5892 :-----  
5893 031166 012737 000127 001226 TST127: MOV #127,TSTNO  
5894 031174 012737 031344 001216 MOV #TST130,NEXT  
5895 031202 104412 MSTCLR :RESET DV11  
5896 031204 012777 000010 150150 MOV #BIT3,@DVSCR :SET SOURCE SEL.  
5897 031212 013703 001366 MOV DVRIC,R3 :SET POINTERS  
5898 031216 013702 001400 MOV DVSFR,R2  
5899 031222 052777 000060 150132 BIS #BIT5+BIT4,@DVSCR :SET EA BITS.  
5900 031230 012777 177320 150140 MOV #177320,@DVSRA :LOAD 'NXM'.  
5901 031236 012712 020000 MOV #RAM,(R2) :RAM READ  
5902 031242 104415 ROMCLK :EXECUTE  
5903 031244 012712 031663 MOV #XFR+BIT9+BIT8+BIT7+BIT5+BIT4+BIT1+BIT0,(R2)  
5904 031250 104415 ROMCLK :EXECUTE  
5905 031252 012712 040000 MOV #NPR,(R2) :DO THE NPR  
5906 031256 104415 ROMCLK :\*\*\*NOW\*\*\*  
5907 031260 012712 003000 MOV #BIT10+BIT9,(R2) :BRANCH 'A' NXM  
5908 031264 017704 150100 MOV @DVLCR,R4 :READ BRANCH TEST POINTS.  
5909 031270 010405 MOV R4,R5 :GET IMAGE  
5910 031272 042705 000001 BIC #BIT0,R5 :BR 'A' TRUE  
5911 031276 052705 000002 BIS #BIT1,R5 :BR B FALSE  
5912 031302 020504 CMP R5,R4 :NXM TRUE?  
5913 031304 001401 BEQ .+4 :BR IF YES  
5914 031306 104006 HLT 6 :BR 'A' OF NXM FAILED.  
5915 031310 012712 050017 MOV #S.C+BIT3+BIT2+BIT1+BIT0,(R2)



```

5916                                     ;SET/CLEAR CLEAR NXM.
5917 031314 104415 ROMCLK                ;EXECUTE.
5918 031316 012712 003000 MOV #BIT10+BIT9,(R2) ;BRANCH 'A' NXM
5919 031322 017704 150042 MOV @DVLCR,R4 ;READ BRANCH TEST POINTS.
5920 031326 010405 MOV R4,R5 ;GET IMAGE
5921 031330 052705 000003 BIS #BIT1+BIT0,R5 ;BR A FALSE BR B FALSE
5922 031334 020504 CMP R5,R4 ;NXM TRUE?
5923 031336 001401 BEQ .+4 ;BR !F NO
5924 031340 104006 HLT 6 ;BR 'A' OF NXM FAILED TO CLEAR.
5925 031342 104400 SCOPE ;SCOPE TEST
5926
5927
5928

```

```

5929 ;***** TEST 130 *****
5930 ;*TEST MEMORY EXT. BITS 16 AND 17 UTILIZING
5931 ;*THE ALU SECTION OF THE DV11 MICROPROCESSOR.
5932 ;*ENABLES LINE 0, E.A. BIT 17 = 1, BIT 16 = 0
5933 ;*SEC REG 0 -> REG 'A' -> ALU -> SEC REG 2
5934 ;*SEC REG 0 = SEC REG 2?
5935 ;*SEC REG 0 -> REG 'B' -> ALU -> SEC REG 4
5936 ;*SEC REG 0 = SEC REG 4?
5937 ;*REPEAT ABOVE STEPS FOR E.A. BIT 16 = 1, BIT 17 = 0
5938 ;*****

```

```

5939 ; TEST 130
5940 -----
5941 031344 012737 000130 001226 TST130: MOV #130,TSTNO
5942 031352 012737 002436 001216 MOV #.EOP,NEXT
5943 031360 104412 MSTCLR ;RESET DV11,DISABLE MICROPROCESSOR
5944 031362 012701 014622 MOV #MEMEXT,R1 ;R1 IS SEC REG POINTER
5945 031366 013700 001400 MOV DVSFR,R0 ;SET SP. FUNCTIONS REG POINTER
5946 031372 012737 000002 001254 MOV #2,TEMP4 ;ENABLE LOOP COUNTER
5947 031400 012703 000003 MOV #3,R3 ;R3 IS SEC REG COUNTER
5948 031404 005077 147762 CLR @DVSRS ;ENABLE LINE 0
5949 031410 111177 147760 1$: MOVB (R1),@DVSRSRSH ;SELECT THE SEC REG
5950 031414 005077 147756 CLR @DVSRA ;CLEAR THE SEC REG
5951 031420 005201 INC R1 ;ADD 1 TO SEC REG POINTER
5952 031422 005303 DEC R3 ;ANOTHER SEC REG TO CLEAR?
5953 031424 001371 BNE 1$ ;YES
5954 031426 012777 000050 147726 MOV #BIT5+BIT3,@DVSCR
5955 ;ENABLE E.A. BIT 17 & ROM DATA SOURCE
5956 031434 012701 014622 MOV #MEMEXT,R1 ;RESTORE SEC REG POINTER
5957 031440 004537 031640 2$: JSR R5,SCREG0 ;HAVE SUBROUTINE SET UP SEC REG 0
5958 031444 012710 030261 MOV #XFR+BIT7+BIT5+BIT4+BIT0,(R0)
5959 ;SEC REG 0 -> REG 'A'
5960 031450 104415 ROMCLK ;EXECUTE
5961 031452 012710 010037 MOV #ALU+BIT4+BIT3+BIT2+BIT1+BIT0,(R0) ;F - A
5962 031456 104415 ROMCLK ;EXECUTE
5963 031460 116177 000001 147706 MOVB 1(R1),@DVSRSRSH ;SET FOR SEC REG 2
5964 031466 012710 020002 MOV #RAM+BIT1,(R0) ;CLEAR 'FOOTPRINT' REG. FOR SEC. REG.2
5965 031472 104415 ROMCLK ;EXECUTE
5966 031474 012710 020762 MOV #RAM+BIT8+BIT7+BIT6+BIT5+BIT4+BIT1,(R0)
5967 ;ALU RESULT 0-17 -> SEC REG 2
5968 031500 104415 ROMCLK ;EXECUTE
5969 031502 012710 020002 MOV #RAM+BIT1,(R0) ;READ SEC REG 2
5970 031506 104415 ROMCLK ;EXECUTE
5971 031510 017702 147654 MOV @DVLCR,R2 ;SAVE DVLINE PARM. REG

```

```

5972 031514 042702 177717      BIC    #^C<BIT5+BIT4>,R2      ;CLEAR ALL EXCEPT E.A. BITS
5973                                     ;SEC REG 0 = SEC REG 2?
5974 031520 120402      CMPB   R4,R2                  ;NO,ERROR
5975 031522 001041      BNE    3$                    ;HAVE SUBROUTINE SET UP SEC REG 0
5976 031524 004537 031640      JSR    R5,SCREG0             ;SEC REG 0 -> REG 'B'
5977 031530 012710 030262      MOV    #XFR+BIT7+BIT5+BIT4+BIT1,(R0)
5978                                     ;EXECUTE
5979 031534 104415      ROMCLK
5980 031536 012710 010005      MOV    #ALU+BIT2+BIT0,(R0)   ;F = B
5981 031542 104415      ROMCLK
5982 031544 116177 000002 147622  MOVB   2(R1),@DVSRSH         ;SET TO LOAD SEC REG 4
5983 031552 012710 020004      MOV    #RAM+BIT2,(R0)        ;CLEAR 'FOOTPRINT' REG. FOR SEC. REG.4
5984 031556 104415      ROMCLK
5985 031560 012710 020764      MOV    #RAM+BIT8+BIT7+BIT6+BIT5+BIT4+BIT2,(R0)
5986                                     ;ALU RESULT 0-17 -> SEC REG 4
5987 031564 104415      ROMCLK
5988 031566 012710 020004      MOV    #RAM+BIT2,(R0)        ;READ SEC REG 4
5989 031572 104415      ROMCLK
5990 031574 017702 147570      MOV    @DVLCR,R2             ;SAVE DVLINE PARM. REG
5991 031600 042702 177717      BIC    #^C<BIT5+BIT4>,R2     ;CLEAR ALL BUT E.A. BITS
5992 031604 120402      CMPB   R4,R2                  ;SEC REG 0 = SEC REG 4?
5993 031606 001007      BNE    3$                    ;NO,ERROR
5994 031610 005337 001254      DEC    TEMP4                  ;1ST TIME THROUGH?
5995 031614 001405      BEQ    4$                    ;NO, FINISHED OK
5996 031616 012777 000030 147536  MOV    #BIT4+BIT3,@DVSCR
5997                                     ;ENABLE E.A. BIT 16 & ROM DATA SOURCE
5998 031624 000705      BR     2$                    ;PROCESS E.A. BIT 16
5999 031626 104015      HLT    1$                    ;ERROR MESSAGE
6000 031630 012777 000001 147524 3$:  MOV    #1,@DVSCR             ;SET MICROPROCESSOR GO
6001 031636 104400      SCOPE
6002                                     ;SCOPE TEST
6003                                     ;SUBROUTINE TO SET UP SEC REG 0 FOR
6004                                     ;MEMORY EXTENSION BITS/ALU TEST
6005 031640 005077 147526      SCREG0: CLR    @DVSR           ;ENABLE LINE 0
6006 031644 111177 147524      MOVB   (R1),@DVSRSH         ;SELECT SEC REG 0
6007 031650 112777 000100 147520  MOVB   #100,@DVSRA          ;SET LOW BYTE TRACE DATA
6008 031656 012710 020000      MOV    #RAM,(R0)            ;READ SEC REG 0
6009 031662 104415      ROMCLK
6010 031664 117704 147500      MOVB   @DVLCR,R4            ;SAVE DVLINE PARM REG
6011 031670 042704 177717      BIC    #^C<BIT5+BIT4>,R4
6012                                     ;CLEAR ALL BUT E.A. BITS
6013 031674 000205      RTS    R5                    ;RETURN TO MAIN PROGRAM
6014
    
```

```

6015
6016 031676          SETSCAN:
6017 031676 010346    MOV     R3,-(SP)
6018 031700 052777 000010 147454    BIS     #BIT3,@DVSCR
6019 031706 012503    MOV     (R5)+,R3
6020 031710 001414    BEQ    2$
6021 031712 012777 050102 147460 1$:  MOV     #BIT14+BIT12+BIT6+BIT1,@DVSFR
6022 031720 104415    ROMCLK
6023 031722 005201    INC     R1
6024 031724 012777 050102 147446    MOV     #BIT14+BIT12+BIT6+BIT1,@DVSFR
6025 031732 104415    ROMCLK
6026 031734 005201    INC     R1
6027 031736 005303    DEC     R3
6028 031740 001364    BNE    1$
6029 031742 012603    2$:  MOV     (SP)+,R3
6030 031744 010100    MOV     R1,R0
6031 031746 000241    CLC
6032 031750 006000    ROR     R0
6033 031752 000205    EXIT
6034
6035                ;SUBROUTINE TO LOAD DATA INTO 'A' AND 'B' REG.
6036                ;THE FIRST ARGUMENT WILL LOAD THE 'A' REGISTER;
6037                ;THE SECOND ARGUMENT WILL LOAD THE 'B' REGISTER;
6038                ;AND THE THIRD ARGUMENT WILL SPECIFY THE POLYNOMIAL USED.
6039
6040 031754 012377 147416    L.DATA: MOV     (R3)+,@DVSRA    ;LOAD DATA TO BE PLACED INTO THE 'A' REG
6041 031760 012710 020000    MOV     #RAM,(R0)        ;DO A ROM READ INSTR.
6042 031764 104415          ROMCLK                ;EXECUTE
6043 031766 012710 030261    MOV     #XFR+BIT7+BIT5+BIT4+BIT0,(R0)
6044 031772 104415          ROMCLK                ;DATA XFR FROM RAM OUTPUT TO 'A' REGISTER
6045 031774 012377 147376    MOV     (R3)+,@DVSRA    ;LOAD DATA TO BE PLACED INTO THE 'B' REG.
6046 032000 012710 020000    MOV     #RAM,(R0)        ;DO A ROM READ
6047 032004 104415          ROMCLK                ;
6048 032006 012710 030262    MOV     #XFR+BIT7+BIT5+BIT4+BIT1,(R0)
6049 032012 104415          ROMCLK                ;DO A DATA XFER FROM RAM OUTPUT TO THE 'B' REG.
6050 032014 012377 147356    MOV     (R3)+,@DVSRA    ;PLACE DATA TO REMAIN IN THE RAMOUTPUT REG
6051 032020 012710 020000    MOV     #RAM,(R0)        ;DO A RAM READ TO SPECIFY POLYN.
6052 032024 104415          ROMCLK                ;READ
6053 032026 000203    RTS     R3              ;LEAVE HERE
6054
6055
6056 032030 010046    SIMBCC: MOV     R0,-(SP)
6057 032032 010146    MOV     R1,-(SP)
6058 032034 010246    MOV     R2,-(SP)
6059 032036 012537 001246    MOV     (R5)+,TEMP1
6060 032042 012537 001250    MOV     (R5)+,TEMP2
6061 032046 012537 001252    MOV     (R5)+,TEMP3
6062 032052 005037 032204    1$:  CLR     BCCFBK
6063 032056 013700 001252    MOV     TEMP3,R0
6064 032062 006037 001250    ROR     TEMP2
6065 032066 005500    ADC     R0
6066 032070 032700 000001    BIT     #BIT0,R0
6067 032074 001402    BEQ    2$
6068 032076 005137 032204    COM     BCCFBK
6069 032102 013700 032202    2$:  MOV     XPOLY,R0
6070 032106 005100    COM     R0
    
```

6071	032110	040037	032204		BIC	R0,BCCFBK
6072	032114	000241			CLC	
6073	032116	006037	001252		ROR	TEMP3
6074	032122	013700	032204		MOV	BCCFBK,R0
6075	032126	013701	001252		MOV	TEMP3,R1
6076	032132	010102			MOV	R1,R2
6077	032134	040100			BIC	R1,R0
6078	032136	043702	032204		BIC	BCCFBK,R2
6079	032142	050200			BIS	R2,R0
6080	032144	043737	032202	001252	BIC	XPOLY,TEMP3
6081	032152	050037	001252		BIS	R0,TEMP3
6082	032156	005337	001246		DEC	TEMP1
6083	032162	001333			BNE	1\$
6084	032164	013737	001252	032206	MOV	TEMP3,CALBCC
6085	032172	012602			MOV	(SP)+,R2
6086	032174	012601			MOV	(SP)+,R1
6087	032176	012600			MOV	(SP)+,R0
6088	032200	000205			RTS	R5
6089	032202	000000				
6090	032204	000000				
6091	032206	000000				
6092		000200				
6093		120001				
6094		102010				
6095						
6096						
6097	032210					
6098	032210	012577	147136			
6099	032214	012577	147136			
6100	032220	112577	147130			
6101	032224	112577	147130			
6102	032230	000205				
6103						
6104	032232					
6105	032232	104011				
6106	032234	000002				
6107						
6108	032236					
6109	032236	104012				
6110	032240	000002				
6111						
6112						

XPOLY: 0  
BCCFBK: 0  
CALBCC: 0  
LRC8=200  
CRC16=120001  
CRC.CCITT=102010

SETVEC:  
MOV (R5)+,@DVRVEC  
MOV (R5)+,@DVTVEC  
MOVB (R5)+,@DVRLVL  
MOVB (R5)+,@DVTLVL  
RTS R5

NO.ATRAP:  
HLT 11  
RTI

NO.BTRAP:  
HLT 12  
RTI

```

6113      032242 050377 044522 040515 .NLIST BEX
          032310 051777 041505 047117 EM1: .ASCIZ <377>/PRIMARY REGISTER ADDRESSING TIME-OUT/
          032354 050377 044522 040515 EM2: .ASCIZ <377>^SECONDARY REGISTER READ/WRITE TEST^
          032416 046777 046505 051117 EM3: .ASCIZ <377>^PRIMARY REGISTER READ/WRITE TEST^
          032460 051777 042520 044503 EM4: .ASCIZ <377>^MEMORY EXTENSION READ/WRITE TEST^
          032513      377 042526 052103 EM5: .ASCIZ <377>/SPECIAL FUNCTION REG TEST/
          032552 053377 041505 047524 EM6: .ASCIZ <377>/VECTOR 'A' FAILED TO INTERRUPT/
          032611      377 047125 054105 EM7: .ASCIZ <377>/VECTOR 'B' FAILED TO INTERRUPT/
          032654 052777 042516 050130 EM8: .ASCIZ <377>/UNEXPECTED INTERRUPT ON VECTOR 'A'/
          032717      377 051120 046511 EM9: .ASCIZ <377>/UNEXPECTED INTERRUPT ON VECTOR 'B'/
          032747      377 044514 042516 EM10: .ASCIZ <377>/PRIMARY REGISTER ERROR/
          032776 046777 046505 051117 EM11: .ASCIZ <377>/LINE CARD STATIC TEST/
          033036 051377 043505 051511 EM12: .ASCIZ <377>^MEMORY EXTENSION BITS/ALU TEST^
          033101      377 054105 042520 DH1: .ASCIZ <377>/REGISTER REFERENCED TRAPPED FROM/
          033153      377 054105 042520 DH2: .ASCIZ <377>/EXPECTED FOUND LINE SEC REG PRI REG/
          033206 045377 051123 050040 DH3: .ASCIZ <377>/EXPECTED FOUND PRI REG/
          033247      377 053104 043123 DH4: .ASCIZ <377>/JSR PC DVSFR EXPECTED FOUND/
          033300 046777 052123 041523 DH5: .ASCIZ <377>/DVSFR EXPECTED FOUND/
          033350      000000 DH6: .ASCIZ <377>/MSTSCAN DVSFR EXPECTED FOUND LINE/
          000000 .EVEN
          000000 SKIP=000000
6114 033350 000000 DATA: 0
6115 033352 000000 NPRLOC: 0
6116 033354 000002 DT1: 2
6117 033356 006 017 .BYTE 6,15.
6118 033360 001262 SAVR1
6119 033362 006 002 .BYTE 6,2
6120 033364 001264 SAVR2
6121 033366 000005 DT2: 5
6122 033370 006 004 .BYTE 6,4
6123 033372 001270 . 94
6124 033374 006 002 .BYTE 6,2
6125 033376 001266 SAVR3
6126 033400 002 004 .BYTE 2,4
6127 033402 001260 SAVR0
6128 033404 002 007 .BYTE 2,7
6129 033406 001262 SAVR1
6130 033410 006 002 .BYTE 6,2
6131 033412 001264 SAVR2
6132 033414 000003 DT3: 3
6133 033416 006 004 .BYTE 6,4
6134 033420 001272 SAVR5
6135 033422 006 002 .BYTE 6,2
6136 033424 001270 SAVR4
6137 033426 006 002 .BYTE 6,2
6138 033430 001266 SAVR3
6139 033432 000004 DT4: 4
6140 033434 006 002 .BYTE 6,2
6141 033436 001262 SAVR1
6142 033440 006 002 .BYTE 6,2
6143 033442 001266 SAVR3
6144 033444 006 004 .BYTE 6,4
6145 033446 001272 SAVR5
6146 033450 006 001 .BYTE 6,1
6147 033452 001270 SAVR4
6148 033454 DT5:
    
```

6149	033454	000003		3	
6150	033456	006	002	.BYTE	6.2
6151	033460	001264		SAVR2	
6152	033462	006	004	.BYTE	6.4
6153	033464	001272		SAVR5	
6154	033466	006	001	.BYTE	6.1
6155	033470	001270		SAVR4	
6156	033472	000005		DT6:	5
6157	033474	006	003	.BYTE	6.3
6158	033476	001260		SAVR0	
6159	033500	006	001	.BYTE	6.1
6160	033502	001264		SAVR2	
6161	033504	006	004	.BYTE	6.4
6162	033506	001272		SAVR5	
6163	033510	006	001	.BYTE	6.1
6164	033512	001270		SAVR4	
6165	033514	002	001	.BYTE	2.1
6166	033516	001262		SAVR1	
6167					
6168	033520			.ERRTAB:	
6169	033520	000000		0	
6170	033522	000000		0	
6171	033524	000000		0	
6172	033526	032242		EM1	
6173	033530	033036		DH1	;HALT 1
6174	033532	033354		DT1	
6175	033534	032310		EM2	
6176	033536	033101		DH2	;HALT 2
6177	033540	033366		DT2	
6178	033542	032354		EM3	
6179	033544	033153		DH3	;HALT 3
6180	033546	033414		DT3	
6181	033550	032416		EM4	
6182	033552	033101		DH2	;HALT 4
6183	033554	033366		DT2	
6184	033556	032460		EM5	
6185	033560	033206		DH4	;HALT 5
6186	033562	033432		DT4	
6187	033564	032460		EM5	
6188	033566	033247		DH5	;HALT 6
6189	033570	033454		DT5	
6190	033572	032513		EM6	
6191	033574	000000		0	;HALT 7
6192	033576	000000		0	
6193	033600	032552		EM7	
6194	033602	000000		0	;HALT 10
6195	033604	000000		0	
6196	033606	032611		EM8	
6197	033610	000000		0	;HALT 11
6198	033612	000000		0	
6199	033614	032654		EM9	
6200	033616	000000		0	;HALT 12
6201	033620	000000		0	
6202	033622	032717		EM10	
6203	033624	033153		DH3	;HALT 13
6204	033626	033414		DT3	

CZDVA-C MACY11 30A(1052) 03-OCT-79 14:07 PAGE 120  
CZDVAC.P11 03-OCT-79 14:04 DV11 DEVICE DIAGNOSTICS.

SEQ 0119

6205	033630	032747	EM11	
6206	033632	033300	DH6	;HALT 14
6207	033634	033472	DT6	
6208	033636	032776	EM12	
6209	033640	033101	DH2	;HALT 15
6210	033642	033366	DT2	
6211	033644			
6212	000001			

CORMAX:  
.END

CROSS REFERENCE TABLE -- USER SYMBOLS

ADRCNT= 003443	1157*	1193*	1202#											
ALU = 010000	611#	4106	4113	4148	4171	4173	4187	4197	4204	4224	4230	4252	4256	
	4258	4275	4294	4439	4467	4495	4523	4551	4579	4607	4635	4722	4846	
	4861	4876	4929	5961	5980									
ASYNCR = 004000	619#													
AUTO.S 006624	1667#													
BCC = 060000	616#	4838	4850	4865	4882	4916	4967	4989	5011	5033	5055	5077	5099	
	5121	5143	5165	5187	5209	5253	5298	5343						
BCCFBK 032204	6062*	6068*	6071*	6074	6078	6090#								
BINWRD 003746	1243*	1244	1281#											
BIT0 = 000001	609#	1424	2153	2258	2264	2286	2292	2314	2320	2342	2348	2370	2376	
	2398	2404	2423	2732	2952	2967	3140	3145	3164	3279	3344	3447	3456	
	3463	3465	3472	3474	3481	3483	3490	3492	3514	3541	3543	3550	3552	
	3559	3561	3568	3570	3608	3663	3687	3721	3723	3730	3732	3734	3736	
	3743	3745	3747	3749	3758	3762	3771	3775	3782	3784	3786	3788	3795	
	3797	3799	3801	3810	3814	3823	3827	3859	3860	3871	3879	3889	3895	
	3916	3923	3932	3945	3951	4020	4021	4042	4043	4106	4187	4195	4204	
	4224	4226	4258	4296	4437	4439	4443	4452	4453	4465	4467	4471	4480	
	4481	4493	4495	4499	4508	4509	4518	4521	4523	4527	4536	4537	4549	
	4551	4555	4564	4565	4577	4579	4583	4592	4593	4605	4607	4611	4620	
	4621	4633	4635	4639	4648	4649	4676	4677	4681	4687	4697	4700	4722	
	4726	4741	4743	4767	4769	4863	4878	4908	5636	5671	5724	5767	5810	
	5818	5856	5864	5903	5910	5915	5921	5958	5961	5980	6043	6066		
BIT1 = 000002	608#	1424	1435	2258	2264	2286	2292	2314	2320	2342	2348	2370	2376	
	2398	2404	2449	2758	3140	3164	3279	3344	3413	3445	3447	3454	3456	
	3463	3465	3472	3474	3481	3483	3490	3492	3514	3541	3543	3550	3552	
	3559	3561	3568	3570	3597	3608	3618	3719	3723	3730	3734	3736	3749	
	3756	3760	3762	3775	3782	3786	3788	3801	3808	3812	3814	3827	3859	
	3916	3931	3945	3998	4018	4020	4021	4029	4040	4042	4043	4051	4099	
	4106	4113	4147	4187	4188	4190	4196	4198	4200	4204	4205	4207	4224	
	4228	4229	4230	4231	4233	4254	4277	4296	4324	4333	4335	4383	4439	
	4443	4444	4452	4453	4467	4471	4472	4480	4481	4495	4499	4500	4508	
	4509	4523	4527	4528	4536	4537	4546	4551	4555	4556	4564	4565	4579	
	4583	4584	4592	4593	4607	4611	4612	4620	4621	4635	4639	4640	4648	
	4649	4677	4681	4687	4690	4700	4722	4743	4767	4769	4837	4848	4880	
	4910	4918	4929	4931	4953	5235	5280	5325	5636	5671	5724	5731	5767	
	5774	5810	5856	5903	5911	5915	5921	5961	5964	5966	5969	5977	6021	
	6024	6048												
BIT10 = 002000	599#	1424	2075	2283	2579	3140	3164	3452	3470	3479	3488	3511	3566	
	3617	3718	3722	3783	3787	3796	3800	3809	3813	3822	3826	3862	3875	
	3886	4581	4590	4609	4618	4637	4646	4672	4737	4762	5415	5422	5907	
	5918													
BIT11 = 004000	598#	1424	2311	2605	3140	3164	3259	3325	3479	3511	3539	3548	3557	
	3566	3718	3722	3862	3875	3886	4469	4478	4497	4506	4525	4534	4553	
	4562	4581	4590	4609	4618	4637	4646	4672	4698	4737	4762			
BIT12 = 010000	597#	611	613	615	617	2101	2339	3686	4490	5413	5422	6021	6024	
BIT13 = 020000	596#	612	613	616	617	2127	2367	3674	4462	5455	5463	5498		
BIT14 = 040000	595#	614	615	616	617	1064	2395	3662	4934	4984	4987	6021	6024	
BIT15 = 100000	594#	2182	2186	2192	2193	3650	4434	5453	5463	5498	5670	5686		
BIT2 = 000004	607#	991	1424	1860	2475	2784	3140	3164	3597	3675	3687	3717	3721	
	3730	3743	3756	3769	3782	3795	3808	3821	3998	4018	4020	4029	4031	
	4040	4042	4051	4053	4099	4106	4113	4147	4148	4171	4173	4187	4197	
	4204	4224	4230	4252	4256	4258	4275	4277	4294	4296	4324	4383	4403	
	4439	4467	4495	4523	4551	4574	4579	4607	4635	4722	4785	4837	4846	
	4861	4876	4918	4929	4953	5235	5280	5325	5636	5671	5731	5774	5818	
	5864	5915	5961	5980	5983	5985	5988							











L.DATA	031754	4963	4985	5007	5029	5051	5073	5095	5117	5139	5161	5183	5205	5249
		5294	5339	6040#										
LOO.03	001422	793#	1548*	1583										
LO4.07	001424	794#	1550*	1588										
LO8.11	001426	795#	1552*	1593										
L12.15	001430	796#	1554*	1598										
MAR17	014226	3136#												
MASKX	001244	689#												
MASK.A	001406	783#	1584											
MASK.B	001410	784#	1589											
MASK.C	001412	785#	1594											
MASK.D	001414	786#	1599											
MASTEK	005400	1340	1498#											
MCRLF	005104	1109	1232	1336	1337	1345	1486	1498#	1607	1625				
MCSRX	005330	1015	1498#											
MDATA	005624	1259	1269	1511#										
MEMEXT	014622	3210	3226	3236#	5944	5956								
MEPASS	005145	1014	1498#											
MERRPC	005454	1343	1498#											
MERRX	005355	1021	1498#											
MERR2	005174	1498#	1524	1714										
MERR3	005243	969	1498#											
MLOCK	005301	993	1498#											
MNEW	005402	964	1498#											
MPASSX	005344	1019	1498#											
MPFAIL	005107	1399	1498#											
MOM	005100	1139	1498#	1630										
MR	005171	1001	1498#											
MRESET=	004000	619#	1418	1431	3330									
MSTCLR	104412	749#	1403	3403	3439	3509	3535	3593	3647	3660	3672	3684	3710	3856
		3914	3971	4096	4144	4162	4219	4241	4250	4273	4292	4321	4361	4432
		4448	4460	4476	4488	4504	4516	4532	4544	4560	4572	4588	4600	4616
		4628	4644	4670	4719	4834	4859	4874	4902	4956	4978	5000	5022	5044
		5066	5088	5110	5132	5154	5176	5198	5239	5284	5329	5367	5407	5447
		5490	5530	5555	5580	5605	5631	5664	5717	5760	5803	5849	5895	5943
MTITLE	001000	657#	946											
MTSTN	005366	1341	1498#	1609										
MTSTPC	005267	1498#												
MVECX	005336	1017	1498#											
NEXT	001216	674#	1082	1372	1759*	1790*	1858*	1884*	1910*	1936*	1962*	1990*	2021*	2047*
		2073*	2099*	2125*	2151*	2179*	2208*	2232*	2253*	2281*	2309*	2337*	2365*	2393*
		2421*	2447*	2473*	2499*	2525*	2551*	2577*	2603*	2630*	2667*	2707*	2730*	2756*
		2782*	2808*	2834*	2860*	2886*	2912*	2942*	2984*	3043*	3087*	3202*	3254*	3320*
		3402*	3438*	3508*	3534*	3591*	3646*	3708*	3855*	3913*	3969*	4095*	4139*	4320*
		4360*	4428*	4669*	4718*	4833*	4901*	4950*	5233*	5278*	5323*	5365*	5405*	5445*
		5488*	5528*	5553*	5578*	5603*	5630*	5662*	5715*	5758*	5801*	5847*	5894*	5942*
NOLIST	***** U	539												
NO.ATR	032232	5369	5386	5409	5426	5449	5467	5532	5557	5582	6104#			
NO.BTR	032236	5370	5387	5410	5427	5450	5468	5533	5558	5583	5608	6108#		
NPR	040000	614#	5729	5772	5820	5866	5905							
NPRLOC	033352	5721	5726*	5734	5739*	5764	5769*	5777	5783*	5807	5812*	5815*	5823	5853
		5858*	5861*	5869	6115#									
PARAM -	104405	739#	1610											
PARAM1	003304	1159#	1176											
PARBIT-	040000	619#	1654											
PARERR	003360	1162	1164	1166	1175#	1182	1184	1186						









TST115	027452	5488	5527#	
TST116	027542	5528	5552#	
TST117	027632	5553	5577#	
TST12	010326	2021	2046#	
TST120	027722	5578	5602#	
TST121	030014	5603	5629#	
TST122	030116	5630	5661#	
TST123	030316	5662	5714#	
TST124	030456	5715	5757#	
TST125	030620	5758	5800#	
TST126	031002	5801	5846#	
TST127	031166	5847	5893#	
TST13	010402	2047	2072#	
TST130	031344	5894	5941#	6113
TST131=	***** U	5942		
TST14	010456	2073	2098#	
TST15	010532	2099	2124#	
TST16	010606	2125	2150#	
TST17	010662	2151	2178#	
TST2	007370	1759	1789#	
TST20	010756	2179	2207#	
TST21	011034	2208	2231#	
TST22	011074	2232	2252#	
TST23	011160	2253	2280#	
TST24	011244	2281	2308#	
TST25	011330	2309	2336#	
TST26	011414	2337	2364#	
TST27	011500	2365	2392#	
TST3	007622	1790	1857#	
TST30	011564	2393	2420#	
TST31	011640	2421	2446#	
TST32	011714	2447	2472#	
TST33	011770	2473	2498#	
TST34	012044	2499	2524#	
TST35	012120	2525	2550#	
TST36	012174	2551	2576#	
TST37	012250	2577	2602#	
TST4	007676	1858	1883#	
TST40	012324	2603	2629#	
TST41	012426	2630	2666#	
TST42	012536	2667	2706#	
TST43	012600	2707	2729#	
TST44	012654	2730	2755#	
TST45	012730	2756	2781#	
TST46	013004	2782	2807#	
TST47	013060	2808	2833#	
TST5	007752	1884	1909#	
TST50	013134	2834	2859#	
TST51	013210	2860	2885#	
TST52	013264	2886	2911#	
TST53	013340	2912	2941#	
TST54	013474	2942	2983#	
TST55	013676	2984	3042#	
TST56	014036	3043	3085#	
TST57	014434	3087	3201#	
TST6	010026	1910	1935#	



CROSS REFERENCE TABLE -- USER SYMBOLS

SN = 000130

1962	1963#	1990	1991#	2021	2022#	2047	2048#	2073	2074#	2099	2100#	2125
2126#	2151	2152#	2179	2180#	2208	2209#	2232	2233#	2253	2254#	2281	2282#
2309	2310#	2337	2338#	2365	2366#	2393	2394#	2421	2422#	2447	2448#	2473
2474#	2499	2500#	2525	2526#	2551	2552#	2577	2578#	2603	2604#	2630	2631#
2667	2668#	2707	2708#	2730	2731#	2756	2757#	2782	2783#	2808	2809#	2834
2835#	2860	2861#	2886	2887#	2912	2913#	2942	2943#	2984	2986#	3043	3045#
3087	3089#	3202	3204#	3254	3255#	3320	3321#	3402	3403#	3438	3439#	3508
3509#	3534	3535#	3591	3593#	3646	3647#	3708	3710#	3855	3856#	3913	3914#
3969	3971#	4095	4096#	4139	4140#	4320	4321#	4360	4361#	4428	4429#	4669
4670#	4718	4719#	4833	4834#	4901	4902#	4950	4951#	5233	5234#	5278	5279#
5323	5324#	5365	5366#	5405	5406#	5445	5446#	5488	5489#	5528	5529#	5553
5554#	5578	5579#	5603	5604#	5630	5631#	5662	5664#	5715	5717#	5758	5760#
5801	5803#	5847	5849#	5894	5895#	5942	5943#	1849	1855	1859#	1875	1881
1#	1751	1756	1761#	1780	1787	1792#	1849	1855	1859#	1875	1881	1885#
1901	1907	1911#	1927	1933	1937#	1953	1959	1963#	1979	1987	1991#	2012
2018	2022#	2038	2044	2048#	2064	2070	2074#	2090	2096	2100#	2116	2122
2126#	2142	2148	2152#	2168	2176	2180#	2200	2205	2209#	2222	2229	2233#
2244	2250	2254#	2272	2278	2282#	2300	2306	2310#	2328	2334	2338#	2356
2362	2366#	2384	2390	2394#	2412	2418	2422#	2438	2444	2448#	2464	2470
2474#	2490	2496	2500#	2516	2522	2526#	2542	2548	2552#	2568	2574	2578#
2594	2600	2604#	2620	2627	2631#	2657	2664	2668#	2695	2704	2708#	2721
2727	2731#	2747	2753	2757#	2773	2779	2783#	2799	2805	2809#	2825	2831
2835#	2851	2857	2861#	2877	2883	2887#	2903	2909	2913#	2929	2939	2943#
2974	2981	2986#	3033	3040	3045#	3076	3083	3089#	3186	3199	3204#	3244
3251	3255#	3310	3317	3321#	3375	3392	3399	3403#	3421	3435	3439#	3499
3505	3509#	3521	3531	3535#	3579	3588	3593#	3630	3643	3647#	3699	3705
3710#	3843	3852	3856#	3901	3910	3914#	3960	3966	3971#	4078	4092	4096#
4122	4136	4140#	4309	4317	4321#	4349	4357	4361#	4409	4425	4429#	4657
4666	4670#	4708	4715	4719#	4811	4820	4830	4834#	4892	4898	4902#	4940
4947	4951#	5222	5230	5234#	5267	5275	5279#	5312	5320	5324#	5357	5362
5366#	5397	5402	5406#	5437	5442	5446#	5478	5485	5489#	5519	5525	5529#
5544	5550	5554#	5569	5575	5579#	5594	5600	5604#	5620	5627	5631#	5652
5659	5664#	5702	5712	5717#	5745	5755	5760#	5789	5798	5803#	5835	5844
5849#	5882	5891	5895#	5928	5939	5943#	6113#	739#	741#	743#	745#	747#
1#	720#	729	731#	733#	735#	737#	739#	741#	743#	745#	747#	749#
751#	753#	755#	757#	759#								
630#	631	634#	641#	642#	643#	644#	647#	649#	652#	656#	658#	703#
704#	705#	706#	707#	708#	819#	821#	822#	823#	824#	825#	826#	827#
828#	829#	830#	832#	833#	834#	835#	836#	837#	838#	839#	840#	841#
843#	844#	845#	846#	847#	848#	849#	850#	851#	852#	854#	855#	856#
857#	858#	859#	860#	861#	862#	863#	865#	866#	867#	868#	869#	870#
871#	872#	873#	874#	876#	877#	878#	879#	880#	881#	882#	883#	884#
885#	887#	888#	889#	890#	891#	892#	893#	894#	895#	896#	898#	899#
900#	901#	902#	903#	904#	905#	906#	907#	971	1060	1307	1389	1398
1412	1450#	1460	1467	1481	1508#	1510#	1512#	1526	1717	1738	3241#	4679
4745	4748	4771	5247#	5248#	5250#	5251#	5292#	5293#	5295#	5296#	5337#	5338#
5340#	5341#	5502	5674	5913	5923	6113#						
987#												

.BEGIN 002332  
 .CNVRT 003542  
 .CONVR 003536  
 .DATAC 004576  
 .DELAY 004476  
 .EOP 002436  
 .ERRTA 033520  
 .HLT 004002  
 .INSTE 003224

748 1233#  
 746 1232#  
 758 1438#  
 754 1409#  
 1009# 5942  
 1328 6168#  
 637 1301#  
 738 1139#

.INSTR	003120	736	1118#		
.INST1	003140	1122#	1142		
.MSG	003142	1120*	1123#		
.MSTCL	004556	750	1430#		
.PARAM	003244	740	1150#		
.PFAIL	004402	635	920	1386#	1594
.RAMCL	004516	752	1417#		
.RES05	003504	744	1221#		
.ROMCL	004566	756	1434#		
.SAV05	003444	742	1207#		
.SCOPE	002634	730	1054#		
.SCOPI	003020	732	1092#		
.START	001742	653	918#	930	
.TRPSR	003750	639	1289#		
.TRPTA	001314	728#	1294		
.TYPE	003044	734	1102#		



\$NSR3	538#	5881													
\$PFAL	1#	1382													
\$PRIO	538#	5518	5543	5568	5593										
\$RAMCL	1#	1409													
\$RXSHI	1#														
\$SCOPE	1#	1050													
\$SECT1	535#	2974													
\$SECT2	535#	3033													
\$SECT3	535#	3076													
\$SETCL	538#	3578													
\$SETLI	1#														
\$SETSC	1#	6015													
\$SETSY	1#														
\$SET.T	1#														
\$SILOI	1#														
\$SIMBC	1#	6056													
\$TRPDE	1#	729	731	733	735	737	739	741	743	745	747	749	751	753	755
	757														
\$TSTN	1#	1756	1787	1855	1881	1907	1933	1959	1987	2018	2044	2070	2096	2122	2148
	2176	2205	2229	2250	2278	2306	2334	2362	2390	2418	2444	2470	2496	2522	2548
	2574	2600	2627	2664	2704	2727	2753	2779	2805	2831	2857	2883	2909	2939	2981
	3040	3083	3199	3251	3317	3399	3435	3505	3531	3588	3643	3705	3852	3910	3966
	4092	4136	4317	4357	4425	4666	4715	4830	4898	4947	5230	5275	5320	5360	5402
	5442	5485	5525	5550	5575	5600	5627	5659	5712	5755	5798	5844	5891	5939	
\$TXSHI	1#														
\$VARIA	1#	655													
\$XFER	538#	3959													
\$XXCRC	538#	4939													
\$XZ	1#	1751	1754	1780	1785	1849	1853	1875	1879	1901	1905	1927	1931	1953	1957
	1979	1985	2012	2016	2038	2042	2064	2068	2090	2094	2116	2120	2142	2146	2168
	2174	2200	2203	2222	2227	2244	2248	2272	2276	2300	2304	2328	2332	2356	2360
	2384	2388	2412	2416	2438	2442	2464	2468	2490	2494	2516	2520	2542	2546	2568
	2572	2594	2598	2620	2624	2657	2661	2695	2702	2721	2725	2747	2751	2773	2777
	2799	2803	2825	2829	2851	2855	2877	2881	2903	2907	2929	2937	2974	2979	3033
	3038	3076	3081	3186	3196	3244	3249	3310	3315	3375	3390	3392	3397	3421	3433
	3499	3503	3521	3529	3579	3586	3630	3641	3699	3703	3843	3850	3901	3908	3960
	3964	4078	4090	4122	4134	4309	4315	4349	4355	4409	4423	4657	4664	4708	4713
	4811	4818	4820	4828	4892	4896	4940	4945	5222	5228	5267	5273	5312	5318	5357
	5360	5397	5400	5437	5440	5478	5483	5519	5523	5544	5548	5569	5573	5594	5598
	5620	5625	5652	5657	5702	5710	5745	5753	5789	5796	5835	5842	5882	5889	5928
	5937														

. ABS. 033644 000

ERRORS DETECTED: 0

CZDVAC.BIN,CZDVAC.LST/CRF/SOL/NL:TOC=CZDVXX.SML,CZDVAC.P11

RUN-TIME: 52 75 6 SECONDS

RUN-TIME RATIO: 2158/134-16.0

CORE USED: 34K (67 PAGES)