

DR11C

DEVICE REGISTER TEST
CZDRCH0

AH-8648H-MC
COPYRIGHT © 72-78
FICHE 1 OF 1

DEC 1978
digital
MADE IN USA

TEST	TEST	TEST
TEST 1	TEST 1	TEST 1
TEST 2	TEST 2	TEST 2
TEST 3	TEST 3	TEST 3
TEST 4	TEST 4	TEST 4
TEST 5	TEST 5	TEST 5
TEST 6	TEST 6	TEST 6
TEST 7	TEST 7	TEST 7
TEST 8	TEST 8	TEST 8
TEST 9	TEST 9	TEST 9
TEST 10	TEST 10	TEST 10
TEST 11	TEST 11	TEST 11
TEST 12	TEST 12	TEST 12
TEST 13	TEST 13	TEST 13
TEST 14	TEST 14	TEST 14
TEST 15	TEST 15	TEST 15
TEST 16	TEST 16	TEST 16
TEST 17	TEST 17	TEST 17
TEST 18	TEST 18	TEST 18
TEST 19	TEST 19	TEST 19
TEST 20	TEST 20	TEST 20
TEST 21	TEST 21	TEST 21
TEST 22	TEST 22	TEST 22
TEST 23	TEST 23	TEST 23
TEST 24	TEST 24	TEST 24
TEST 25	TEST 25	TEST 25
TEST 26	TEST 26	TEST 26
TEST 27	TEST 27	TEST 27
TEST 28	TEST 28	TEST 28
TEST 29	TEST 29	TEST 29
TEST 30	TEST 30	TEST 30
TEST 31	TEST 31	TEST 31
TEST 32	TEST 32	TEST 32
TEST 33	TEST 33	TEST 33
TEST 34	TEST 34	TEST 34
TEST 35	TEST 35	TEST 35
TEST 36	TEST 36	TEST 36
TEST 37	TEST 37	TEST 37
TEST 38	TEST 38	TEST 38
TEST 39	TEST 39	TEST 39
TEST 40	TEST 40	TEST 40
TEST 41	TEST 41	TEST 41
TEST 42	TEST 42	TEST 42
TEST 43	TEST 43	TEST 43
TEST 44	TEST 44	TEST 44
TEST 45	TEST 45	TEST 45
TEST 46	TEST 46	TEST 46
TEST 47	TEST 47	TEST 47
TEST 48	TEST 48	TEST 48
TEST 49	TEST 49	TEST 49
TEST 50	TEST 50	TEST 50
TEST 51	TEST 51	TEST 51
TEST 52	TEST 52	TEST 52
TEST 53	TEST 53	TEST 53
TEST 54	TEST 54	TEST 54
TEST 55	TEST 55	TEST 55
TEST 56	TEST 56	TEST 56
TEST 57	TEST 57	TEST 57
TEST 58	TEST 58	TEST 58
TEST 59	TEST 59	TEST 59
TEST 60	TEST 60	TEST 60
TEST 61	TEST 61	TEST 61
TEST 62	TEST 62	TEST 62
TEST 63	TEST 63	TEST 63
TEST 64	TEST 64	TEST 64
TEST 65	TEST 65	TEST 65
TEST 66	TEST 66	TEST 66
TEST 67	TEST 67	TEST 67
TEST 68	TEST 68	TEST 68
TEST 69	TEST 69	TEST 69
TEST 70	TEST 70	TEST 70
TEST 71	TEST 71	TEST 71
TEST 72	TEST 72	TEST 72
TEST 73	TEST 73	TEST 73
TEST 74	TEST 74	TEST 74
TEST 75	TEST 75	TEST 75
TEST 76	TEST 76	TEST 76
TEST 77	TEST 77	TEST 77
TEST 78	TEST 78	TEST 78
TEST 79	TEST 79	TEST 79
TEST 80	TEST 80	TEST 80
TEST 81	TEST 81	TEST 81
TEST 82	TEST 82	TEST 82
TEST 83	TEST 83	TEST 83
TEST 84	TEST 84	TEST 84
TEST 85	TEST 85	TEST 85
TEST 86	TEST 86	TEST 86
TEST 87	TEST 87	TEST 87
TEST 88	TEST 88	TEST 88
TEST 89	TEST 89	TEST 89
TEST 90	TEST 90	TEST 90
TEST 91	TEST 91	TEST 91
TEST 92	TEST 92	TEST 92
TEST 93	TEST 93	TEST 93
TEST 94	TEST 94	TEST 94
TEST 95	TEST 95	TEST 95
TEST 96	TEST 96	TEST 96
TEST 97	TEST 97	TEST 97
TEST 98	TEST 98	TEST 98
TEST 99	TEST 99	TEST 99
TEST 100	TEST 100	TEST 100

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

.REM %

IDENTIFICATION

PRODUCT CODE: AC-8647H-MC
PRODUCT NAME: CZDRCH0 DR11C DEVICE REGISTER
DATE : APRIL, 1977
MAINTAINER: DIAGNOSTIC GROUP

COPYRIGHT (C) DIGITAL EQUIPMENT CORPORA ON
1972,1978
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.
THE SOFTWARE DESCRIBED IN THIS DOCUMENT IS FURNISHED UNDER A
LICENSE AND MAY ONLY BE USED OR COPIED IN ACCORDANCE WITH
THE TERMS OF SUCH LICENSE.
DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY
FOR THE USE OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT
THAT IS NOT SUPPLIED BY DIGITAL.

45
46
47
48
49
50
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

1. ABSTRACT

THIS IS A LOGIC TEST OF THE DR11C. FOR THIS TEST TO OPERATE
A SPECIAL MAINTENANCE CABLE MUST BE CONNECTED (BC08R).
THIS TEST WILL CHECK UP TO 32 SEQUENTIAL DR11C'S.

2. REQUIREMENTS

2.1 EQUIPMENT

PDP-11 STANDARD COMPUTER

DR11C

BC08R FOR EACH DR11C

2.2 STORAGE

2.2.1 PROGRAM STORAGE - THE ROUTINE USES MEMORY
FROM 0000 TO 5200.

3. LOADING PROCEDURE

3.1 METHOD

ABSOLUTE LOADER

4. STARTING PROCEDURE

4.1 CONTROL SWITCH SETTING

FOR SWITCHLESS PROCESSORS, THE PROGRAM WILL USE THE
CONTENTS OF LOCATION 176 AS THE VALUE OF THE SWITCHES
IF NO HARDWARE SWITCH REGISTER IS FOUND. THE PROGRAM
ALLOWS THE DYNAMIC LOADING OF THE SOFTWARE SWITCH REGISTER
THROUGH THE CONTROL G (^G) ROUTINE. LOCATION 174
WILL BE USED AS THE SOFTWARE DISPLAY REGISTER.

STARTING AT SA 200 ALL SWITCHES SHOULD BE DOWN OR ZERO.
(IF NOT ZERO, BIT 0 TO 8 WILL BE STARTING VECTOR.)

4.2 STARTING ADDRESS OR ADDRESSES

(A) 200 = START OF TEST--FOR NORMAL TESTING
(B) 204 = SPECIAL ENTRANCE --FOR TESTING UNIQUE DR11C
(C) 210 = RESTART--FOR STARTING AFTER SHUT DOWN

4.3 PROGRAM AND/OR OPERATOR ACTION

LOAD PROGRAM INTO MEMORY.

101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156

SET SWITCH REGISTER TO STARTING ADDRESS.
LOAD ADDRESS.
PRESS START.
THE PROGRAM WILL STAY IN SECTION AND LOOP.

4.3.1 FOR SPECIAL ENTRANCE - SA204

1ST HALT SET SWITCH REGISTER EQUAL TO CSR ADDRESS OF DR11C
PRESS CONTINUE
2ND HALT SET SWITCH REGISTER EQUAL TO VECTOR ADDRESS OF DR11C
PRESS CONTINUE
RAISE SWITCH 10 TO '1' TO INHIBIT SEQUENCING TO NEXT DR11C

5. OPERATING PROCEDURE

5.1 OPERATIONAL SWITCH SETTINGS

5.1.1 AT SA 200 .. THE INSTRUCTION AND LOGIC TEST.
WITH ALL SWITCHES DOWN THE PROGRAM WILL PRINT
OUT ON ERRORS AND CONTINUE IN TEST. (* WILL
BE PRINTED AT COMPLETION OF TESTING EACH DR11C)

5.1.2 SWITCH SETTINGS ARE

SW15 = 1 OR UP ... HALT ON ERROR
SW14 = 1 OR UP ... SCOPE LOOP
SW13 = 1 OR UP ... INHIBIT PRINTOUT
SW12 = 1 OR UP ... NOT USED
SW11 = 1 OR UP ... INHIBIT ITERATION LOOP
SW10 = 1 OR UP ... DO NOT ADVANCE TO NEXT DR11C
SW09 = 1 OR UP ... INHIBIT PRINTOUT OF DEVICE TESTED.

5.1.3

5.2. SUBROUTINE ABSTRACTS

5.2.1 BEGIN SA 200

5.2.2 SCOPE

THIS SUBROUTINE CALL IS PLACED BETWEEN EACH SUBTEST
IN THE INSTRUCTION SECTION. IT RECORDS THE STARTING
ADDRESS OF EACH SUB-TEST AS IT IS BEING ENTERED.
IF A SCOPE LOOP IS REQUESTED, IT WILL JUMP TO THE
START OF THE SUBTEST THAT THE SCOPE LOOP IS RE-
QUESTED FOR.

5.2.3 HALT

IS A ROUTINE THAT PRINTS-OUT AN ADDRESS THAT TAGS
THE FAILING SUBTEST, AND THE INCORRECT DATA AT
THE TIME OF THE FAILURE. SEE 6.1

(5. OPERATING PROCEDURE CONT'D)

157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212

5.3 PROGRAM AND/OR OPERATOR ACTION

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

6. ERRORS

6.1 ERROR PRINTOUT

ARE IN A FOUR WORD FORMAT. THE 1ST IS THE PC+2 OF THE DETECTED ERROR. THE 2ND IS THE PROCESSOR STATUS REGISTER. THE 3RD IS DEVICE ADDRESS, THE 4TH IS VECTOR ADDRESS.

6.2 ERROR RECOVERY

DEPRESS CONTINUE TO RESTART SECTION

7. RESTRICTIONS

7.1 STARTING RESTRICTION

NONE

7.2 OPERATIONAL RESTRICTION

THE DR11C MUST HAVE THE BC08R CABLE TO RUN THIS TEST.

NOTE THAT THE DR11C HAS FLOATING VECTORS:

THE BELOW IS THE ASSIGNMENT OF FLOATING VECTORS, THE ASSIGNED SEQUENCES ARE:

1. STARTING AT 300 AND WORKING UPWARD ALL DC11'S WILL BE ASSIGNED.
2. THEN ANY EXTRA KL11 CALLED FOR (VT05, VT06, LC11)
3. THEN ANY DP11 CALLED FOR.
4. THEN ANY DM11 CALLED FOR.
5. THEN ANY DN11 CALLED FOR.
6. THEN ANY DM11BB CALLED FOR.
7. THEN ANY DR11A CALLED FOR.
8. THEN ANY DR11C CALLED FOR.

213 THE DR11A AND DR11C DEVICE ADDRESSES WILL BE ASSIGNED IN THE USER
214 AREA OF 767776 TO 764000. THE ASSIGNMENT OF ADDRESSES WILL START AT
215 THE HIGH ADDRESS LIMIT AND PROCEED DOWNWARD. USERS AND SPECIAL SYSTEMS
216 SHOULD START THEIR ASSIGNMENTS OF SPECIAL DEVICES AT THE LOW ADDRESS
217 LIMIT AND WORK UP. AFTER ASSIGNING ALL DR11A'S, ASSIGN DR11C'S
218
219 767776 TO 767770 DR11C #0 ;ASSUMING NO DR11A'S
220 767766 TO 767760 DR11C #1
221 :
222 :
223 :
224 767706 TO 767700 DR11C #7
225 :
226 :
227 :
228 767606 TO 767600 DR11C #15
229
230 8. MISCELLANEOUS
231
232 WHERE THERE ARE MULTIPLE DR11C OR A SYSTEM AND IT IS DESIRED TO
233 TEST ONLY ONE OF THEM. THIS MAY BE ACHIEVED BY USING THE SPECIAL
234 STARTING ADDRESS AND PLACING SW10 ON A ONE (UP) TO INHIBIT SEQUENCING
235 TO THE NEXT DR11C. SEE 4.3.1.
236
237 8.1 EXECUTION TIME
238
239 FOR EACH DR11C ABOUT 1 MINUTE
240
241 8.2 UNTESTED LOGIC
242
243 SIGNALS TO USER NOT TESTED:
244 'NEW DATA READY'
245 'DATA TRANSMITTED'
246 'INIT' TO THE USER
247
248 9. PROGRAM DESCRIPTION
249
250 THIS PROGRAM WHEN STARTED AT 200 CHECKS THE STANDARD DR11-C'S
251
252 THE PROGRAM THEN PERFORMES AN INCREMENTAL LOGIC CHECK FOR
253 THE SELECTED DR11C.
254
255 THE DATA REGISTER IS TESTED TO SEE IF 'RESET' CLEARS IT,
256 AND IF IT WILL HOLD ALL COMBINATIONS OF NUMBERS.
257
258 THE READ/WRITE BITS OF THE STATUS REGISTER ARE ALSO TESTED.
259
260 BOTH THE 'A' AND 'B' INTERRUPTS ARE TESTED TO SEE IF THEY
261 INTERRUPT AT THE CORRECT BUS REQUEST LEVEL BR-5.
262
263 AT THE END OF THE TEST AN '*' IS TYPED AND ALSO THE
264 ADDRESSES OF THE DR11-C CONTROL STATUS REGISTER AND
265 IT'S SIDE INTERRUPT VECTOR IS TYPED (IF SELECTED VIA
266 SWITCH 9.). THE PROGRAM THEN RETESTS THE UNIT (IF SELECTED
267 VIA SWITCH 10) OR SCANS TO THE NEXT DR11-C. IF ANOTHER
268 DR11-C IS ON THE SYSTEM THEN THE PROGRAM RESTARTS TESTING

269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286

THE NEW DR11-C.

AFTER ALL DR11-C'S HAVE BEEN TESTED THE PROGRAM WILL
TYPE 'END PASS' AND RESTART TESTING WITH THE INITIAL DR11-C.

IF NO ERRORS OCCUR AND THREE DR11-C'S ARE AVAILABLE
AND SWITCH 9 IS DOWN THE PROGRAM WILL TYPE.

160000 770 *
157770 1000 *
157760 1010 *

/

ETC.

IF SWITCH 9 IS UP THEN

*
*
*

IF A POWER FAIL OCCURS THE PROGRAM WILL RESTART AT 'START'.

```
287  
288  
289          10.    LISTING  
290  
291          %  
292  
293  
294  
295          ;GENERAL REGISTER LOGIC TEST  
296  
297          177776      PSW=177776  
298          104000      HLT=104000  
299          167770      CSR= 67770  
300          ;REGISTER DEFINITIONS  
301          000000      R0=%0  
302          000001      R1=%1  
303          000002      R2=%2  
304          000003      R3=%3  
305          000004      R4=%4  
306          000005      R5=%5  
307          000006      SP=%6  
308          000007      PC=%7  
309  
310          ;SWITCHES  
311  
312          001000      SW9=1000  
313          002000      SW10=2000  
314          004000      SW11=4000  
315          020000      SW13=20000  
316          040000      SW14=40000  
317  
318  
319          .SBTTL  BASIC DEFINITIONS  
320  
321          ;*INITIAL ADDRESS OF THE STACK POINTER *** 1200 ***  
322          001200      STACK= 1200  
323          .EQUIV  EMT,ERROR      ;;BASIC DEFINITION OF ERROR CALL  
324          .EQUIV  IOT,SCOPE      ;;BASIC DEFINITION OF SCOPE CALL  
325  
326          ;*MISCELLANEOUS DEFINITIONS  
327          000011      HT= 11      ;;CODE FOR HORIZONTAL TAB  
328          000012      LF= 12      ;;CODE FOR LINE FEED  
329          000015      CR= 15      ;;CODE FOR CARRIAGE RETURN  
330          000200      CRLF= 200    ;;CODE FOR CARRIAGE RETURN-LINE FEED  
331          177776      PS= 177776  ;;PROCESSOR STATUS WORD  
332          .EQUIV  PS,PSW  
333          177774      STKLMT= 177774 ;;STACK LIMIT REGISTER  
334          177772      PIRQ= 177772 ;;PROGRAM INTERRUPT REQUEST REGISTER  
335          177570      DSWR= 177570 ;;HARDWARE SWITCH REGISTER  
336          177570      DDISP= 177570 ;;HARDWARE DISPLAY REGISTER  
337  
338          ;*GENERAL PURPOSE REGISTER DEFINITIONS  
339          000000      R0= %0      ;;GENERAL REGISTER  
340          000001      R1= %1      ;;GENERAL REGISTER  
341          000002      R2= %2      ;;GENERAL REGISTER  
342          000003      R3= %3      ;;GENERAL REGISTER
```



```
343      000004      R4=    %4      ;;GENERAL REGISTER
344      000005      R5=    %5      ;;GENERAL REGISTER
345      000006      R6=    %6      ;;GENERAL REGISTER
346      000007      R7=    %7      ;;GENERAL REGISTER
347      000006      SP=    %6      ;;STACK POINTER
348      000007      PC=    %7      ;;PROGRAM COUNTER
349
350      ;*PRIORITY LEVEL DEFINITIONS
351      000000      PR0=    0      ;;PRIORITY LEVEL 0
352      000040      PR1=    40     ;;PRIORITY LEVEL 1
353      000100      PR2=   100     ;;PRIORITY LEVEL 2
354      000140      PR3=   140     ;;PRIORITY LEVEL 3
355      000200      PR4=   200     ;;PRIORITY LEVEL 4
356      000240      PR5=   240     ;;PRIORITY LEVEL 5
357      000300      PR6=   300     ;;PRIORITY LEVEL 6
358      000340      PR7=   340     ;;PRIORITY LEVEL 7
359
360      ;*'SWITCH REGISTER' SWITCH DEFINITIONS
361      100000      SW15= 100000
362      040000      SW14=  40000
363      020000      SW13=  20000
364      010000      SW12=  10000
365      004000      SW11=  4000
366      002000      SW10=  2000
367      001000      SW09=  1000
368      000400      SW08=   400
369      000200      SW07=   200
370      000100      SW06=   100
371      000040      SW05=   40
372      000020      SW04=   20
373      000010      SW03=   10
374      000004      SW02=    4
375      000002      SW01=    2
376      000001      SW00=    1
377      .EQUIV SW09,SW9
378      .EQUIV SW08,SW8
379      .EQUIV SW07,SW7
380      .EQUIV SW06,SW6
381      .EQUIV SW05,SW5
382      .EQUIV SW04,SW4
383      .EQUIV SW03,SW3
384      .EQUIV SW02,SW2
385      .EQUIV SW01,SW1
386      .EQUIV SW00,SW0
387
388      ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
389      100000      BIT15= 100000
390      040000      BIT14=  40000
391      020000      BIT13=  20000
392      010000      BIT12=  10000
393      004000      BIT11=  4000
394      002000      BIT10=  2000
395      001000      BIT09=  1000
396      000400      BIT08=   400
397      000200      BIT07=   200
398      000100      BIT06=   100
```

```
399          000040      BIT05= 40
400          000020      BIT04= 20
401          000010      BIT03= 10
402          000004      BIT02= 4
403          000002      BIT01= 2
404          000001      BIT00= 1
405          .EQUIV BIT09,BIT9
406          .EQUIV BIT08,BIT8
407          .EQUIV BIT07,BIT7
408          .EQUIV BIT06,BIT6
409          .EQUIV BIT05,BIT5
410          .EQUIV BIT04,BIT4
411          .EQUIV BIT03,BIT3
412          .EQUIV BIT02,BIT2
413          .EQUIV BIT01,BIT1
414          .EQUIV BIT00,BIT0
415
416          ;*BASIC "CPU" TRAP VECTOR ADDRESSES
417          000004      ERRVEC= 4          ;;TIME OUT AND OTHER ERRORS
418          000010      RESVEC= 10         ;;RESERVED AND ILLEGAL INSTRUCTIONS
419          000014      TBITVEC=14         ;;'T' BIT
420          000014      TRTVEC= 14         ;;TRACE TRAP
421          000014      BPTVEC= 14         ;;BREAKPOINT TRAP (BPT)
422          000020      IOTVEC= 20         ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
423          000024      PWRVEC= 24         ;;POWER FAIL
424          000030      EMTVEC= 30         ;;EMULATOR TRAP (EMT) **ERROR**
425          000034      TRAPVEC=34         ;;'TRAP' TRAP
426          000060      TKVEC= 60          ;;TTY KEYBOARD VECTOR
427          000064      TPVEC= 64          ;;TTY PRINTER VECTOR
428          000240      PIRQVEC=240        ;;PROGRAM INTERRUPT REQUEST VECTOR
429
430          .ENABLE ABS
431          000000      .-0
432          000002      .+2
433          000004      000000      HALT
434          000006      000000      .+2
435          000010      000012      HALT
436          000012      000000      .+2
437          000014      000016      HALT
438          000016      000000      .+2
439          000020      000022      HALT
440          000022      000000      .+2
441          000024      000026      HALT
442          000026      000000      .+2
443          000030      000032      HALT
444          000032      000000      .+2
445          000034      000036      HALT
446          000036      000000      .+2
447          000040      000042      HALT
448          000042      000000      .+2
449          000044      000046      HALT
450          000046      000000      .+2
451          000050      000052      HALT
452          000052      000000      .+2
453          000054      000056      HALT
454          000056      000000      .+2
```

455	000060	000062	.+2
456	000062	000000	HALT
457	000064	000066	.+2
458	000066	000000	HALT
459	000070	000072	.+2
460	000072	000000	HALT
461	000074	000076	.+2
462	000076	000000	HALT
463	000100	000102	.+2
464	000102	000000	HALT
465	000104	000106	.+2
466	000106	000000	HALT
467	000110	000112	.+2
468	000112	000000	HALT
469	000114	000116	.+2
470	000116	000000	HALT
471	000120	000122	.+2
472	000122	000000	HALT
473	000124	000126	.+2
474	000126	000000	HALT
475	000130	000132	.+2
476	000132	000000	HALT
477	000134	000136	.+2
478	000136	000000	HALT
479	000140	000142	.+2
480	000142	000000	HALT
481	000144	000146	.+2
482	000146	000000	HALT
483	000150	000152	.+2
484	000152	000000	HALT
485	000154	000156	.+2
486	000156	000000	HALT
487	000160	000162	.+2
488	000162	000000	HALT
489	000164	000166	.+2
490	000166	000000	HALT
491	000170	000172	.+2
492	000172	000000	HALT
493	000174	000176	.+2
494	000176	000000	HALT
495	000200	000202	.+2
496	000202	000000	HALT
497	000204	000206	.+2
498	000206	000000	HALT
499	000210	000212	.+2
500	000212	000000	HALT
501	000214	000216	.+2
502	000216	000000	HALT
503	000220	000222	.+2
504	000222	000000	HALT
505	000224	000226	.+2
506	000226	000000	HALT
507	000230	000232	.+2
508	000232	000000	HALT
509	000234	000236	.+2
510	000236	000000	HALT

511	000240	000242	.+2
512	000242	000000	HALT
513	000244	000246	.+2
514	000246	000300	HALT
515	000250	000252	.+2
516	000252	000000	HALT
517	000254	000256	.+2
518	000256	000000	HALT
519	000260	000262	.+2
520	000262	000000	HALT
521	000264	000266	.+2
522	000266	000000	HALT
523	000270	000272	.+2
524	000272	000000	HALT
525	000274	000276	.+2
526	000276	000000	HALT
527	000300	000302	.+2
528	000302	000000	HALT
529	000304	000306	.+2
530	000306	000000	HALT
531	000310	000312	.+2
532	000312	000000	HALT
533	000314	000316	.+2
534	000316	000000	HALT
535	000320	000322	.+2
536	000322	000000	HALT
537	000324	000326	.+2
538	000326	000000	HALT
539	000330	000332	.+2
540	000332	000000	HALT
541	000334	000336	.+2
542	000336	000000	HALT
543	000340	000342	.+2
544	000342	000000	HALT
545	000344	000346	.+2
546	000346	000000	HALT
547	000350	000352	.+2
548	000352	000000	HALT
549	000354	000356	.+2
550	000356	000000	HALT
551	000360	000362	.+2
552	000362	000000	HALT
553	000364	000366	.+2
554	000366	000000	HALT
555	000370	000372	.+2
556	000372	000000	HALT
557	000374	000376	.+2
558	000376	000000	HALT
559	000400	000402	.+2
560	000402	000000	HALT
561	000404	000406	.+2
562	000406	000000	HALT
563	000410	000412	.+2
564	000412	000000	HALT
565	000414	000416	.+2
566	000416	000000	HALT

567	000420	000422	.+2
568	000422	000000	HALT
569	000424	000426	.+2
570	000426	000000	HALT
571	000430	000432	.+2
572	000432	000000	HALT
573	000434	000436	.+2
574	000436	000000	HALT
575	000440	000442	.+2
576	000442	000000	HALT
577	000444	000446	.+2
578	000446	000000	HALT
579	000450	000452	.+2
580	000452	000000	HALT
581	000454	000456	.+2
582	000456	000000	HALT
583	000460	000462	.+2
584	000462	000000	HALT
585	000464	000466	.+2
586	000466	000000	HALT
587	000470	000472	.+2
588	000472	000000	HALT
589	000474	000476	.+2
590	000476	000000	HALT
591	000500	000502	.+2
592	000502	000000	HALT
593	000504	000506	.+2
594	000506	000000	HALT
595	000510	000512	.+2
596	000512	000000	HALT
597	000514	000516	.+2
598	000516	000000	HALT
599	000520	000522	.+2
600	000522	000000	HALT
601	000524	000526	.+2
602	000526	000000	HALT
603	000530	000532	.+2
604	000532	000000	HALT
605	000534	000536	.+2
606	000536	000000	HALT
607	000540	000542	.+2
608	000542	000000	HALT
609	000544	000546	.+2
610	000546	000000	HALT
611	000550	000552	.+2
612	000552	000000	HALT
613	000554	000556	.+2
614	000556	000000	HALT
615	000560	000562	.+2
616	000562	000000	HALT
617	000564	000566	.+2
618	000566	000000	HALT
619	000570	000572	.+2
620	000572	000000	HALT
621	000574	000576	.+2
622	000576	000000	HALT

623	000600	000602	.+2
624	000602	000000	HALT
625	000604	000606	.+2
626	000606	000000	HALT
627	000610	000612	.+2
628	000612	000000	HALT
629	000614	000616	.+2
630	000616	000000	HALT
631	000620	000622	.+2
632	000622	000000	HALT
633	000624	000626	.+2
634	000626	000000	HALT
635	000630	000632	.+2
636	000632	000000	HALT
637	000634	000636	.+2
638	000636	000000	HALT
639	000640	000642	.+2
640	000642	000000	HALT
641	000644	000646	.+2
642	000646	000000	HALT
643	000650	000652	.+2
644	000652	000000	HALT
645	000654	000656	.+2
646	000656	000000	HALT
647	000660	000662	.+2
648	000662	000000	HALT
649	000664	000666	.+2
650	000666	000000	HALT
651	000670	000672	.+2
652	000672	000000	HALT
653	000674	000676	.+2
654	000676	000000	HALT
655	000700	000702	.+2
656	000702	000000	HALT
657	000704	000706	.+2
658	000706	000000	HALT
659	000710	000712	.+2
660	000712	000000	HALT
661	000714	000716	.+2
662	000716	000000	HALT
663	000720	000722	.+2
664	000722	000000	HALT
665	000724	000726	.+2
666	000726	000000	HALT
667	000730	000732	.+2
668	000732	000000	HALT
669	000734	000736	.-2
670	000736	000000	HALT
671	000740	000742	.+2
672	000742	000000	HALT
673	000744	000746	.+2
674	000746	000000	HALT
675	000750	000752	.+2
676	000752	000000	HALT
677	000754	000756	.+2
678	000756	000000	HALT

679	000760	000762	
680	000762	000000	
681	000764	000766	
682	000766	000000	
683	000770	000772	
684	000772	000000	
685	000774	000776	
686	000776	000000	
687		000020	
688	000020	004652	
689	000022	000340	
690	000024	005124	
691	000026	000340	
692	000030	004126	
693	000032	000340	
694	000034	006562	
695	000036	000340	
696		000046	
697	000046	004112	
698		000052	
699	000052	000000	
700		000174	
701	000174	000000	
702	000176	000000	
703		000200	
704	000200	000137	001250
705	000204	000137	001616
706	000210	000137	001726
707		001200	
708			
709			
710			
711	001200	167770	
712	001202	167772	
713	001204	167774	
714	001206	167773	
715	001210	000300	
716	001212	000302	
717	001214	000304	
718			
719			
720			
721	001216	177570	
722	001220	177570	
723	001222	167770	
724	001224	167772	
725	001226	167774	
726	001230	167773	
727			
728	001232	000300	
729	001234	000302	
730	001236	000304	
731	001240	000000	
732			
733	001242	000000	
734	001244	000240	

```

.+2
HALT
.+2
HALT
.+2
HALT
.+2
HALT
.-20
.SCOPE
340
PFAIL
340
.HLT
340
$TRAP
340
.=46
$ENDAD
.=52
0
.=174
DISPRE: 0
SWREG: 0
.=200
JMP @#START1 ;INITIAL START
JMP @#SPEC ;TO SELECT UNIQUE ADDRESS AND VECTOR
JMP @#START ;RESTART
.=1200

;THIS TABLE CONTAINS INITIAL REGISTER AND VECTOR ADDRESSES
RCSR: CSR
CSR+2
CSR+4
CSR+3
RCSR1: 300
302
304

;THIS TABLE CONTAINS REGISTER AND VECTOR ADDRESSES OF THE DR11-C UNDER TEST
SWR: 177570
DISPLA: 177570
DRCSR: 167770 ;ADDRESS OF DR11-C STATUS REGISTER
DROBUF: 167772 ;ADDRESS OF DR OUTPUT BUFFER REG.
DRIBUF: 167774 ;ADDRESS OF DR INPUT BUFFER REG.
DRBH!O: 167773 ;HIGH BYTE OF OUTPUT BUFFER REG.

DRVECA: 300 ;INTERRUPT VECTOR OF UNIT UNDER TEST
DRLVL: 302
DRVECB: 304 ;INTERRUPT VECTOR
XORFLG: 0

COUNT: 0 ;COUNT LOCATION
PL: 240 ;PRIORITY LEVEL

```

```

735 001246 000000 PASCNT: 0 ;NUMBER OF PASSES COMPLETED
736
737 001250 START1:
738 .SBTTL INITIALIZE THE COMMON TAGS
739 001250 012706 001200 MOV #STACK,SP ;:SETUP THE STACK POINTER
740 ;:SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
741 ;:EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
742 001254 013746 000004 MOV @#ERRVEC,-(SP) ;:SAVE ERROR VECTOR
743 001260 012737 001314 000004 MOV #64$,@#ERRVEC ;:SET UP ERROR VECTOR
744 001266 012767 177570 177722 MOV #DSWR,SWR ;:SETUP FOR A HARDWARE SWICH REGISTER
745 001274 012767 177570 177716 MOV #DDISP,DISPLAY ;:AND A HARDWARE DISPLAY REGISTER
746 001302 022777 177777 177706 CMP #-1,@SWR ;:TRY TO REFERENCE HARDWARE SWR
747 001310 001012 BNE 66$ ;:BRANCH IF NO TIMEOUT TRAP OCCURRED
748 ;:AND THE HARDWARE SWR IS NOT - -1
749 001312 000403 BR 65$ ;:BRANCH IF NO TIMEOUT
750 001314 012716 001322 64$: MOV #65$,(SP) ;:SET UP FOR TRAP RETURN
751 001320 000002 RTI
752 001322 012767 000176 177666 65$: MOV #SWREG,SWR ;:POINT TO SOFTWARE SWR
753 001330 012767 000174 177662 MOV #DISPREG,DISPLAY
754 001336 012637 000004 66$: MOV (SP)+,@#ERRVEC ;:RESTORE ERROR VECTOR
755
756 001342 023737 000042 000046 CMP @#42,@#46
757 001350 001433 BEQ 3$ ;YES, SKIP TITLE
758 .SBTTL TYPE PROGRAM NAME
759 ;:TYPE THE NAME OF THE PROGRAM IF FIRST PASS
760 001352 005227 177777 INC #-1 ;:FIRST TIME?
761 001356 001027 BNE 67$ ;:BRANCH IF NO
762 001360 104401 001416 TYPE ,68$ ;:TYPE ASCIZ STRING
763 .SBTTL GET VALUE FOR SOFTWARE SWITCH REGISTER
764 001364 005737 000042 TST @#42 ;:ARE WE RUNNING UNDER XXDP/ACT?
765 001370 001006 BNE 69$ ;:BRANCH IF YES
766 001372 026727 177620 000176 CMP SWR,#SWREG ;:SOFTWARE SWITCH REG SELECTED?
767 001400 001005 BNE 70$ ;:BRANCH IF NO
768 001402 104405 GTSWR ;:GET SOFT-SWR SETTINGS
769 001404 000403 BR 70$
770 001406 112767 000001 005144 69$: MOV#B #1,$AUTOB ;:SET AUTO-MODE INDICATOR
771 001414 70$:
772 001414 000410 BR 67$ ;:GET OVER THE ASCIZ
773 ;:68$: .ASCIZ <CRLF>#MD-11-DZDRC-G#<CRLF>
774 001436 67$:
775 001436 000463 BR 8$ ;SKIP INTERRUPT VECTOR SIZER IF
776 ;:IN AUTOMATIC MODE
777 ;:SIZE FOR INTERRUPT VECTOR IF IN AUTOMATIC MODE.
778
779 001440 012700 000302 3$: MOV #302,R0 ;SET UP FLOATING VECTOR AREA
780 001444 010060 177776 4$: MOV R0,-2(R0)
781 001450 012720 000003 MOV #3,(R0)+
782 001454 005720 TST (R0)+
783 001456 022700 000776 CMP #776,R0
784 001462 100370 BPL 4$
785 001464 012737 001516 000014 MOV #5$,@#14 ;SET UP BPT
786 001472 012737 000340 000016 MOV #340,@#16 ;SET UP BPT PSW
787 001500 012777 000101 177472 MOV #101,@RCSR ;FORCE AN INTERRUPT
788 001506 000240 NOP
789 001510 000240 NOP
790 001512 000000 HALT ;NO INTERRUPT OCCURRED

```



```

791 001514 000423
792 001516 162716 000004
793 001522 011637 001210
794 001526 013737 001210 001212
795 001534 062737 000002 001212
796 001542 013737 001210 001214
797 001550 062737 000004 001214
798 001556 012737 000016 000014
799 001564 012700 000302
800 001570 010060 177776
801 001574 005020
802 001576 005720
803 001600 022700 000776
804 001604 100371
805 001606 004767 000020
806 001612 000137 001726
807 001616 012706 001200
808 001622 004767 000004
809 001626 000167 003360
810 001632 013746 000004
811 001636 012737 001712 000004
812 001644 012737 000031 177060
813 001652 012637 000004
814 001656 012737 177777 001240
815 001664 012701 160000
816 001670 004737 005246
817 001674 012701 000770
818 001700 004737 005276
819 001704 104401
820 001706 006640
821 001710 000207
822 001712 022626
823 001714 012637 000004
824 001720 005037 001240
825 001724 000207
826
827 001726 012700 001200
828 001732 012701 001222
829 001736 012021
830 001740 012021
831 001742 012021
832 001744 012021
833 001746 012021
834 001750 012021
835 001752 012021
836 001754 012706 001200
837 001760 012767 002010 003004
838 001766 005037 004770
839
840
841 001772 104406
842 001774 032777 001000 177214
843 002002 001002
844 002004 004737 005010
845 002010 016705 177206
846 002014 012777 000240 175754

BR 6$
5$: SUB #4,(SP) ;FIGURE OUT INTERRUPT VECTOR
MOV (SP),@#RCSR1 ;PUT IT IN TABLE
MOV @#RCSR1,@#RCSR1+2
ADD #2,@#RCSR1+2 ;FILL IN NEXT TABLE ENTRY
MOV @#RCSR1,@#RCSR1+4
ADD #4,@#RCSR1+4 ;FILL IN LAST TABLE ENTRY
MOV #16,@#14 ;RESTORE BPT
6$: MOV #302,R0 ;SET UP TRAP CATCHER
7$: MOV R0,-2(R0)
CLR (R0)+
TST (R0)+
CMP #776,R0
BPL 7$
8$: JSR PC,FIRST
JMP @#START
SPEC: MOV #STACK,%6
JSR PC,FIRST
JMP SPEC0
FIRST: MOV @#4,-(%6)
MOV #XORA,@#4
MOV #31,@#177060
MOV (%6)+,@#4
MOV #-1,@#XORFLG
MOV #160000,R1
JSR PC,@#SPEC1
MOV #770,R1
JSR PC,@#SPEC2
TYPE
MESS1
XORA: RTS PC
CMP (%6)+,(%6)+
MOV (%6)+,@#4
CLR @#XORFLG
RTS PC
;INITIALIZE ADDRESS AND VECTORS
START: MOV #RCSR,R0 ;GET ADDRESS OF FIRST POSSIBLE DR11-C'S
MOV #DRCSR,R1
MOV (R0)+,(R1)+ ;LOAD INITIAL TEST ADDRESSES
MOV (R0)+,(R1)+
MOV (R0)+,(R1)+
MOV (R0)+,(R1)+
MOV (R0)+,(R1)+
MOV (R0)+,(R1)+
MOV (R0)+,(R1)+
RSTART: MOV #STACK,%6 ;SET UP STACK
MOV #BEGIN,RETURN ;SET SCOPE RETURN
CLR @#SCOPEF

;DOES RESET CLEAR REGISTER?
CKSWR
BIT #SW9,@SWR
BNE BEGIN
BEGIN: JSR PC,@#MOREID
MOV DRCSR,R5 ;GET ADDRESS OF STATUS REGISTER
MOV #240,@PSW ;SET PRIORITY LEVEL 6

```

```

847 002022 012737 002060 000004      MOV      #1$,@#4      ;SET TIME OUT TRAP VECTOR
848 002030 012767 000010 002730      MOV      #10,ICOUNT
849 002036 012777 177777 177160      MOV      #-1,@DROBUF ;PRESET OUTPUT BUFFER
850 002044 000005      RESET    ;CLEAR DATA REGISTER
851 002046 017700 177152      MOV      @DROBUF,%0  ;GET RESULT OF RESET
852 002052 001403      BEQ      2$
853 002054 104000      HLT
854 002056 000401      BR       2$          ;DATA REGISTER NOT CLEAR
855 002060 104000      1$:      HLT          ;ERROR! TIMED OUT WHEN REFERENCING DROBUF.
856 002062 012706 001200      2$:      MOV      #STACK,SP ;RESET STACK POINTER
857 002066 012737 000006 000004      MOV      #6,@#4      ;RESTORE TIME OUT TRAP
858
859 002074 000004      SCOPE
860 002076 012767 004000 002662      MOV      #4000,ICOUNT
861 002104 012777 177777 177112      MOV      #-1,@DROBUF ;ALL ONES TO REGISTER
862 002112 017700 177106      MOV      @DROBUF,%0
863 002116 022700 177777      CMP      #-1,%0
864 002122 001401      BEQ      .+4
865 002124 104000      HLT          ;REG WILL NOT HOLD ONES
866
867 002126 000004      SCOPE
868 002130 012767 000010 002630      MOV      #10,ICOUNT
869 002136 012777 177777 177060      MOV      #-1,@DROBUF
870 002144 000005      RESET    ;SET DATA TO ALL ONES
871 002146 005777 177054      TST      @DRIBUF    ;SHOULD CLEAR REGISTER
872 002152 001401      BEQ      .+4
873 002154 104000      HLT          ;REG FAILED TO CLEAR
874
875 002156 000004      SCOPE
876 002160 012767 004000 002600      MOV      #4000,ICOUNT
877 002166 012777 052525 177030      MOV      #52525,@DROBUF
878 002174 017700 177024      MOV      @DROBUF,%0
879 002200 022700 052525      CMP      #52525,%0
880 002204 001401      BEQ      .+4
881 002206 104000      HLT          ;DATA NOT=52525
882
883 002210 000004      SCOPE
884 002212 012777 125252 177004      MOV      #125252,@DROBUF
885 002220 017700 177000      MOV      @DROBUF,%0
886 002224 022700 125252      CMP      #125252,%0
887 002230 001401      BEQ      .+4
888 002232 104000      HLT          ;DATA NOT=125252
889
890      ;TEST RELIABILITY OF DR11-C OUTPUT BUFFER REGISTER
891 002234 000004      SCOPE
892 002236 012737 000040 004766      MOV      #40,@#ICOUNT
893 002244 010502      BUFTST: MOV      R5,R2      ;GET ADDRESS OF DRCSR
894 002246 005722      TST      (R2)+      ;R2=ADDRESS OF OUTPUT BUFFER REG.
895 002250 012703 0004C1      MOV      #401,R3    ;LOAD CONSTANT
896 002254 012704 000400      1$:      MOV      #256.,R4   ;SET COUNT
897 002260 005000      CLR      R0        ;PRESET EXPECTED RESULT
898 002262 005012      CLR      (R2)      ;CLEAR REGISTER
899 002264 060300      2$:      ADD      R3,R0
900 002266 060312      ADD      R3,(R2)
901 002270 021200      CMP      (R2),R0
902 002272 001401      BEQ      .+4

```

```

903 002274 104000          HLT
904 002276 005304          DEC      R4
905 002300 001371          BNE     2$
906 002302 006303          ASL     R3
907 002304 001363          BNE     1$
908
909          ;TEST THAT BYTE REFERENCE TO DROBUF AFFECT PROPER BYTE ONLY
910
911 002306 000004          SCOPE
912 002310 012777 177777 176706 TAG:  MOV     #-1,@DROBUF
913 002316 105077 176702          CLRB   @DROBUF          ;CLEAR LOW BYTE
914 002322 017700 176676          MOV     @DROBUF,%0
915 002326 022700 177400          CMP     #177400,%0
916 002332 001401          BEQ
917 002334 104000          HLT          ;BYTE LOW FAILED TO CLEAR
918
919 002336 000004          SCOPE
920 002340 012777 177777 176656          MOV     #-1,@DROBUF
921 002346 105077 176656          CLRB   @DRBHIO          ;CLEAR HIGH BYTE
922 002352 017700 176646          MOV     @DROBUF,%0
923 002356 022700 000377          CMP     #377,%0
924 002362 001401          BEQ
925 002364 104000          HLT          ;HIGH BYTE CLEAR FAILED
926
927 002366 000004          SCOPE
928 002370 005037 002440          CLR     @#2$
929 002374 012704 002440          MOV     #2$,R4
930 002400 005077 176620          CLR     @DROBUF
931 002404 105077 176620          CLRB   @DRBHIO
932 002410 105277 176614          1$:  INCB   @DRBHIO          ;INCREMENT HIGH BYTE
933 002414 105264 000001          INCB   1(R4)
934 002420 027714 176600          CMP     @DROBUF,(R4)
935 002424 001401          BEQ     .+4
936 002426 104000          HLT          ;HIGH BYTE HAS BAD DATA
937 002430 105764 000001          TSTB   1(R4)
938 002434 001402          BEQ     3$
939 002436 000764          BR      1$
940 002440 000000          2$:  .WORD 0
941 002442 000004          3$:  SCOPE
942          ;CONTROL STATUS REGISTER (DRCSR) TESTS.
943 002444 005015          CLR     (R5)
944 002446 011500          MOV     (R5),R0
945 002450 001401          BEQ     .+4
946 002452 104000          HLT
947 002454 012715 000140          MOV     #140,@R5          ;INTERRUPT ENABLE FOR A+B
948 002460 011500          MOV     @R5,%0
949 002462 022700 000140          CMP     #140,%0          ;ENABLE BITS
950 002466 001401          BEQ     .+4
951 002470 104000          HLT
952
953 002472 000004          SCOPE
954 002474 012767 000010 002264          MOV     #10,ICOUNT
955 002502 012715 000140          MOV     #140,@R5          ;SET INTERRUPT ENABLE FLOPS
956 002506 000005          RESET          ;CLEAR THOSE FLOPS
957 002510 011500          MOV     @R5,%0
958 002512 001401          BEQ     .+4

```

959	002514	104000			HLT		;RESET DID NOT CLEAR INTERRUPT ENABLE BITS
960							
961	002516	000004			SCOPE		
962	002520	052715	000001		BIS	#1,@R5	;SHOULD SET REQ A ALSO
963	002524	021527	000201		CMP	@R5,#201	.
964	002530	001401			BEQ	+.4	
965	002532	104000			HLT		
966	002534	005015			CLR	@R5	
967							
968	002536	000004			SCOPE		
969	002540	052715	000002		BIS	#2,@R5	;SHOULD SET REQ B
970	002544	021527	100002		CMP	@R5,#100002	.
971	002550	001401			BEQ	+.4	
972	002552	104000			HLT		
973	002554	005015			CLR	@R5	
974							
975	002556	000004			SCOPE		
976	002560	052737	000340	177776	BIS	#340,@PSW	
977	002566	052715	177777		BIS	#-1,@R5	
978	002572	022715	100343		CMP	#100343,(R5)	
979	002576	001401			BEQ	+.4	
980	002600	104000			HLT		
981	002602	042715	000003		BIC	#3,@R5	
982	002606	022715	000140		CMP	#140,@R5	
983	002612	001401			BEQ	+.4	
984	002614	104000			HLT		;WRONG BITS SET
985							
986	002616	000004			SCOPE		
987	002620	012737	000340	177776	MOV	#340,@PSW	
988	002626	052715	000003		BIS	#3,@R5	
989	002632	000005			RESET		
990	002634	005715			TST	@R5	
991	002636	001401			BEQ	+.4	
992	002640	104000			HLT		;RESET DID NOT CLEAR
993							
994	002642	000004			SCOPE		
995	002644	012767	004000	002114	MOV	#4000,ICOUNT	
996	002652	005015			CLR	@R5	
997	002654	005215			INC	@R5	
998	002656	105715			TSTB	@R5	
999	002660	100401			BMI	+.4	
1000	002662	104000			HLT		;BIT 0 DID NOT SET BIT 7
1001							
1002	002664	000004			SCOPE		
1003	002666	012715	000002		MOV	#2,@R5	
1004	002672	005715			TST	@R5	
1005	002674	100401			BMI	+.4	
1006	002676	104000			HLT		;BIT 1 DID NOT SET BIT 15
1007							
1008							;TEST EXTERNAL TRANSFERS - CABLE MUST BE CONNECTED
1009	002700	000004			SCOPE		
1010	002702	005077	176316		CLR	@DROBUF	
1011	002706	017777	176314	176310	MOV	@DRIBUF,@DROBUF	;TEST TRANSFER OF ZERO
1012	002714	001401			BEQ	+.4	
1013	002716	104000			HLT		;NOT EQUAL TO ZERO
1014							

```

1015 002720 000004          SCOPE
1016 002722 012777 177777 176274    MOV    #-1,@DROBUF
1017 002730 017777 176272 176266    MOV    @DRIBUF,@DROBUF ;MOV ALL ONES
1018 002736 022777 177777 176260    CMP    #-1,@DROBUF
1019 002744 001401          BEQ    .+4
1020 002746 104000          HLT                    ;NOT ALL ONES
1021
1022 002750 000004          SCOPE
1023 002752 005067 002010          CLR    ICOUNT
1024 002756 005000          CLR    %0
1025 002760 010077 176240          TST6: MOV    %0,@DROBUF ;TEST ALL NUMBERS
1026 002764 017777 176236 176232    MOV    @DRIBUF,@DROBUF
1027 002772 020077 176226          CMP    %0,@DROBUF
1028 002776 001401          BEQ    .+4
1029 003000 104000          HLT                    ;ERROR - CHECK %0 FOR GOOD
1030 003002 005200          INC    %0 ;DROBUF FOR BAD
1031 003004 001403          BEQ    TST9
1032 003006 005077 176212          CLR    @DROBUF
1033 003012 000762          BR     TST6
1034 003014 000004          TST9: SCOPE
1035 003016 012737 000005 004766    MOV    #5,@ICOUNT
1036                                ;TEST DATA FROM BLACK BOX (NOT CONNECTED)
1037 003024 012777 177777 176172    MOV    #-1,@DROBUF
1038 003032 017777 176166 176166    MOV    @DROBUF,@DRIBUF ;STATIC LINES EQUAL ONES
1039 003040 017700 176162          MOV    @DRIBUF,%0 ;DATA REGISTER TO %0
1040 003044 022700 177777          CMP    #-1,%0
1041 003050 001401          BEQ    .+4
1042 003052 104000          HLT                    ;REG 0 SHOULD ALL ONES
1043
1044                                ;READY BIT IS IN A ONE STATE
1045 003054 000004          SCOPE
1046 003056 012715 000003          MOV    #3,@R5 ;CSRO AND CSR1
1047 003062 011500          MOV    (R5),R0
1048 003064 022700 100203          CMP    #100203,R0
1049 003070 001401          BEQ    .+4
1050 003072 104000          HLT
1051
1052                                ;CAN WE RAISE INTERUPT 'A'
1053 003074 000004          SCOPE
1054 003076 052737 000340 177776    BIS    #340,@PSW ;LOCK OUT INTERRUPTS
1055 003104 012706 001200          MOV    #STACK,%6
1056 003110 012777 003132 176114    MOV    #TST4,@DRVECA ;INTERRUPT RETURN POINTER
1057 003116 012715 000101          MOV    #101,@R5 ;INTERRUPT ENABLE AND CSRO
1058 003122 005037 177776          CLR    @PSW
1059 003126 000240          NOP
1060 003130 104000          HLT                    ;NO 'A' INTERRUPT
1061 003132 005015          TST4: CLR    @R5
1062 003134 016777 176074 176070    MOV    DRLVL,@DRVECA ;MOVE .+2 TO 'A' INTERRUPT VECTOR
1063
1064                                ;RAISE INTERRUPT 'B'
1065 003142 000004          SCOPE
1066 003144 012706 001200          MOV    #STACK,%6
1067 003150 052737 000340 177776    BIC    #340,@PSW
1068 003156 012777 003202 176052    MOV    #TST5,@DRVECB
1069 003164 012715 000042          MOV    #42,@R5 ;IE AND CSR1
1070 003170 042737 000377 177776    BIC    #377,@PSW

```

```

1071 003176 000240          NOP
1072 003200 104000          HLT
1073 003202 005015          TST5: CLR @R5 ;NO B INTERRUPT
1074
1075          ;TEST FOR INTERRUPT FROM DEVICE
1076 003204 016777 176034 176022 MOV PL,@DRLVL
1077 003212 042737 000340 177776 BIC #340,@#PSW ;PROCESSOR LEVEL ZERO
1078 003220 012777 003252 176004 MOV #TINT1,@DRVECA
1079 003226 012706 001200 MOV #STACK,%6 ;STACK POINTER
1080 003232 042777 000100 175762 BIC #100,@DRCSR ;CLEAR INTERRUPT ENABLE
1081 003240 052777 000101 175754 BIS #101,@DRCSR ;SET INTERRUPT ENABLE-AND CSRO
1082 003246 000240          NOP
1083 003250 104000          HLT
1084 003252 000004          TINT1: SCOPE ;NO DEVICE INTERRUPT OCCURED
1085
1086          ;TEST FOR INTERRUPT FROM THE DEVICE
1087 003254 042737 000340 177776 BIC #340,@#PSW
1088 003262 052737 000040 177776 BIS #040,@#PSW ;SET TO PRIORITY LEVEL 1
1089 003270 012777 003322 175734 MOV #TINT2,@DRVECA ;INTERRUPT VECTOR ADDRESS
1090 003276 012706 001200 MOV #STACK,%6 ;SET UP STACK POINTER
1091 003302 042777 000100 175712 BIC #100,@DRCSR ;CLEAR INTERRUPT ENABLE
1092 003310 052777 000101 175704 BIS #101,@DRCSR ;SET INTERRUPT ENABLE-AND CSRO
1093 003316 000240          NOP
1094 003320 104000          HLT ;NO DEVICE INTERRUPT OCCURED
1095
1096 003322 000004          TINT2: SCOPE
1097 003324 042737 000340 177776 BIC #340,@#PSW
1098 003332 052737 000100 177776 BIS #100,@#PSW ;SET TO PRIORITY LEVEL 2
1099 003340 012777 003372 175664 MOV #TINT3,@DRVECA ;INTERRUPT VECTOR ADDRESS
1100 003346 012706 001200 MOV #STACK,%6 ;SET UP STACK POINTER
1101 003352 042777 000100 175642 BIC #100,@DRCSR ;CLEAR INTERRUPT ENABLE
1102 003360 052777 000101 175634 BIS #101,@DRCSR ;SET INTERRUPT ENABLE-AND CSRO
1103 003366 000240          NOP
1104 003370 104000          HLT ;NO DEVICE INTERRUPT OCCURED
1105
1106 003372 000004          TINT3: SCOPE
1107          ;TEST FOR INTERRUPT FROM THE DEVICE
1108 003374 042737 000340 177776 BIC #340,@#PSW
1109 003402 052737 000140 177776 BIS #140,@#PSW ;SET TO PRIORITY LEVEL 3
1110 003410 012777 003442 175614 MOV #TINT4,@DRVECA ;INTERRUPT VECTOR ADDRESS
1111 003416 012706 001200 MOV #STACK,%6 ;SET UP STACK POINTER
1112 003422 042777 000100 175572 BIC #100,@DRCSR ;CLEAR INTERRUPT ENABLE
1113 003430 052777 000101 175564 BIS #101,@DRCSR ;SET INTERRUPT ENABLE-AND CSRO
1114 003436 000240          NOP
1115 003440 104000          HLT ;NO DEVICE INTERRUPT OCCURED
1116 003442 000004          TINT4: SCOPE
1117
1118          ;TEST FOR INTERRUPT FROM DEVICE
1119 003444 042737 000340 177776 BIC #340,@#PSW
1120 003452 052737 000200 177776 BIS #200,@#PSW ;RAISE PROCESSOR PRIORITY TO LEVEL 4
1121 003460 012777 003522 175544 MOV #TINT5,@DRVECA ;IN CASE OF INTERRUPT
1122 003466 012706 001200 MOV #STACK,%6 ;SET STACK POINTER
1123 003472 042777 000100 175522 BIC #100,@DRCSR ;CLEAR INTERRUPT ENABLE
1124 003500 052777 000101 175514 BIS #101,@DRCSR ;SET INTERRUPT ENABLE AND CSRO
1125 003506 000240          NOP ;LET INTERRUPT OCCUR
1126 003510 042777 000100 175504 BIC #100,@DRCSR

```

```

1127 003516 000240          NOP
1128 003520 104000          HLT                    ;NO DEVICE INTERRUPT OCCURED
1129 003522 000004          TINT5: SCOPE
1130
1131                          ;TEST FOR NO INTERRUPT FROM DEVICE (HIGHEST PROCESSOR PRIORITY)
1132 003524 052737 000340 177776  BIS #340,@#PSW          ;RAISE PROCESSOR PRIORITY TO HIGHEST LEVEL
1133 003532 012777 003572 175472  MOV #TINT6,@DRVECA     ;IN CASE OF INTERRUPT
1134 003540 012706 001200          MOV #STACK,%6          ;SET STACK POINTER
1135 003544 042777 000100 175450  BIC #100,@DRCSR        ;CLEAR INTERRUPT ENABLE
1136 003552 052777 000101 175442  BIS #101,@DRCSR
1137 003560 000240          NOP
1138 003562 042777 000100 175432  BIC #100,@DRCSR
1139 003570 000401          BR .+4                  ;WITH NO INTERRUPT, BRANCH OVER HALT
1140 003572 104000          TINT6: HLT              ;INTERRUPT OCCURED
1141 003574 000004          SCOPE
1142
1143                          ;TEST FOR NO INTERRUPT FROM DEVICE
1144 003576 042737 000340 177776  BIC #340,@#PSW
1145 003604 052737 000240 177776  BIS #240,@#PSW          ;RAISE PROCESSOR PRIORITY TO LEVEL 5
1146 003612 012777 003652 175412  MOV #TINT7,@DRVECA     ;IN CASE OF INTERRUPT
1147 003620 012706 001200          MOV #STACK,%6          ;SET STACK POINTER
1148 003624 042777 000100 175370  BIC #100,@DRCSR        ;CLEAR INTERRUPT ENABLE
1149 003632 052777 000101 175362  BIS #101,@DRCSR        ;SET INTERRUPT ENABLE AND CSRO
1150 003640 000240          NOP
1151 003642 042777 000100 175352  BIC #100,@DRCSR        ;DON'T LEAVE IT SET
1152 003650 000401          BR .+4                  ;WITH NO INTERRUPT, BRANCH OVER HALT
1153 003652 104000          TINT7: HLT              ;INTERRUPT OCCURED
1154 003654 000004          SCOPE
1155
1156                          ;TEST FOR NO INTERRUPT FROM DEVICE
1157 003656 042737 000340 177776  BIC #340,@#PSW
1158 003664 052737 000300 177776  BIS #300,@#PSW          ;RAISE PROCESSOR PRIORITY TO LEVEL 6
1159 003672 012777 003732 175332  MOV #TINT8,@DRVECA     ;IN CASE OF INTERRUPT
1160 003700 012706 001200          MOV #STACK,%6          ;SET STACK POINTER
1161 003704 042777 000100 175310  BIC #100,@DRCSR        ;CLEAR INTERRUPT ENABLE
1162 003712 052777 000101 175302  BIS #101,@DRCSR        ;SET INTERRUPT ENABLE-AND CSRO
1163 003720 042777 000100 175274  BIC #100,@DRCSR        ;DON'T LEAVE IT SET
1164 003726 000240          NOP
1165 003730 000401          BR .+4                  ;WITH NO INTERRUPT, BRANCH OVER HALT
1166 003732 104000          TINT8: HLT              ;INTERRUPT OCCURED
1167 003734 000004          SCOPE
1168
1169 003736 016777 175272 175266  MOV DRLVL,@DRVECA      ;FOR FALSE INTERRUPT
1170 003744 005077 175262          CLR @DRVECA
1171
1172                          ;END OF TEST ROUTINE
1173 003750 012777 000052 000356  END: MOV #'*,@TDBR      ;TYPE '*'
1174 003756 105777 000354          1$: TSTB @TCSR
1175 003762 100375          BPL 1$
1176 003764 005077 000344          CLR @TDBR
1177 003770 105777 000342          2$: TSTB @TCSR
1178 003774 100375          BPL 2$
1179 003776 104406          CKSWR
1180 004000 032777 002000 175210  BIT #SW10,@SWR         ;LOOP ON SELECTED DR?
1181 004006 001402          BEQ 4$
1182 004010 000137 001754          JMP @#RSTART          ;REPEAT TEST ON DR11C SELECTED

```

```
1183 ;STEP TO NEXT DR11-C
1184 004014 012700 000010 4$: MOV #10,R0 ;STEPPING CONSTANT
1185 004020 012737 004072 000004 MOV #5$,@#4 ;SET TIME OUT TRAP
1186 004026 160005 SUB R0,R5 ;STEP TO NEXT DR11-C ADDRESS
1187 004030 005715 TST (R5) ;WILL TIME OUT IF NOT AVAILABLE
1188 004032 012705 001222 MOV #DRCSR,R5 ;SET TABLE POINTER
1189 004036 160025 SUB R0,(R5)+
1190 004040 160025 SUB R0,(R5)+
1191 004042 160025 SUB R0,(R5)+
1192 004044 160025 SUB R0,(R5)+
1193 004046 060025 ADD R0,(R5)+
1194 004050 060025 ADD R0,(R5)+
1195 004052 060025 ADD R0,(R5)+
1196 004054 000137 001754 JMP @#RSTART ;RESTART TEST USING NEXT DR11-C
1197 004060 104406 CKSWR
1198 004062 032777 001000 175126 BIT #SW9,@SWR
1199 004070 001004 BNE 8$
1200 004072 104401 5$: TYPE ;TYPE 'END PASS'
1201 004074 006675 MESS3
1202 004076 005267 175144 INC PASCNT ;KEEP TRACK OF PASSES COMPLETED
1203 004102 013700 000042 8$: MOV @#42,RC
1204 004106 001405 BEQ END1
1205 004110 000005 RESET
1206 004112 004710 $ENDAD: JSR PC,(R0)
1207 004114 000240 NOP
1208 004116 000240 NOP
1209 004120 000240 NOP
1210 004122 000137 001726 END1: JMP @#START
1211
1212 ;ENTERED WITH SYSTEM TRAP CALL(HLT)
1213 ;PRINT OUT THE ERROR PC AND STATUS REGISTER
1214 004126 104406 .HLT: CKSWR
1215 004130 037727 175062 020000 BIT @SWR,#SW13 ;TEST FOR INHIBIT PRINT OUT
1216 004136 001401 BEQ .+4 ;BRANCH TO PRINT
1217 004140 000002 RTI ;INHIBIT, RETURN TO MAIN STREAM
1218 004142 012667 000172 MOV (6)+,SAVPC ;PC OF FAILING ROUTINE
1219 004146 012667 000170 MOV (6)+,SAVCC ;CC OF ERROR CONDITION
1220 004152 024646 CMP -(6),-(6) ;REPOSITION THE STACK
1221 004154 105777 000156 TSTB @TCSR ;WAIT FOR FLAG
1222 004160 100375 BPL .-4 ;IF NOT UP.
1223 004162 012777 000215 000144 MOV #215,@TDBR ;CR
1224 004170 105777 000142 TSTB @TCSR
1225 004174 100375 BPL .-4
1226 004176 012777 000212 000130 MOV #212,@TDBR ;LINE FEED
1227 004204 105777 000126 TSTB @TCSR
1228 004210 100375 BPL .-4
1229 004212 010267 000110 MOV %2,SAVR2 ;SAVE R2
1230 004216 010367 000106 MOV %3,SAVR3 ;SAVE R3
1231 004222 010467 000104 MOV %4,SAVR4 ;SAVE R4
1232 004226 016702 000106 MOV SAVPC,%2
1233 004232 004767 000106 JSR %7,PRTAB ;PRINT OCTAL NUMBER
1234 004236 012777 000240 000070 MOV #240,@TDBR
1235 004244 105777 000066 TSTB @TCSR ;SPACE BETWEEN WORDS
1236 004250 100375 BPL .-4
1237 004252 016702 000064 MOV SAVCC,%2
1238 004256 004767 000062 JSR %7,PRTAB ;PRINT OCTAL NUMBER
```


1239	004262	004767	000522		JSR	%7,MOREID	:DEVICE ADDRESS AND VECTORS
1240	004266	016702	000034		MOV	SAVR2,%2	:RESTORE REGISTERS
1241	004272	016703	000032		MOV	SAVR3,%3	
1242	004276	016704	000030		MOV	SAVR4,%4	
1243	004302	023737	000042	000046	CMP	@#42,@#46	:ARE WE IN ACT-11 AUTO MODE?
1244	004310	001404			BEQ	1\$:HALT ON ERROR IF YES
1245	004312	104406			CKSWR		
1246	004314	005777	174676		TST	@SWR	:TEST FOR HALT SWITCH
1247	004320	100001			BPL	.+4	
1248	004322	000000		1\$:	HALT		:HALT ON ERROR SET
1249	004324	000002			RTI		:RETURN TO MAIN STREAM
1250	004326	000000			SAVR2:	0	
1251	004330	000000			SAVR3:	0	
1252	004332	000000			SAVR4:	0	
1253	004334	177566			TDBR:	177566	:DATA
1254	004336	177564			TCSR:	177564	:STATUS
1255	004340	000000			SAVPC:	0	
1256	004342	000000			SAVCC:	0	
1257							
1258	004344	005067	000252		PRTAB:	CLR BINCT	
1259	004350	005067	000244			CLR WGTCT	
1260	004354	012704	004626		MOV	#LIST,%4	:GET LIST ADDRESS
1261	004360	012767	000005	000236	MOV	#5,ASCNT	
1262	004366	012767	000007	000220	MOV	#7,SEVEN	
1263	004374	012767	000001	000214	MOV	#1,DECML	
1264	004402	105777	177730		WAIT1:	TSTB @TCSR	
1265	004406	100375			BPL	WAIT1	
1266	004410	005702			TST	%2	
1267	004412	100404			BMI	MINUS	:NEG SIGN PRINT 1
1268	004414	012777	000260	177712	MOV	#260,@TDBR	:POS SIGN PRINT 0
1269	004422	000403			BR	STAR	
1270	004424	012777	000261	177702	MINUS:	MOV #261,@TDBR	
1271	004432	016703	000156		STAR:	MOV SEVEN,%3	:PUT MASK IN R3
1272	004436	010267	000150		MOV	%2,TOODLE	:GET READY TO DOODLE NUMBER IN TOODLE
1273	004442	005167	000144		COM	TOODLE	:COMPENSATES FOR COMPLEMENT DURING BIC
1274	004446	046703	000140		BIC	TOODLE,%3	:AND IN OCTAL CHARACTER
1275	004452	001410			BEQ	WRTSC	:ZERO, WRITE 0 IN LIST
1276	004454	066767	000136	000136	MKNUM:	ADD DECML,WGTCT	:COUNT UP TO
1277	004462	005267	000134		INC	BINCT	:AND RECORD
1278	004466	026703	000126		CMP	WGTCT,%3	:SAME BINARY WEIGHT
1279	004472	001370			BNE	MKNUM	:KEEP COUNTN
1280	004474	062767	000260	000120	WRTOC:	ADD #260,BINCT	:ADD ASCII PREFIX
1281	004502	016724	000114		MOV	BINCT,(4)+	:WRITE ASCII CHAR IN LIST
1282	004506	066767	000102	000102	ADD	SEVEN,DECML	:EXPAND BINARY WEIGHT
1283	004514	005067	000100		CLR	WGTCT	
1284	004520	005067	000076		CLR	BINCT	
1285	004524	005367	000074		DEC	ASCNT	
1286	004530	001410			BEQ	XLIST	:5 CHAR IN LIST
1287	004532	012703	000003		MOV	#3,%3	:SET X3 FOR ADD LOOP
1288	004536	066767	000052	000050	MOADD:	ADD SEVEN,SEVEN	:MAKING SEVENTY BY SEVEN
1289	004544	005303			DEC	%3	
1290	004546	001373			BNE	MOADD	
1291	004550	000730			BR	STAR	:NX SEVEN SET GET NX OCTAL
1292	004552	012767	000005	000044	XLIST:	MOV #5,ASCNT	:SEND 5 CHAR TO TTY
1293	004560	105777	177552		WAIT2:	TSTB @TCSR	
1294	004564	100375			BPL	WAIT2	

```
1295 004566 014477 177542      MOV      -(4),@TDBR
1296 004572 005367 000026      DEC      ASCNT
1297 004576 001401                BEQ      HDFHM          ;FINISH PRINTING GET NXT NUM
1298 004600 000767                BR       WAIT2
1299 004602 105777 177530      HDFHM:  TSTB @TCSR
1300 004606 100375                BPL     :-4
1301 004610 000207                RTS     %7          ;HEAD FOR HOME
1302 004612 000000      TODDLE: 0
1303 004614 000000      SEVEN:  0
1304 004616 000000      DECML:  0
1305 004620 000000      WGTCT:  0
1306 004622 000000      BINCT:  0
1307 004624 000000      ASCNT:  0
1308 004626 000000      LIST:   0
1309 004630 000000                0
1310 004632 000000                0
1311 004634 000000                0
1312 004636 000000                0
1313                ;SCOPE LOOP ROUTINE ENTERED BY USER TRAP
1314 004640 022606      SCOPEB:  CMP      (6)+,%6          ;REPOSITION THE STACK
1315 004642 012637 177776      MOV      (6)+,@#PSW
1316 004646 000177 000120      JMP      @RETURN          ;SCOPE RETURN
1317
1318                ;SCOPE OR/AND ITERATION LOOP FOR EACH TEST 4000 TIMES
1319 004652 104406      .SCOPE:  CKSWR
1320 004654 032777 040000 174334      BIT      #SW14,@SWR          ;TEST SR FOR SCOPE
1321 004662 001366                BNE     SCOPEB          ;YES SCOPE
1322 004664 005737 001240      TST     @#XORFLG
1323 004670 100012                BPL     1$
1324 004672 013746 000004      MOV     @#4,-(%6)
1325 004676 012737 005000 000004      MOV     #XOR,@#4
1326 004704 012737 000031 177060      MOV     #31,@#177060
1327 004712 012637 000004      MOV     (%6)+,@#4
1328 004716 104406      1$:     CKSWR
1329 004720 032777 004000 174270      BIT     #SW11,@SWR          ;NO - TEST FOR ITERATION
1330 004726 001014                BNE     SCOPEA          ;INHIBIT ITERATION
1331 004730 005767 174312      TST     PASCNT          ;ARE WE IN FIRST PASS?
1332 004734 001411                BEQ     SCOPEA          ;BRANCH IF YES; INHIBIT ITERATIONS
1333 004736 026767 000026 000022      CMP     SCOPEF,ICOUNT
1334 004744 001403                BEQ     SCOPEG          ;EXIT - DONE
1335 004746 005267 000016      INC     SCOPEF          ;INCREMENT COUNT
1336 004752 000732                BR      SCOPEB          ;LOOP SOME MORE
1337 004754 005067 000010      SCOPEG: CLR     SCOPEF          ;CLEAR COUNT
1338 004760 011667 000006      SCOPEA: MOV     @%6,RETURN          ;SAVE SCOPE RETURN POINTER
1339 004764 000002                RTI          ;RETURN INLINE-NEXT TEST
1340 004766 004000      ICOUNT: 4000
1341 004770 000000      SCOPEF: 0          ;COUNT LOCATION FOR ITERATION LOOP
1342 004772 002010      RETURN:  BEGIN          ;ADDRESS OF LAST TEST
1343 004774 000167 173200      JMP     200
1344
1345 005000 022626      XOR:    CMP     (%6)+,(%6)+
1346 005002 012637 000004      MOV     (%6)+,@#4
1347 005006 000714                BR      SCOPEB
1348                ;FRINT DEVICE ADDRESS AND VECTOR
1349 005010 012777 000240 177316      MOREID: MOV     #240,@TDBR
1350 005016 105777 177314      TSTB   @TCSR
```

```

1351 005022 100375          BPL      -4
1352 005024 013702 001222    MOV      @WDRCSR,%2
1353 005030 004767 177310    JSR      %7,PRTAB
1354 005034 J12777 000240 177272    MOV      #240,@TDBR
1355 005042 105777 177270    TSTB    @TCSR
1356 005046 100375          BPL      -4
1357 005050 016702 174156    MOV      DRVFCA,%2
1358 005054 004767 177264    JSR      %7,PRTAB
1359 005060 012777 000215 177246    MOV      #215,@TDBR
1360 005066 105777 177244    TSTB    @TCSR
1361 005072 100375          BPL      -4
1362 005074 012777 000212 177232    MOV      #212,@TDBR
1363 005102 105777 177230    TSTB    @TCSR
1364 005106 100375          BPL      -4
1365 005110 005077 177220    CLR      @TDBR
1366 005114 105777 177216    TSTB    @TCSR
1367 005120 100375          BPL      -4
1368 005122 000207          RTS      %7          ;BACK TO PRINT
1369
1370          ;ENTER HERE FOR POWER FAIL
1371
1372 005124 010046          PFAIL:  MOV      %0,-(6)          ;SAVE REGISTER OR STACK
1373 005126 010146          MOV      %1,-(6)          ;WHEN POWERING DOWN
1374 005130 010246          MOV      %2,-(6)
1375 005132 010346          MOV      %3,-(6)
1376 005134 010446          MOV      %4,-(6)
1377 005136 010546          MOV      %5,-(6)
1378 005140 016746 172660    MOV      24,-(6)
1379 005144 010637 005160    MOV      %6,@SAVR6          ;STORE STACK POSITION
1380 005150 012737 005162 000024    MOV      #RESTAR,@#24
1381 005156 000000          HALT
1382 005160 000000          SAVR6:  0          ;HALT ON POWER DOWN NORMAL
1383 005162 016706 177772    RESTAR:MOV      SAVR6,%6          ;STACK IS SAVED HERE
1384 005166 012667 172632    MOV      (6)+,%24          ;RESTORE REGISTER OFF STACK
1385 005172 012605          MOV      (6)+,%5          ;WHEN POWERING UP
1386 005174 012604          MOV      (6)+,%4
1387 005176 012603          MOV      (6)+,%3
1388 005200 012602          MOV      (6)+,%2
1389 005202 012601          MOV      (6)+,%1
1390 005204 012600          MOV      (6)+,%0
1391 005206 000137 001754    JMP      @#RSTART
1392
1393          ;ENTER HERE FOR UNIQUE SELECTION OF DR11C
1394
1395 005212 000000          SPEC0:  HALT          ;PLACE ADDRESS OF DR11-C CONTROL STATUS
1396 005214 104406          CKSWR
1397 005216 017701 173774    MOV      @SWR,R1
1398 005222 004737 005246    JSR      PC,@#SPEC1
1399 005226 000000          HALT
1400 005230 104406          CKSWR
1401 005232 017701 173760    MOV      @SWR,R1
1402 005236 004737 005276    JSR      PC,@#SPEC2
1403 005242 000137 001726    JMP      @#START
1404
1405 005246 012700 001200          SPEC1:  MOV      #RCSR,R0          ;SET TABLE ADDRESS
1406 005252 010120          MOV      R1,(R0)+          ;LOAD INTO TABLE STARTING AT RCSR

```

```

1407 005254 062701 000002          ADD    #2,R1          ;STEP TO ADDRESS OF DROUTBUF
1408 005260 010120          MOV    R1,(R0)+      ;LOAD INTO TABLE
1409 005262 062701 000002          ADD    #2,R1          ;STEP TO ADDRESS OF DRINBUF
1410 005266 010120          MOV    R1,(R0)+      ;LOAD INTO TABLE
1411 005270 005301          DEC    R1            ;FORM ADDRESS OF DROUTBUF+1
1412 005272 010120          MOV    R1,(R0)+      ;LOAD INTO TABLE
1413 005274 000207          RTS    PC
1414
1415 005276 012700 001210          SPEC2: MOV    #RCSR1,R0
1416 005302 010120          MOV    R1,(R0)+      ;LOAD INTO TABLE
1417 005304 005721          TST    (R1)+
1418 005306 010120          MOV    R1,(R0)+
1419 005310 005721          TST    (R1)+
1420 005312 010120          MOV    R1,(R0)+
1421 005314 000207          RTS    PC
1422
1423          .LIST    ME
1424 005316          TYP:
1425          .SBTTL  TYPE ROUTINE
1426
1427          ;:*****
1428          ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
1429          ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
1430          ;*NOTE1:          $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
1431          ;*NOTE2:          $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
1432          ;*NOTE3:          $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
1433          ;*
1434          ;*CALL:
1435          ;*1) USING A TRAP INSTRUCTION
1436          ;*          TYPE    ,MESADR          ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
1437          ;*OR
1438          ;*          TYPE
1439          ;*          MESADR
1440          ;*
1441
1442 005316 105767 000223          $TYPE: TSTB    $TPFLG          ;;IS THERE A TERMINAL?
1443 005322 100002          BPL    1$            ;;BR IF YES
1444 005324 000000          HALT                    ;;HALT HERE IF NO TERMINAL
1445 005326 000407          BR     3$            ;;LEAVE
1446 005330 010046          1$:  MOV    R0,-(SP)      ;;SAVE R0
1447 005332 017600 000002          MOV    @2(SP),R0      ;;GET ADDRESS OF ASCIZ STRING
1448 005336 112046          2$:  MOVB  (R0)+,-(SP)   ;;PUSH CHARACTER TO BE TYPED ONTO STACK
1449 005340 001005          BNE    4$            ;;BR IF IT ISN'T THE TERMINATOR
1450 005342 005726          TST    (SP)+          ;;IF TERMINATOR POP IT OFF THE STACK
1451 005344 012600          60$: MOV    (SP)+,R0     ;;RESTORE R0
1452 005346 062716 000002          3$:  ADD    #2,(SP)      ;;ADJUST RETURN PC
1453 005352 000002          RTI                    ;;RETURN
1454 005354 122716 000011          4$:  CMPB  #HT,(SP)      ;;BRANCH IF <HT>
1455 005360 001430          BEQ    8$            ;;
1456 005362 122716 000200          CMPB  #CRLF,(SP)     ;;BRANCH IF NOT <CRLF>
1457 005366 001006          BNE    5$            ;;
1458 005370 005726          TST    (SP)+          ;;POP <CR><LF> EQUIV
1459 005372 104401          TYPE                    ;;TYPE A CR AND LF
1460 005374 005547          $CRLF
1461 005376 105067 000130          CLRB  $CHARCNT      ;;CLEAR CHARACTER COUNT
1462 005402 000755          BR     2$            ;;GET NEXT CHARACTER

```

```
1463 005404 004767 000056      5$:   JSR    PC,$TYPEC      ;;GO TYPE THIS CHARACTER
1464 005410 126726 000130      6$:   CMPB   $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
1465 005414 001350                BNE    2$                  ;;IF NO GO GET NEXT CHAR.
1466 005416 0167 000120          MOV    $NULL,-(SP)        ;;GET # OF FILLER CHARS. NEEDED
1467                                ;;AND THE NULL CHAR.
1468 005422 105366 000001      7$:   DECB   1(SP)          ;;DOES A NULL NEED TO BE TYPED?
1469 005426 002770                BLT    6$                  ;;BR IF NO--GO POP THE NULL OFF OF STACK
1470 005430 004767 000032      JSR    PC,$TYPEC          ;;GO TYPE A NULL
1471 005434 105367 000072      DECB   $CHARCNT          ;;DO NOT COUNT AS A COUNT
1472 005440 000770                BR     7$                  ;;LOOP
1473
1474                                ;HORIZONTAL TAB PROCESSOR
1475
1476 005442 112716 000040      8$:   MOVB   #' ,(SP)      ;;REPLACE TAB WITH SPACE
1477 005446 004767 000014      9$:   JSR    PC,$TYPEC      ;;TYPE A SPACE
1478 005452 132767 000007 000052 BITB   #7,$CHARCNT        ;;BRANCH IF NOT AT
1479 005460 001372                BNE    9$                  ;;TAB STOP
1480 005462 005726                TST   (SP)+                ;;POP SPACE OFF STACK
1481 005464 000724                BR     2$                  ;;GET NEXT CHARACTER
1482 005466 105777 000044      $TYPEC: TSTB  @ $TPS        ;;WAIT UNTIL PRINTER IS READY
1483 005472 100375                BPL   $TYPEC
1484 005474 116677 000002 000036 MOVB   2(SP),@ $TPB      ;;LOAD CHAR TO BE TYPED INTO DATA REG.
1485 005502 122766 000015 000002 CMPB   #CR,2(SP)        ;;IS CHARACTER A CARRIAGE RETURN?
1486 005510 001003                BNE    1$                  ;;BRANCH IF NO
1487 005512 105067 000014      CLRB   $CHARCNT          ;;YES--CLEAR CHARACTER COUNT
1488 005516 000406                BR     $TYPEX              ;;EXIT
1489 005520 122766 000012 000002 1$:   CMPB   #LF,2(SP)        ;;IS CHARACTER A LINE FEED?
1490 005526 001402                BEQ   $TYPEX              ;;BRANCH IF YES
1491 005530 105227                INCB  (PC)+                ;;COUNT THE CHARACTER
1492 005532 000000      $CHARCNT: .WORD 0        ;;CHARACTER COUNT STORAGE
1493 005534 000207      $TYPEX: RTS    PC
1494
1495 005536 177564      $TPS: .WORD 177564        ;;TTY PRINTER STATUS REG. ADDRESS
1496 005540 177566      $TPB: .WORD 177566        ;;TTY PRINTER BUFFER REG. ADDRESS
1497 005542 000                $NULL: .BYTE 0           ;;CONTAINS NULL CHARACTER FOR FILLS
1498 005543 002                $FILLS: .BYTE 2          ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
1499 005544 012                $FILLC: .BYTE 12         ;;INSERT FILL CHARS. AFTER A 'LINE FEED'
1500 005545 000                $TPFLG: .BYTE 0          ;;'TERMINAL AVAILABLE' FLAG (BIT<07>=0=YES)
1501 005546 077                $QUES: .ASCII '?'        ;;QUESTION MARK
1502 005547 015                $CRLF: .ASCII <15>       ;;CARRIAGE RETURN
1503 005550 000012      $LF: .ASCIZ <12>        ;;LINEFEED
1504                                .SBTTL BINARY TO OCTAL (ASCII) AND TYPE
1505
1506                                ;*****
1507                                ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
1508                                ;*OCTAL (ASCII) NUMBER AND TYPE IT.
1509                                ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
1510                                ;*CALL:
1511                                ;*   MOV    NUM,-(SP)      ;;NUMBER TO BE TYPED
1512                                ;*   TYPOS      ;;CALL FOR TYPEOUT
1513                                ;*   .BYTE  N          ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
1514                                ;*   .BYTE  M          ;;M=1 OR 0
1515                                ;*                                ;;1-TYPE LEADING ZEROS
1516                                ;*                                ;;0-SUPPRESS LEADING ZEROS
1517                                ;*
1518                                ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
```

```

1519      ;*$TYPOS OR $TYPOC
1520      ;*CALL:
1521      ;*      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
1522      ;*      TYPON      ;;CALL FOR TYPEOUT
1523      ;*
1524      ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
1525      ;*CALL:
1526      ;*      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
1527      ;*      TYPOC      ;;CALL FOR TYPEOUT
1528
1529 005552 017646 000000      $TYPOS: MOV      @ (SP),-(SP)      ;;PICKUP THE MODE
1530 005556 116667 000001 000211      MOV      1(SP), $OFILL      ;;LOAD ZERO FILL SWITCH
1531 005564 112667 000207      MOV      (SP)+, $OMODE+1      ;;NUMBER OF DIGITS TO TYPE
1532 005570 062716 000002      ADD      #2, (SP)      ;;ADJUST RETURN ADDRESS
1533 005574 000406      BR      $TYPON
1534 005576 112767 000001 000171      $TYPOC: MOV      #1, $OFILL      ;;SET THE ZERO FILL SWITCH
1535 005604 112767 000006 000165      MOV      #6, $OMODE+1      ;;SET FOR SIX(6) DIGITS
1536 005612 112767 000005 000154      $TYPON: MOV      #5, $OCNT      ;;SET THE ITERATION COUNT
1537 005620 010346      MOV      R3, -(SP)      ;;SAVE R3
1538 005622 010446      MOV      R4, -(SP)      ;;SAVE R4
1539 005624 010546      MOV      R5, -(SP)      ;;SAVE R5
1540 005626 116704 000145      MOV      $OMODE+1, R4      ;;GET THE NUMBER OF DIGITS TO TYPE
1541 005632 005404      NEG      R4
1542 005634 062704 000006      ADD      #6, R4      ;;SUBTRACT IT FOR MAX. ALLOWED
1543 005640 110467 000132      MOV      R4, $OMODE      ;;SAVE IT FOR USE
1544 005644 116704 000125      MOV      $OFILL, R4      ;;GET THE ZERO FILL SWITCH
1545 005650 016605 000012      MOV      12(SP), R5      ;;PICKUP THE INPUT NUMBER
1546 005654 005003      CLR      R3      ;;CLEAR THE OUTPUT WORD
1547 005656 006105      1$:      ROL      R5      ;;ROTATE MSB INTO 'C'
1548 005660 000404      BR      3$      ;;GO DO MSB
1549 005662 006105      2$:      ROL      R5      ;;FORM THIS DIGIT
1550 005664 006105      ROL      R5
1551 005666 006105      ROL      R5
1552 005670 010503      MOV      R5, R3
1553 005672 006103      3$:      ROL      R3      ;;GET LSB OF THIS DIGIT
1554 005674 105367 000076      DECB      $OMODE      ;;TYPE THIS DIGIT?
1555 005700 100016      BPL      7$      ;;BR IF NO
1556 005702 042703 177770      BIC      #177770, R3      ;;GET RID OF JUNK
1557 005706 001002      BNE      4$      ;;TEST FOR 0
1558 005710 005704      TST      R4      ;;SUPPRESS THIS 0?
1559 005712 001403      BEQ      5$      ;;BR IF YES
1560 005714 005204      4$:      INC      R4      ;;DON'T SUPPRESS ANYMORE 0'S
1561 005716 052703 000060      BIS      #'0, R3      ;;MAKE THIS DIGIT ASCII
1562 005722 052703 000040      5$:      BIS      #' , R3      ;;MAKE ASCII IF NOT ALREADY
1563 005726 110367 000040      MOV      R3, 8$      ;;SAVE FOR TYPING
1564 005732 104401 005772      TYPE      , 8$      ;;GO TYPE THIS DIGIT
1565 005736 105367 000032      7$:      DECB      $OCNT      ;;COUNT BY 1
1566 005742 003347      BGT      2$      ;;BR IF MORE TO DO
1567 005744 002402      BLT      6$      ;;BR IF DONE
1568 005746 005204      INC      R4      ;;INSURE LAST DIGIT ISN'T A BLANK
1569 005750 000744      BR      2$      ;;GO DO THE LAST DIGIT
1570 005752 012605      6$:      MOV      (SP)+, R5      ;;RESTORE R5
1571 005754 012604      MOV      (SP)+, R4      ;;RESTORE R4
1572 005756 012603      MOV      (SP)+, R3      ;;RESTORE R3
1573 005760 016666 000002 000004      MOV      2(SP), 4(SP)      ;;SET THE STACK FOR RETURNING
1574 005766 012616      MOV      (SP)+, (SP)

```

```
1575 005770 000002          RTI          ;;RETURN
1576 005772      000      8$: .BYTE 0      ;;STORAGE FOR ASCII DIGIT
1577 005773      000      .BYTE 0      ;;TERMINATOR FOR TYPE ROUTINE
1578 005774      000      $OCNT: .BYTE 0      ;;OCTAL DIGIT COUNTER
1579 005775      000      $OFILL: .BYTE C      ;;ZERO FILL SWITCH
1580 005776 000000      $OMODE: .WORD 0      ;;NUMBER OF DIGITS TO TYPE
1581          .SBTTL TTY INPUT ROUTINE
1582
1583          ;*****
1584 006000 177560      $TKS: .WORD 177560      ;;TTY KBD STATUS
1585 006002 177562      $TKB: .WORD 177562      ;;TTY KBD BUFFER
1586          .ENABL LSB
1587
1588          ;*****
1589          ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
1590          ;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
1591          ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
1592          ;*WHEN OPERATING IN TTY FLAG MODE.
1593 006004 022767 000176 173204 $CKSWR: CMP #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED?
1594 006012 001074          BNE 15$      ;;BRANCH IF NO
1595 006014 105777 177760      TSTB @TKS      ;;CHAR THERE?
1596 006020 100071          BPL 15$      ;;IF NO, DON'T WAIT AROUND
1597 006022 117746 177754      MOVB @TKB,-(SP)      ;;SAVE THE CHAR
1598 006026 042716 177600      BIC #^C177,(SP)      ;;STRIP-OFF THE ASCII
1599 006032 022726 000007      CMP #7,(SP)+      ;;IS IT A CONTROL G?
1600 006036 001062          BNE 15$      ;;NO, RETURN TO USER
1601 006040 126727 000514 000001 CMPB $AUTOB,#1      ;;ARE WE RUNNING IN AUTO-MODE?
1602 006046 001456          BEQ 15$      ;;BRANCH IF YES
1603
1604 006050 104401 006531      $GTSWR: TYPE , $CNTLG      ;;ECHO THE CONTROL-G (^G)
1605 006054 104401 006536      TYPE , $MSWR      ;;TYPE CURRENT CONTENTS
1606 006060 016746 172112      MOV SWREG,-(SP)      ;;SAVE SWREG FOR TYPEOUT
1607 006064 104402          TYPOC      ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
1608 006066 104401 006547      TYPE , $MNEW      ;;PROMPT FOR NEW SWR
1609 006072 005046          19$: CLR -(SP)      ;;CLEAR COUNTER
1610 006074 005046          CLR -(SP)      ;;THE NEW SWR
1611 006076 105777 177676          7$: TSTB @TKS      ;;CHAR THERE?
1612 006102 100375          BPL 7$      ;;IF NOT TRY AGAIN
1613
1614 006104 117746 177672          MOVB @TKB,-(SP)      ;;PICK UP CHAR
1615 006110 042716 177600          BIC #^C177,(SP)      ;;MAKE IT 7-BIT ASCII
1616
1617
1618
1619 006114 021627 000025          9$: CMP (SP),#25      ;;IS IT A CONTROL-U?
1620 006120 001005          BNE 10$      ;;BRANCH IF NOT
1621 006122 104401 006524          TYPE , $CNTLU      ;;YES, ECHO CONTROL-U (^U)
1622 006126 062706 000006          20$: ADD #6,SP      ;;IGNORE PREVIOUS INPUT
1623 006132 000757          BR 19$      ;;LET'S TRY IT AGAIN
1624
1625
1626 006134 021627 000015          10$: CMP (SP),#15      ;;IS IT A <CR>?
1627 006140 001022          BNE 16$      ;;BRANCH IF NO
1628 006142 005766 000004          TST 4(SP)      ;;YES, IS IT THE FIRST CHAR?
1629 006146 001403          BEQ 11$      ;;BRANCH IF YES
1630 006150 016677 000002 173040      MOV 2(SP),@SWR      ;;SAVE NEW SWR
```

```

1631 006156 062706 000006      11$:  ADD    #6,SP      ;;CLEAR UP STACK
1632 006162 104401 005547      14$:  TYPE   $,CRLF    ;;ECHO <CR> AND <LF>
1633 006166 126727 C00367 000001      CMPB  $INTAG,#1    ;;RE-ENABLE TTY KBD INTERRUPTS?
1634 006174 001003      BNE    15$        ;;BRANCH IF NOT
1635 006176 012777 000100 177574      MOV   #100,@$TKS  ;;RE-ENABLE TTY KBD INTERRUPTS
1636 006204 000002      15$:  RTI                    ;;RETURN
1637 006206 004767 177254      16$:  JSR   PC,$TYPEC  ;;ECHO CHAR
1638 006212 021627 000060      CMP   (SP),#60    ;;CHAR < 0?
1639 006216 002420      BLT   18$        ;;BRANCH IF YES
1640 006220 021627 000067      CMP   (SP),#67    ;;CHAR > 7?
1641 006224 003015      BGT   18$        ;;BRANCH IF YES
1642 006226 042726 000060      BIC   #60,(SP)+   ;;STRIP-OFF ASCII
1643 006232 005766 000002      TST   2(SP)       ;;IS THIS THE FIRST CHAR
1644 006236 001403      BEQ   17$        ;;BRANCH IF YES
1645 006240 006316      ASL   (SP)        ;;NO, SHIFT PRESENT
1646 006242 006316      ASL   (SP)        ;;CHAR OVER TO MAKE
1647 006244 006316      ASL   (SP)        ;;ROOM FOR NEW ONE.
1648 006246 005266 000002      17$:  INC   2(SP)       ;;KEEP COUNT OF CHAR
1649 006252 056616 177776      BIS   -2(SP),(SP) ;;SET IN NEW CHAR
1650 006256 000707      BR    7$         ;;GET THE NEXT ONE
1651 006260 104401 005546      18$:  TYPE   $,QUES    ;;TYPE ?<CR><LF>
1652 006264 000720      BR    20$        ;;SIMULATE CONTROL-U
1653      .DSABL  LSB
1654
1655
1656      ;;*****
1657      ;;THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
1658      ;;CALL:
1659      ;;*   RDCHR      ;;INPUT A SINGLE CHARACTER FROM THE TTY
1660      ;;*   RETURN HERE ;;CHARACTER IS ON THE STACK
1661      ;;*                ;;WITH PARITY BIT STRIPPED OFF
1662      ;;
1663
1664 006266 011646      $RDCHR: MOV   (SP),-(SP) ;;PUSH DOWN THE PC
1665 006270 016666 000004 000002      MOV   4(SP),2(SP) ;;SAVE THE PS
1666 006276 105777 177476      1$:   TSTB  @$TKS     ;;WAIT FOR
1667 006302 100375      BPL   1$         ;;A CHARACTER
1668 006304 117766 177472 000004      MOVB  @$TKB,4(SP) ;;READ THE TTY
1669 006312 042766 177600 000004      BIC   #^C<177>,4(SP) ;;GET RID OF JUNK IF ANY
1670 006320 026627 000004 000023      CMP   4(SP),#23  ;;IS IT A CONTROL-S?
1671 006326 001013      BNE   3$        ;;BRANCH IF NO
1672 006330 105777 177444      2$:   TSTB  @$TKS     ;;WAIT FOR A CHARACTER
1673 006334 100375      BPL   2$        ;;LOOP UNTIL ITS THERE
1674 006336 117746 177440      MOVB  @$TKB,-(SP) ;;GET CHARACTER
1675 006342 042716 177600      BIC   #^C177,(SP) ;;MAKE IT 7-BIT ASCII
1676 006346 022627 000021      CMP   (SP)+,#21  ;;IS IT A CONTROL-Q?
1677 006352 001366      BNE   2$        ;;IF NOT DISCARD IT
1678 006354 000750      BR    1$        ;;YES, RESUME
1679 006356 026627 000004 000140      3$:   CMP   4(SP),#140 ;;IS IT UPPER CASE?
1680 006364 002407      BLT   4$        ;;BRANCH IF YES
1681 006366 026627 000004 000175      CMP   4(SP),#175 ;;IS IT A SPECIAL CHAR?
1682 006374 003003      BGT   4$        ;;BRANCH IF YES
1683 006376 042766 000040 000004      BIC   #40,4(SP)  ;;MAKE IT UPPER CASE
1684 006404 000002      4$:   RTI                    ;;GO BACK TO USER
1685      ;;*****
1686      ;;THIS ROUTINE WILL INPUT A STRING FROM THE TTY

```



```

1687      ;*CALL:
1688      ;*      RDLIN      ;;INPUT A STRING FROM THE TTY
1689      ;*      RETURN HERE  ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
1690      ;*
1691      ;*
1692      $RDLIN: MOV      R3,-(SP)      ;;SAVE R3
1693      1$:      MOV      #$TTYIN,R3  ;;GET ADDRESS
1694      2$:      CMP      #$TTYIN+8.,R3  ;;BUFFER FULL?
1695      BLOS     4$      ;;BR IF YES
1696      RDCHR   ;;GO READ ONE CHARACTER FROM THE TTY
1697      MOVB    (SP)+,(R3)  ;;GET CHARACTER
1698      10$:    CMPB    #177,(R3)  ;;IS IT A RUBOUT
1699      BNE     3$      ;;SKIP IF NOT
1700      4$:      TYPE    ,SQUES      ;;TYPE A '?'
1701      BR      1$      ;;CLEAR THE BUFFER AND LOOP
1702      3$:      MOVB    (R3),9$      ;;ECHO THE CHARACTER
1703      TYPE    ,9$
1704      CMPB    #15,(R3)+  ;;CHECK FOR RETURN
1705      BNE     2$      ;;LOOP IF NOT RETURN
1706      CLRB    -1(R3)     ;;CLEAR RETURN (THE 15)
1707      TYPE    ,S$LF      ;;TYPE A LINE FEED
1708      MOV     (SP)+,R3    ;;RESTORE R3
1709      MOV     (SP),-(SP)  ;;ADJUST THE STACK AND PUT ADDRESS OF THE
1710      MOV     4(SP),2(SP) ;;FIRST ASCII CHARACTER ON IT
1711      MOV     #$TTYIN,4(SP)
1712      RTI      ;;RETURN
1713      9$:      .BYTE   0      ;;STORAGE FOR ASCII CHAR. TO TYPE
1714      .BYTE   0      ;;TERMINATOR
1715      $TTYIN: .BLKB   8.      ;;RESERVE 8 BYTES FOR TTY INPUT
1716      $CNTLU: .ASCIZ  /^U/<15><12>  ;;CONTROL 'U'
1717      $CNTLG: .ASCIZ  /^G/<15><12>  ;;CONTROL 'G'
1718      $MSWR:  .ASCIZ  <15><12>/SWR = /
1719      020075  000
1720      006547  040 047040 053505 $MNEW: .ASCIZ  / NEW - /
1721      006554 036440 000040
1722      006560 000
1723      006561 000
1724
1725
1726      ;;*****
1727      ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE 'TRAP' INSTRUCTION
1728      ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
1729      ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
1730      ;*GO TO THAT ROUTINE.
1731
1732      $TRAP: MOV     R0,-(SP)      ;;SAVE R0
1733      MOV     2(SP),R0      ;;GET TRAP ADDRESS
1734      TST    -(R0)        ;;BACKUP BY 2
1735      MOVB   (R0),R0      ;;GET RIGHT BYTE OF TRAP
1736      ASL   R0            ;;POSITION FOR INDEXING
1737      MOV   $TRPAD(R0),R0  ;;INDEX TO TABLE
1738      RTS   R0            ;;GO TO ROUTINE
1739
1740
1741      ;;THIS IS USE TO HANDLE THE 'GETPRI' MACRO
1742

```

```
1743 006604 011646 $TRAP2: MOV (SP),-(SP) ;;MOVE THE PC DOWN
1744 006606 016666 000004 000002 MOV 4(SP),2(SP) ;;MOVE THE PSW DOWN
1745 006614 000002 RTI ;;RESTORE THE PSW
1746
1747 .SBTTL TRAP TABLE
1748
1749 ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
1750 ;*BY THE 'TRAP' INSTRUCTION.
1751
1752 : ROUTINE
1753 : -----
1754 006616 006604 $TRPAD: .WORD $TRAP2
1755 006620 005316 $TYPE ;;CALL=TYPE TRAP+1(104401) TTY TYPEOUT ROUTINE
1756 006622 005576 $TYPOC ;;CALL=TYPOC TRAP+2(104402) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
1757 006624 005552 $TYPOS ;;CALL=TYPOS TRAP+3(104403) TYPE OCTAL NUMBER (NO LEADING ZEROS)
1758 006626 005612 $TYPON ;;CALL=TYPON TRAP+4(104404) TYPE OCTAL NUMBER (AS PER LAST CALL)
1759
1760 006630 006054 $GTSWR ;;CALL=GTSWR TRAP+5(104405) GET SOFT-SWR SETTING
1761
1762 006632 006004 $CKSWR ;;CALL=CKSWR TRAP+6(104406) TEST FOR CHANGE IN SOFT-SWR
1763 006634 006266 $RDCHR ;;CALL=RDCHR TRAP+7(104407) TTY TYPEIN CHARACTER ROUTINE
1764 006636 006406 $RDLIN ;;CALL=RDLIN TRAP+10(104410) TTY TYPEIN STRING ROUTINE
1765 006640 005015 047531 020125 MESS1: .ASCIZ <15><12>'YOU ARE ON AN XOR TESTER'<15><12>
1766 006646 051101 020105 047117
1767 006654 040440 020116 047530
1768 006662 020122 042524 052123
1769 006670 051105 005015 000
1770 006675 105 042116 050040 MESS3: .ASCIZ /END PASS/<15><12>
1771 006702 051501 006523 000012
1772 000001 .END
```


TRTVEC=	000014	420#																	
TST4	003132	1056	1061#																
TST5	003202	1068	1073#																
TST6	002760	1025#	1033																
TST9	003014	1031	1034#																
TYP	005316	1424#																	
TYPE =	104401	762	819	1200	1459	1564	1604	1605	1608	1621	1632	1651	1700	1703					
		1707	1755#																
		1607	1756#																
TYPOC =	104402																		
TYPON =	104404	1758#																	
TYPOS =	104403	1757#																	
WAIT1	004402	1264#	1265																
WAIT2	004560	1293#	1294	1298															
WGTCT	004620	1259*	1276*	1278	1283*	1305#													
WRTOC	004474	1275	1280#																
XLIST	004552	1286	1292#																
XOR	005000	1325	1345#																
XORA	001712	811	822#																
XORFLG	001240	731#	814*	824*	1322														
\$AUTOB	006560	770*	1601	1722#															
\$CHARC	005532	1461*	1471*	1478	1487*	1492#													
\$CKSWR	006004	1593#	1762																
\$CMTAG=	***** U	739																	
\$CNTLG	006531	1604	1717#																
\$CNTLU	006524	1621	1716#																
\$CRLF	005547	1460	1502#	1632	1716														
\$ENDAD	004112	697	1206#																
\$FILLC	005544	1464	1499#																
\$FILLS	005543	1498#																	
\$GTSWR	006054	1605#	1760																
\$INTAG	006561	1633	1723#																
\$LF	005550	1503#	1707	1716															
\$MAIL =	***** U	756	766	1448															
\$MNEW	006547	1608	1720#																
\$MSWR	006536	1605	1718#																
\$NULL	005542	1466	1497#																
\$OCNT	005774	1536*	1565*	1578#															
\$OMODE	005776	1531*	1535*	1540	1543*	1554*	1580#												
\$QUES	005546	1501#	1651	1700	1716														
\$RDCHR	006266	1664#	1763																
\$RDDEC=	***** U	1765																	
\$RDLIN	006406	1692#	1764																
\$RDOCT=	***** U	1765																	
\$RDSZ =	000010	1685#																	
\$R2A =	***** U	1765																	
\$SAVRE=	***** U	1765																	
\$SETUP=	000100	429#	740	762	763	1588	1722												
\$STUP =	177777	429#																	
\$TKB	006002	1585#	1597	1614	1668	1674													
\$TKS	006000	1584#	1595	1611	1635*	1666	1672												
\$TPB	005540	1484*	1496#																
\$TPFLG	005545	1442	1500#																
\$TPS	005536	1482	1495#																
\$TRAP	006562	69#	1732#																
\$TRAP2	006604	1743#	1754																
\$TRP =	000011	1747#	1756#	1757#	1758#	1759#	1760	1761#	1762	1763#	1764#	1765#							

.SRDDE	1#		
.SRDOC	1#		
.SREAD	1#	319#	1581
.SR2AZ	1#		
.\$SAVE	1#		
.\$SB2D	1#		
.\$SB2O	1#		
.\$SCOP	1#		
.\$SIZE	1#		
.\$SUPR	1#		
.\$TRAP	1#	319#	1724
.\$TYPB	1#		
.\$TYPD	1#		
.\$TYPE	1#	319#	1424
.\$TYPO	1#	1504	
.\$4OCA	1#		
.1170	1#		

. ABS. 006710 000

ERRORS DETECTED: 0

CZDRCH.BIN,CZDRCH.LST/CRF/SOL=CZDRCH.SML,CZDRCH.P11
RUN-TIME: 8 10 .6 SECONDS
RUN-TIME RATIO: 110/20=5.4
CORE USED: 32K (63 PAGES)