

KD11Z KD11W
DL11W

DL11W/1144 MFM SLU
CZOLDIO

COPYRIGHT (c) 1975-84
AH-8529I-MC
FICHE 01 OF 01

FEB 1985
digital
Made In USA

Grid of microfiche frames containing data tables and diagrams.

1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1

.REM

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

IDENTIFICATION

PRODUCT CODE: AC-8528I-MC
PRODUCT NAME: CZDL010 DL11-W/1144 MFM SLU
DATE CREATED: JULY, 1981
MAINTAINER: DIAGNOSTIC ENGINEERING
AUTHOR: DAN CASALETTO
REVISED BY: DAN MILLEVILLE

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, 1981, 1984 BY DIGITAL EQUIPMENT CORPORATION

THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION:

DIGITAL	PDP	UNIBUS	MASSBUS
DEC	DECUS	DECTAPE	

C1

39
40
41
42
43
44
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
85HISTORY SECTION

CZDL000 WAS RELEASED OCT 79

THE FOLLOWING CHANGES WERE MADE. ALL CHANGES ARE INDICATED
BY ;** IN THE COMMENT FIELD:

1. USE THE MFPT INSTRUCTION, AND IF THE CPU IS A 11/44:
 - A. DO NOT PERFORM READER ENABLE AND RECEIVER ACTIVE TESTS.
 - B. IF ERROR FLAGS TESTS AND BREAK TESTS ARE ENABLED, PERFORM THESE TESTS ONLY FOR THE SLU AT 176500. PERFORM THESE TESTS FOR THE CONSOLE SLU IF BIT03 OF SWR IS ADDITIONALLY SET.
 - C. WHILE IN MAINTENANCE MODE DO NOT WRITE AND READ A 1P OR AN ASCII 220. THIS WILL FORCE THE CONSOLE INTO 'CONSOLE MODE'.
 - D. IN THE CLOCK REPEATABILITY TEST, ALLOW FOR A TOLERANCE OF 2 BETWEEN CLOCK COUNTS. THIS IS TO ALLOW ENOUGH TOLERANCE WHEN MOS MEMORY IS USED AND REFRESH CYCLES ARE OCCURING.
2. BECAUSE 11/44 SLU INTERRUPT REQUESTS ARE DEPENDANT ON A MFM CLOCK ENOUGH TIME MUST BE GIVEN FOR THEM TO OCCUR. THEREFORE, ALL TESTS ASSOCIATED WITH XMIT & RECEIVE INTERRUPTS SHOULD HAVE A MINIMUM OF 4 NOP'S ACTING AS WAIT FOR INTERRUPT.
3. IT WAS FOUND THAT AN ATTACHED TUSB WOULD BE ACTIVATED BY THE DIAGNOSTIC, AND, AS A RESULT, CHARACTERS WOULD BE SENT TO THE RECEIVER BUFFER. THIS WOULD CAUSE SOME TESTS TO FAIL. THEREFORE TO INHIBIT ANY COMMUNICATION TO THE TUSB FROM THE DIAGNOSTIC:
 - A. ENABLE MAINTENANCE MODE IN ALL TESTS THAT WRITE TO THE XMITTER.
 - B. IN ALL TESTS THAT WITE TO XMITTER AND BEFORE LEAVING TEST; DELAY ENOUGH TIME TO ALLOW ANY CHARACTERS THAT MIGHT BE IN THE PROCESS OF BEING XMITTED TO FINISH.
 - C. DISABLE MAINTENANCE MODE FOR PURPOSE OF ERROR PRINTING ONLY TO THE SLU WHICH THE TERMINAL IS ATTACHED.

86
87
88
89
90
91
92
93
94
95
96
97

4. IT WAS FOUND THAT UPON POWERUP AND WITH THE TUSB ATTACHED, CHARACTERS WOULD BE SENT TO THE RECEIVER FROM THE TUSB. SOME RECEIVER TESTS WOULD FAIL DUE TO THIS. THEREFORE, ON ALL TESTS THAT PERFORM RECEIVER TESTS AND FOLLOWING MAINTENANCE MODE BEING ENABLED, DELAY ENOUGH TIME TO ALLOW TO ALLOW ANY CHARACTERS THAT MIGHT BE IN THE PROCESS OF BEING RECEIVED TO FINISH.
5. ADD SOFTWARE THAT WILL IMPLEMENT AUTO INITIATION OF 11/44 T/A CONSOLE TEST VIA WRAP CABLE FROM TUSB TO CONSOLE PORTS. IT IS SELECTED BY SWR BIT02 AND IS PERFORMED ONLY AFTER ALL SLUS ARE TESTED.

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

CZDLDE0 WAS RELEASED JUL 80

1. PROGRAM WAS CHANGED TO WORK WITH VECTORS GREATER THAN 377.

CZDLDF0 WAS RELEASED OCT 80

1. IN TESTS 17 THROUGH 22 THE CLOCK DONE FLAG CAN GET SET BETWEEN THE TIME THAT IT IS CLEARED (BIC) AND THE TIME THAT INTERRUPTING IS ENABLED (BIS). THE (BIC-BIS) COMBINATION WAS CHANGED TO A (MOV) INSTRUCTION TO SET INTERRUPT ENABLE AND CLEAR DONE FLAG ALL WITH ONE INSTRUCTION.
2. IN TEST 20, 3 (NOP)'S WERE ADDED BEFORE ERROR +47 TO ALLOW SUFFICIENT TIME BETWEEN CLOCK FLAG AND CLOCK INTERRUPT.
3. IN TEST 23, THE COUNT DIFFERENCE ALLOWED WAS CHANGED ON 11/44'S ONLY FROM 2 TO 3.
4. TEST 45 HAD SEVERAL PROBLEMS.
 - A. PROGRAM CAN KICK OUT OF DELAY LOOP PREMATURELY. A FIXED DELAY WAS ADDED.
 - B. ERROR +113 DID NOT PRINT. A RESET WAS ADDED IMMEDIATELY BEFORE THE ERROR CALL TO BREAK THE MAINTENANCE WRAPAROUND.
 - C. CLKCNT WAS NOT INITIALIZED.

CZDLDG0 WAS RELEASED APR 81

1. THE LINE CLOCK SYNCHRONIZING PROBLEM, WHICH WAS CORRECTED BY PATCH ORIGINALLY IN CZDLDD, WAS REMOVED IN THE NEW REVISION. THEREFORE, TEST 20 NOW CONTAINS THE PATCH TO CORRECT THIS PROBLEM.
2. RANDOM IQ'S READ ON ANY INPUT MODE WILL BE IGNORED. THIS WAS A PROBLEM WITH CERTAIN SYSTEM HOOKUPS.
3. THE \$SCOPE AND \$ERROR ROUTINES NOW CHECK FOR BIT 0 OF THE CPU ERROR REGISTER BEING SET (POWER MONITOR BIT).

CZDLDH0 WAS RELEASED JUL 81

1. IN TEST 36, THE "CLR @RBUF" INSTRUCTION WAS CHANGED TO "TST @RBUF" BECAUSE AN 11/24 DOES ONLY A WRITE, AND THE PURPOSE OF THAT INSTRUCTION IN THAT POSITION WAS TO DO A READ.

CZDLDI0 WAS RELEASED SEPTEMBER 1984

1. THE TERMINAL OUTPUT TEST DID NOT CHECK FOR XON, XOFF.

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
195
196
197
198
199
200
201

1.0 GENERAL INFORMATION

1.1 ABSTRACT

THIS DIAGNOSTIC IS A LOGIC TEST TO VERIFY THE OPERATION OF THE
THE FOLLOWING MODULES:

1. DL11-W SERIAL LINE/REAL TIME CLOCK INTERFACE
2. 11/44 MULTIFUNCTION MODULE

THE PROGRAM WILL RUN WITHOUT ANY SPECIAL TEST FIXTURES BY DEFAULT.
HOWEVER, THE FOLLOWING OPTIONAL TESTING IS PROVIDED:

1. TEST TO VERIFY XMIT AND RECEIVE OF THE UARTS WITH
A WRAP CABLE. THE UART UNDER TEST IS LOOPED BACK ON
ITSELF. THIS IS SELECTED VIA OPERATIONAL SWITCH SETTING
BIT 7, AND IS APPLICABLE TO THE DL11W AND 11/44 MFM.
2. AUTOMATIC INITIATION OF THE T/A CONSOLE TEST OF THE 11/44
MFM. THE TUSB PORTS ARE LOOPED TO THE CONSOLE PORTS
THIS IS SELECTED VIA OPERATIONAL SWITCH SETTING BIT 2
AND IS APPLICABLE TO 11/44 MFM ONLY.
(SEE CKKFBA0 FOR T/A CONSOLE TEST EXPLANATION)

THIS DIAGNOSTIC OPERATES ON THE CONSOLE SERIAL LINE AND CLOCK INTERFACES
AS WELL AS UP TO FIFTEEN(15) ADDITIONAL IDENTICALLY CONFIGURED
SERIAL LINE INTERFACES. THE DEFAULT ADDRESSES ARE:

A. CONSOLE - 177560 SERIAL LINE
177546 CLOCK

B. OTHER SERIAL LINE - 776500 FIRST SERIAL LINE ADDRESS
OF 15 CONSECUTIVE SERIAL
LINE ADDRESSES

THE PROGRAM IS DESIGNED TO RUN ON ANY PDP-11 WITH 8K OF MEMORY
. IT CAN BE RUN UNDER XXDP,APT, AND ACT MONITORS,
AND ON PROCESSORS WITH NO HARDWARE SWITCH REGISTER,
SOFTWARE SWITCH REGISTER = LOCATION 176

POWER FAILURE IS SUPPORTED FOR SYSTEMS WITH CORE MEMORY OR BATTERY
BACKUP.

NOTE: THIS DIAGNOSTIC WITH THE SWR = 000020
(CLOCK TESTS ONLY) SHOULD BE USED ON SWITCHLESS CPU'S
TO TEST KW-11L LINE CLOCK MODULES.

1.2 SYSTEM REQUIREMENTS

1.2.1 EQUIPMENT

STANDARD 11 FAMILY(COMPUTER WITH A CONSOLE OUTPUT DEVICE
AND 8K OF MEMORY.

202
203
204
205
206
207
208
209
210
211
212

OPTIONAL:

1. LOOP CABLE FOR UART LOOP BACK ON ITSELF
2. WRAP CABLE FOR 11/44 MFM T/E TESTING
(SEE DWG. #7016942 WRAP AROUND CABLE
AND SECTION 5.0 OF THIS DOCUMENT)

1.2.2 STORAGE

THE PROGRAM USES 5K WORDS OF MEMORY

214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270

1.3 ASSUMPTIONS

-
- A. IF THE UNIT UNDER TEST (UUT) IS THE CONSOLE, THE PROGRAM WILL ASSUME THE REAL TIME CLOCK (RTC) IS ENABLED AND WILL TEST IT UNLESS THE TESTS ARE DISABLED BY BIT6 OF THE SWR.
- B. IF THE UUT IS NOT THE CONSOLE, THE RTC IS NOT TESTED FOR THAT DEVICE.
- C. FOR THE DL11-W:
- THE PROGRAM WILL ASSUME THE ERROR FLAG BITS AND THE BREAK FUNCTION OF THE DL11-W ARE DISABLED AND WILL NOT TEST THESE FUNCTIONS UNLESS ENABLED BY BIT10 (FOR ERROR FLAGS) AND BIT8 (FOR BREAK) OF THE SWR. THE DEFAULT CHARACTER SIZE IS 8 BITS (SEE PARA 2.3.2).
- FOR THE 11/44:
- THE PROGRAM ASSUMES THAT THE ERROR FLAG BITS AND THE BREAK FUNCTION ARE DISABLED, AND WILL NOT TEST THESE FUNCTIONS HOWEVER, IF BIT 10 (FOR ERROR FLAGS) AND BIT 08 (FOR BREAK) OF SWR ARE SET, THEN ERROR FLAGS AND BREAK TESTS ARE PERFORMED FOR THE TUS8 SLU ONLY. IF BIT03 OF SWR IS ALSO SET THEN THESE TESTS WILL BE ENABLED FOR THE CONSOLE.
- READER ENABLE AND RECEIVER ACTIVE TESTS ARE NOT PERFORMED SINCE THE 11/44 MFM DOES NOT IMPLEMENT THESE FUNCTIONS.

2.0 OPERATING INSTRUCTIONS

2.1 LOADING PROCEDURE

USE STANDARD PROCEDURE FOR PDP-11 ABSOLUTE BINARY FORMATTED TAPES.

2.2 STARTING PROCEDURE

LOAD THE SWITCH REGISTER WITH SETTING

NOTE: IF USING A CPU WITHOUT HARDWARE SWITCH REGISTER
SOFTWARE SWITCH REGISTER LOCATION = 176.
(FOR A 11/44 CPU USE MFM CONSOLE FOR DEPOSITING
SWITCH REGISTER. TYPE +P TO ENTER CONSOLE)

- A. START AT 200.
AFTER CHECKING THE TRANSMITTER, THE PROGRAM WILL PRINT ITS IDENTIFICATION AND REPORT THE NUMBER OF DEVICES UNDER TEST (NUMBER IS OCTAL).
"END PASS" IS PRINTED AFTER A FULL PASS HAS BEEN MADE ON ALL DEVICES UNDER TEST.
- B. START AT 204. *****NOTE*****
THE "ECHO" TEST WILL BE EXECUTED. AN "*" IS PRINTED AT THE BEGINNING OF THE TEST. THE ECHO TEST READS A CHARACTER FROM

271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
C
297
C

THE TERMINAL WRITES THAT CHARACTER TO THE TERMINAL AND REPORTS ANY ERROR FLAGS SET IN THE RECEIVER BUFFER. A CONTROL-C HALTS THE TEST AND PRINTS "STOP" AT THE TERMINAL CONTINUING RESTARTS THE ECHO TEST.

C. START AT 210. *****NOTE*****
THE TERMINAL OUTPUT TEST WILL BE EXECUTED. DEPRESSING ANY CHARACTER AT THE TERMINAL, HALTS THE TEST. CONTINUING RESTARTS THE TEST. THE TEST OUTPUTS 32 CHARACTERS ON A LINE AND REPEATS THE PATTERN EVERY THREE LINES. THE PATTERN IS AS FOLLOWS (OCTAL CODE 040 --> 377):

!"#\$%&'()*+,-./0123456789:;<=>?	(OCTAL CODE)
ABCDEFGHIJKLMNPOQRSTUVWXYZ[\]^_`	(040 --> 077)
'ABCDEFGHIJKLMNPOQRSTUVWXYZ	(100 --> 137)
	(140 --> 177) [LOWER CASE ALPHA]

THIS BOTTOM LINE COULD BE THE FOLLOWING IF THE TERMINAL DOES NOT HAVE LOWER CASE:

ABCDEFGHIJKLMNPOQRSTUVWXYZ[\] [UPPER CASE ALPHA]

*****NOTE*****

IF THE TESTING ON TERMINALS OTHER THAN THE CONSOLE IS DESIRED : : * *
FOR TESTS B OR C,SEE SECTION 2.3.4. AND 2.3.5. OF THIS DOCUMENT. : : * *

299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
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

2.3 OPERATING PROCEDURE

2.3.1 OPERATIONAL SWITCH SETTINGS

THE DIAGNOSTIC WILL CHECK FOR EXISTENCE OF SWITCH REGISTER AT 177570.

IF NO SWITCH REGISTER IS AVAILABLE THE PROGRAM WILL AUTOMATICALLY USE THE CONTENTS OF LOCATION 176 AS THE SOFTWARE SWITCH REGISTER. THE USER SHOULD SET THIS LOCATION BEFORE STARTING THE PROGRAM. IF A HARDWARE SWITCH IS AVAILABLE AND A SOFTWARE SWR(LOC. 176) IS DESIRED, LOAD ALL 1'S INTO LOCATION 177570. (ALL SWITCHES UP IF PHYSICAL SWITCHES ARE AVAILABLE)

BIT15	- HALT ON ERROR
BIT14	- LOOP ON PRESENT TEST
BIT13	- INHIBIT ERROR TIMEOUT
BIT12	- UNUSED
BIT11	- UNUSED
BIT10	- ENABLE ERROR FLAGS TESTS
BIT09	- LOOP ON ERROR
BIT08	- ENABLE BREAK FUNCTION TESTS
BIT07	- ENABLE DATA TEST THROUGH LOOP-BACK CONNECTOR
BIT06	- INHIBIT RTC TESTS (ALLOW ONLY SLU TESTS)
BIT05	- ALLOW MANUAL SETTING OF "DEVH" (DEVICE MAP)
BIT04	- INHIBIT SLU TESTS (ALLOW ONLY LINE CLOCK TESTS)
BIT03	- FOR 11/44 MFM: ENABLE BOTH 'BREAK TESTS' AND 'ERROR FLAG TESTS' FOR THE CONSOLE SLU. (THIS BIT IS VALID ONLY IF BIT10 OR BIT08 IS SET.)
BIT02	- 11/44 MFM OPTION: SELECT AUTO INITIATION OF T/A CONSOLE TEST VIA WRAP CABLE

FOR DL11-W:

IF THE SOFTWARE SWITCH REGISTER IS USED(LOC. 176) THEN BUILT INTO THE PROGRAM IS THE ABILITY TO DYNAMICALLY CHANGE THE CONTENTS OF SWREG DURING PROGRAM EXECUTION. BY STRIKING +G (CNTRL G) ON CONSOLE TTY THE OPERATOR SETS A REQUEST FLAG TO CHANGE THE CONTENTS OF SWREG, WHICH IS PROCESSED IN KEY AREAS OF THE PROGRAM CODE (IE) ERROR ROUTINES, AFTER HALTS END OF PASS, AND OTHER APPLICABLE AREAS. BECAUSE THIS DIAGNOSTIC USES THE MAINTENANCE BIT OF THE SERIAL LINE, THE CONTROL-G SHOULD BE ISSUED DURING PROGRAM TYPEOUTS AT THAT TIME THE MAINTENANCE BIT IS SURE TO BE CLEAR.

IF A CONTROL-G IS DETECTED, UPON EXECUTION THE CONTENTS OF THE SWREG ARE DUMPED IN OCTAL ON THE CONSOLE TTY AND ANY CHANGES ARE REQUESTED

(IE) SWR=XXXXXX NEW=

POSSIBLE RESPONSES ARE:

356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379

1. <CR> IF NO CHANGES ARE TO BE MADE
2. 6 DIGITS 0-7 TO REPRESENT IN OCTAL THE NEW SWITCH REGISTER VALUE ;LAST DIGIT FOLLOWED BY <CR>.
3. ↑U TO ALLOW REENTERING VALUE IF ERROR IS COMMITTED KEYING IN SWREG VALUE.

FOR 11/44 CPU:

SINCE THE 11/44 HAS A HARDWARE SWITCH REGISTER LOADED BY THE MFM CONSOLE THEN THE DIAGNOSTIC SHOULD ALWAYS FIND EXISTENCE OF 177570. WHEN OPERATING ON THE 11/44 CPU, DYNAMIC CHANGING OF SWREG(177570) DURING PROGRAM EXECUTION CAN BE ACCOMPLISHED BY USING THE MFM CONSOLE. TYPING ↑P<CR> ON THE CONSOLE TTY WILL ENTER THE CONSOLE. THIS IS CONSIDERED "CONSOLE MODE". TO EXAMINE SWREG TYPE ' E SW<CR> ' TO THE CONSOLE PROMPT. TO LOAD THE SWREG TYPE ' D SW DATA<CR> ' WHERE "DATA" IS AN OCTAL NUMBER. IN ORDER FOR THE DIAGNOSTIC TO TYPE TO THE TTY IT IS NECESSARY TO HAVE THE 11/44 MFM IN "PROGRAM I/O MODE". THIS CAN BE ACCOMPLISHED BY TYPING ' C<CR> ' TO THE CONSOLE PROMPT. THEREFORE, WHEN CONSOLE USE IS COMPLETED, PLACE THE MFM IN "PROGRAM I/O MODE". BECAUSE THIS DIAGNOSTIC USES THE MAINTENANCE BIT, ↑P WILL NOT BE ACKNOWLEDGED DURING THESE TESTS. THEREFORE, ISSUE ↑P DURING PROGRAM TYPEOUTS. AT THIS TIME THE MAINTENANCE BIT WILL BE CLEARED.

2.3.2 SETTING BITS PER CHARACTER

THIS PROGRAM DEFAULTS TO TESTING 8 BITS PER CHARACTER. IF THE SERIAL LINE IS SET FOR 5-->7 BITS PER CHARACTER, SET THE MEMORY LOCATION "#USMR" AS FOLLOWS:

CHAR. SIZE (# OF BITS)	"#USMR" CONTENTS	
	(BINARY)	(OCTAL)
8	10000000	400
7	01000000	200
6	00100000	100
5	00010000	40

"#USMR" IS USED IN THE DATA PATH TESTS TO LIMIT THE BINARY COUNT TEST PATTERN TO THE NUMBER OF BITS SELECTED ON THE SERIAL LINE.

2.3.3 RUNNING UNDER APT

THE APT MAILBOX IS LOCATED AT LOCATION 500, TO ALLOW ADDITIONAL SERIAL LINE VECTOR ASSIGNMENTS TO THE 400 AREA OF MEMORY.

FOR DL11W:

THE DEFAULT EXECUTION TIMES PROVIDED(\$TSTM,\$PASTM) ARE FOR EXECUTION WITH AN 11/34 PROCESSOR, CORE MEMORY, AND 110 BAUD.

FOR 11/44:

THE EXECUTION TIMES PROVIDED IN THE APT SCRIPT THAT FOLLOWS AND IN SECT. 2.4 ARE FOR EXECUTION WITH A 11/44 PROCESSOR, CACHE, 16K CORE MEMORY, AND 300 BAUD.

THE FOLLOWING IS A PROGRAM LOAD FILE USED BY APT:

1. E TABLE 'A' IS USED FOR APT DUMP MODE AND RUN TIME MODES.
 - A. TWO SLU'S ARE TESTED.(\$SWREG BIT05=1 AND \$DEVH=3)
A SLU AT 177560(DEFAULT CONSOLE SLU) AND AT 176500
(BASE ADDRESS CODE).
 - B. THE ERROR FLAG AND BREAK TESTS ARE SELECTED FOR THE SLU
AT 176500(TU58 SLU IN MFG.)
(\$SWREG BIT 10 AND 8 =1)
 - C. THE 'ENABLE DATA TEST THROUGH LOOP-BACK CONNECTOR' TEST
IS ENABLED TO ALLOW TEST 44 TO EXECUTE (\$SWREG BIT 7 =1).
2. E TABLE 'B' IS USED FOR APT QV. IT ACCOMPLISHES WHAT E TABLE 'A' DOES, WITHOUT THE ENABLE DATA TEST THROUGH THE LOOPBACK CONNECTOR, BUT ADDITIONALLY IT SUPPRESSES ALL TYPEOUTS TO THE TERMINAL (\$ENVH=240) AND SELECTS AUTO TESTING OF T/A CONSOLE TEST VIA WRAP CABLE (\$SWREG BIT02=1).

1ST PASS
RUN TIME
60

LONGEST
TEST TIME
50

ADDITIONAL
RUN TIME
45

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

.....

E TABLES

.....

E-MODE/S-MODE
(#ENVH/#ENV)

A
200/000

B
240/001

SWITCH REGISTER 1
(#SWREG)

002640

002404

SWITCH REGISTER 2
CPU TYPE/OPTIONS

000400
00/0000

000400
00/0000

MEMORY MAP CODE 1

000/00000000

000/00000000

MEMORY MAP CODE 2

000/00000000

000/00000000

MEMORY MAP CODE 3

000/00000000

000/00000000

MEMORY MAP CODE 4

000/00000000

000/00000000

BUS PRIORITY/INTERRUPT 1

0000

0000

BUS PRIORITY/INTERRUPT 2

0000

0000

BUS ADDRESS CODE

176500

176500

DEVICE MAP CODE
(#DEVH)

000003

000003

CTLR. SPECIFIC WORD 1

000000

000000

CTLR. SPECIFIC WORD 2

000000

000000

462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512

2.3.4 RUN WITH ALTERNATE CONSOLE ADDRESS

TO USE A CONSOLE ADDRESS OTHER THAN 177560, OR VECTOR OTHER THAN 60, THE OPERATOR MUST SUPPLY THE PROGRAM WITH THE CORRECT ADDRESSES BY INSERTING THEM AT THE TAG LABELED "CRCSR":

- CRCSR: ADDRESS OF RECEIVER RCSR
- CRBUF: ADDRESS OF RECEIVER BUFFER
- CTCSR: ADDRESS OF TRANSMITTER CSR
- CTBUF: ADDRESS OF TRANSMITTER BUFFER
- CRVCT: ADDRESS OF RECEIVER VECTOR
- CRPSW: ADDRESS OF ASSOCIATED PSW
- CTVCT: ADDRESS OF TRANSMITTER VECTOR
- CTPSW: ADDRESS OF ASSOCIATED PSW

2.3.5 TESTING ADDITIONAL SERIAL LINES

THIS PROGRAM WILL SUPPORT TESTING OF MULTIPLE SLU'S. IT REQUIRES THE ADDRESS OF THE FIRST ADDITIONAL RCSR (STORED AT "\$BASE") AND ITS INTERRUPT VECTOR (STORED AT "\$VECT1"); AND WILL BE ABLE TO ADDRESS ANY SLU STARTING AT THE SPECIFIED BASE ADDRESS UP TO 15 CONSECUTIVE DEVICES.

EXAMPLE: \$BASE: 776500
 \$VECT1: 300
THE PROGRAM WILL BE ABLE TO TEST THE CONSOLE PLUS ANY ADDITIONAL DL11-W SLU'S WITHIN THE RANGE 776500 --> 776660

\$BASE AND \$VECT1 DEFAULT TO 776500 AND 300 RESPECTIVELY.

THE PROGRAM ASSOCIATES UNIT NUMBERS TO DEVICES AS FOLLOWS:
(NUMBERS IN PARENTHESIS ARE OCTAL)

- UNIT# 0 --> CONSOLE [ADDRESS STORED AT "CRCSR"]
- UNIT# 1 --> BASE ADDRESS STORED AT "\$BASE"
- ASSOCIATED BASE VECTOR STORED AT "\$VECT1"
- UNIT# 2 --> BASE ADDRESS + (10)
- BASE VECTOR + (10)
- UNIT# 3 --> BASE ADDRESS + (20)
- BASE VECTOR + (20)
- UNIT# 4 --> BASE ADDRESS + (30)
- BASE VECTOR + (30)
- ⋮
- V
- UNIT#15 --> BASE ADDRESS + (160)
- BASE VECTOR + (160)

513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539

STARTING AT LOCATION 200 AND HAVING BITS OF SWR CLEAR, THE PROGRAM WILL SELF SIZE THE NUMBER OF DEVICES (STARTING AT THE BAS ADDRESS) AND STORE A BIT MAP AT "#DEVN" (DEVICE MAP) TO INDICATE WHICH UNIT NUMBERS ARE PRESENT AND WILL BE TESTED:

```

-----
! UNIT ! UNIT ! .....! UNIT ! UNIT ! CONSOLE!
! 15 ! 14 ! .....! 2 ! 1 !
-----

```

A BIT MAP CAN BE ENTERED AT "#DEVN" PRIOR TO STARTING THE PROGRAM SETTING BITS OF THE SWR INHIBITS THE SELF-SIZING AND DEVICE MAP GENERATION, AND USES THE VALUE STORED BY THE OPERATOR.

EXAMPLE:

```

SWR = 000040          [BINARY 0 000 000 000 100 000]
#BASE: 776500
#VECT1: 300

```

```

#DEVN: 13          [BINARY - 0 000 000 000 001 011]

```

```

THE PROGRAM WILL TEST -
UNIT# 0 = CONSOLE
UNIT# 1 = 776500 ; 300
UNIT# 3 = 776520 ; 320

```

02

540
541
542
543
544
545
546
547
548
549
550
551

2.4 EXECUTION TIMES

FOR DL11-W: (110 BAUD)
LONGEST SUBJECT TIME • 50 SECONDS
PASS TIME • 60 SECONDS
ADDITIONAL DEVICES • 55 SECONDS/DEVICE

FOR 11/44: (300 BAUD)
LONGEST SUBJECT TIME • 50 SECONDS
PASS TIME • 60 SECONDS
ADDITIONAL DEVICES • 55 SECONDS/DEVICE

552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589

3.0 ERROR REPORTING

IF A ROUTINE FAILS AND THE INHIBIT ERROR TIMEOUT (BIT13) OF THE SWR IS NOT SET, A PRINTOUT RESULTS IN THE FORM:

```
"(SOME ASCII MESSAGE)"
TEST#  ERR PC  RCSR  [ANY APPLICABLE DATA HEADINGS]
XXXXXX XXXXXX XXXXXX [ANY APPLICABLE DATA]
```

NOTE: "RCSR" IS DEPENDENT ON THE FAILURE
& THEREFORE COULD BE TCSR, RBUF, TBUF, OR LKS

WHERE "XXXXXX" IS AN OCTAL NUMBER.
THIS ERROR PRINTOUT OCCURS PROVIDED THE ERROR THAT EXISTS WOULD NOT HINDER THE TIMEOUT. IN CASES WHERE IT IS NOT POSSIBLE TO PRINT AN ERROR MESSAGE (I.E. FATAL CONSOLE TRANSMITTER FAILURES), A HALT OCCURS AND THE PC CAN BE EXAMINED BY THE OPERATOR TO FIND THE ERROR INFORMATION IN THE PROGRAM LISTING.

NOTE: FOR SOFTWARE SWITCH OPERATION, THE SWITCH REGISTER CAN BE CHANGED BY TYPING A CONTROL-G AT THE CONSOLE DURING ERROR PRINTOUTS. AFTER CONTINUING FROM THE ERROR HALT THE OLD SWR CONTENTS IS DISPLAYED AND THE NEW CONTENTS CAN BE ENTERED. IF ERROR HALTS ARE DISABLED, THE CONTROL-G RESPONSE OCCURS IMMEDIATELY FOLLOWING THE TIMEOUT.

IF THE CPU THIS DIAGNOSTIC WILL BE RUN ON DOES NOT HAVE A CPU ERROR REGISTER (ADDRESS 1777766), THE FOLLOWING SECTION DOES NOT APPLY, SINCE THE ROUTINES MENTIONED PLANS FOR THAT POSSIBILITY.

IF AN ERROR SHOULD OCCUR WHEN CHECKING THE POWER MONITOR BIT IN THE SCOPE ROUTINE, THE ERROR WILL BE CALLED FROM THAT ROUTINE. IF THE BIT BECOMES SET AFTER THE SCOPE ROUTINE, AND AN ERROR OCCURS FOR ANY REASON, *TWO* ERRORS WILL CALL. THE POWER MONITOR BIT ERROR WILL CALL FIRST, THEN THE ERROR TO BE CALLED WILL CALL. IT IS A DEFINITE POSSIBILITY THAT THE ERROR WAS CAUSED BY THE POWER SUPPLY(S) THAT WERE OUT OF SPECIFICATION, AND REPAIR SHOULD BE EXECUTED BEFORE RELYING ON THE RESULTS OF THE FAILURE.

590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626

4.0 SUBROUTINE ABSTRACTS

4.1 TRAPCATCHER

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

EACH VECTOR ENTRANCE ADDRESS IS LOADED WITH THE ADDRESS OF THE NEXT LOCATION. THE NEXT LOCATION IS LOADED WITH A BREAK POINT TRAP (000003). THUS AN ILLEGAL TRAP OR INTERRUPT CAUSES A TRAP THROUGH THE BPT VECTOR(14) WHICH POINTS TO THE "CATCH" ROUTINE.

THE "CATCH" ROUTINE REPORTS THE PC THAT CAUSED THE ORIGINAL TRAP AND THE LOCATION OF THE TRAP VECTOR (IF UNDER APT, AN ERROR IS INDICATED TO APT). AFTER REPORTING THE ERROR THE PROGRAM HALTS. THE PROGRAM MUST BE RESTARTED AT THIS POINT.

4.2 WRPSW

THIS ROUTINE IS USED TO WRITE THE PSW BY POPPING VALUES FROM THE STACK. THIS METHOD IS USED TO BE COMPATIBLE WITH ALL 11 FAMILY PROCESSORS.

4.3 SCOPE

THIS ROUTINE CALL IS PLACED BETWEEN EACH SUBTEST. IT RECORDS THE STARTING ADDRESS OF EACH SUBTEST AS IT IS BEING ENTERED AND UPDATES THE TEST NUMBER. IF A SCOPE LOOP IS REQUESTED, IT WILL JUMP TO THE START OF THE SUBTEST AT WHICH THE SCOPE LOOP IS REQUESTED.

F2

628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684

4.4 ERROR

THIS ROUTINE CALL IS PLACED WHEREEVER AN ERROR REPORT IS DESIRED. THE LOWER BYTE OF THIS CALL IS USED AS THE ERROR NUMBER AND AS A POINTER INTO THE ERROR TABLE. THIS ROUTINE REPORTS ERRORS TO APT USING "%AATYPE" AND TYPES ERROR REPORTS TO THE CONSOLE USING "%ERRTYPE".

4.5 %POWER

THIS ROUTINE SAVES ALL GENERAL REGISTERS DURING POWER-DOWN AND RESTORES THEM AT POWER-UP. IF A POWER FAILURE OCCURS "POWER" IS PRINTED AT THE CONSOLE AFTER POWER IS RESTORED.

4.6 CKSMR

THIS ROUTINE CALL IS USED TO DETECT THE RECEPTION OF A CONTROL-G FROM THE CONSOLE. THE CALL USES "%READ" TO PERFORM THE CONTROL-G SEQUENCE OF DISPLAYING THE CONTENTS OF THE SOFTWARE SWITCH REGISTER AND THE ENTERING THE NEW CONTENTS FROM THE TERMINAL.

5.0 AUTOMATIC INITIATION OF 11/44 T/A CONSOLE TEST VIA WRAP CABLE

PURPOSE: THE T/E(OR T/A) CONSOLE TEST CAN BE IMPLEMENTED BY MANUALLY TYPING T/E(OR T/A) ON THE KEYBOARD OF THE TERMINAL WHEN IN CONSOLE MODE. IN ORDER TO IMPLEMENT THIS TEST IN AN AUTOMATIC WAY IN MANUFACTURING WHILE UNDER APT, THE USE OF A WRAP CABLE FROM THE TUSB TO THE CONSOLE CAN BE USED ALLOWING THIS DIAGNOSTIC TO ISSUE THE 'T/A' COMMAND TO THE CONSOLE. THE DIAGNOSTIC WILL ISSUE THE APPROPRIATE SEQUENCE OF COMMANDS TO THE CONSOLE VIA THE WRAP CABLE WHEN BIT02 OF SMR =1. THE DIAGNOSTIC WILL THEN MONITOR THE EXPECTED RESPONSE FROM THE CONSOLE VIA THE WRAP CABLE AND HALT IF THERE IS AN ERROR. THE T/A TEST IS DONE ONLY AFTER ALL SLUS ARE TESTED. IF T/A IS SUCCESSFUL 'END PASS' IS PRINTED AND THE DIAGNOSTIC STARTS AGAIN.

FIGURE 1 SHOWS THE PROPER WRAP CABLE SETUP AND REQUIREMENTS FOR AUTOMATICALLY INITIATING T/A FROM THE THE DIAGNOSTIC. NOTICE THAT THIS ARRANGEMENT ALLOWS FOR THE TERMINAL TO MONITOR ALL COMMUNICATION FROM THE CONSOLE OUTPUT DURING EXECUTION OF THE DIAGNOSTIC IN "WRAP MODE".

TYPEOUT EXAMPLES

1. WITH THE CONFIGURATION OF FIGURE 1 AND "WRAP MODE" SELECTED

62
THE PROGRAM MAY BE LOADED BY APT. IF 'C' TABLE B' OF SECTION 2.3.3 WERE USED WITH THE EXCEPTION OF ALLOWING TYPEOUTS(ENVM=200) THE FOLLOWING WOULD BE SEEN ON THE LOCAL TERMINAL:

685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726

```
CZDLDI0 DL11-W/1144 MFM SLU
02 DEVICES UNDER TEST  ↑P
CONSOLE
>>>T/A
```

```
CONSOLE-TESTB
END PASS ↑P
CONSOLE
>>>T/A
```

```
CONSOLE-TESTB
END PASS ↑P
CONSOLE
>>>T/A
```

```
CONSOLE-TESTB
END PASS ↑P  ....ETC.
```

DESCRIPTION: REFERRING TO THE ABOVE PRINTOUTS:

- A. THE DIAGNOSTIC IS LOADED WITH THE TITLE BEING PRINTED.
- B. TWO SLU DEVICES(CONSOLE, TUSB) ARE SPECIFIED
- C. BOTH SLUS ARE TESTED COMPLETELY. AT THIS POINT THE THE DIAGNOSTIC ISSUES ↑P TO THE WRAP CABLE. THE CONSOLE ENTERS CONSOLE MODE AND THE ↑P TYPED ON THE TERMINAL IS THE MFM CONSOLE ECHO.
- D. THE CONSOLE PERFORMS ITS OWN SELF TEST BY TYPING 'CONSOLE' FOLLOWED BY >>>, THE CONSOLE PROMPT.
- E. THE DIAGNOSTIC THEN ISSUES T/A AND THE TERMINAL SHOWS THE CONSOLE ECHO OF THIS.
- F. THE MFM THEN PERFORMS THE T/A TEST SHOWN BY THE TERMINAL TYPING 'CONSOLE-TESTB'. THE DIAGNOSTIC LOOKS FOR THE 'B' IN THIS TYPEOUT, AND WHEN FOUND , CONSIDERS THE TEST A SUCCESS. THE DIAGNOSTIC WILL TYPE END PASS INDICATING A SUCCESSFUL PASS OF THE DIAGNOSTIC.
- G. THE DIAGNOSTIC CONTINUES WITH FURTHER PASSES OF THE DIAGNOSTIC.

728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746

2. IF 'E TABLE B' OF 2.3.3 WERE USED WITH TYPEOUTS SUPPRESSED
(#ENVM=240) AS STATED, THEN THE FOLLOWING TYPEOUTS WOULD BE
NOTICED:

+P
CONSOLE
>>>T/A

CONSOLE-TESTB+P
CONSOLE
>>>T/A

CONSOLE-TESTB+P
CONSOLE
>>>T/A

CONSOLE-TESTB+PETC.

808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271

```

;*****
;*****
;***** .#ERROR *****
;
;.#ERROR IS THE ERROR HANDLER
;
;ARGUMENTS:
;
;1) ADRESS -- IF NON-BLANK=ADDRESS OF USER ERROR ROUTINE
;              IF BLANK AND SW13 IS USED TO INHIBIT ERROR
;              TYPEOUTS, THE PC OF THE "ERROR" WILL BE TYPED.
;              NOTE: IF SW13 IS USED TO INHIBIT ERROR
;                    TYPEOUTS A "CR" AND "LF" WILL ALWAYS
;                    BE THE LAST THING TYPED AND IS PROVIDED
;                    BY THIS ROUTINE
;
;2) INSTR -- IF NON-BLANK WILL BE THE FIRST INSTRUCTION
;             OF THE ROUTINE
;             (EXAMPLE OF USE <<MOV R1,SAVR1>>
;             NOTE: INSTR CAN BE A MACRO I.E. <<SAVE <R1,R2,R3,R4>>>)
;
;3) INSTR2 -- IF NON-BLANK WILL REPLACE THE LAST INSTRUCTION (RTI).
;             REFER TO ARGUMENT 2 (INSTR) FOR AN EXAMPLE OF USE.
;             BE SURE TO PROVIDE AN RTI FOR EXITING THE ROUTINE.
;
;NOTE: THIS ROUTINE IS CONDITIONALLY ASSEMBLED BY #SWR
;      FOR SW09,SW10,SW13,&SW15
;SW09=1 LOOP ON ERROR
;SW10=1 BELL ON ERROR
;SW13=1 INHIBIT ERROR TYPEOUTS
;SW15=1 HALT ON ERROR
;
;ROUTINES REQUIRED:
;
;1) TYPE AN ASCIZ STRING (.#TYPE) DEPENDING ON #SWR
;
;*****
;*****
;***** .#ERRTYP *****
;
;.#ERRTYP IS USED TO REPORT ERRORS
;ITS INTENDED USE IS TO BE IN CONJUCTION WITH .#ERROR
;
;ARGUMENTS:
;
;1) A -- IF BLANK ALL DATA IN THE "DATA TABLE"(DT) ARE ASSUMED TO
;        BE OCTAL NUMBERS AND ARE TYPED AS SUCH.
;
;        IF NON-BLANK THE "DT" CAN CONTAIN BOTH OCTAL AND DECIMAL
;        NUMBERS WITH THE "DATA FORMAT"(DF) INDICATING HOW EACH
;        NUMBER IS TO BE TYPED. THE "DF" MUST BE A BYTE DATA STRING

```

1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1382
1386
1390
1394
1395
1402
1403
1409
1419
1431
1432
1433
1434
1435
1436
1437
1438

```

; WITH "0" FOR OCTAL AND "1" FOR DECIMAL.
;
; ROUTINES REQUIRED:
;1) .#TYPOCT TYPE AN OCTAL NUMBER
;2) .#TYPDEC TYPE A DECIMAL NUMBER (ONLY NEEDED IF "A" IS NON-BLANK).
;
; NOTES:
;1) THIS ROUTINE PROVIDES AN AUTOMATIC "CARRIAGE RETURN-LINE FEED"
; FOR "EM", "DH" AND "DT".
;2) TWO(2) SPACES ARE TYPED AFTER EACH NUMBER FOR "DT".
;
; EXAMPLE OF USE WITH .#ERROR
;
; .#ERROR ERRTYP ;:#ERRTYP IS ENTRY POINT OF .#ERRTYP
; .#ERRTYP X ;:"DT" CONTAINS OCTAL AND DECIMAL NUMBERS
;
;*****

```

001100
104000
000004

000011
000012
000015
000200
177776
177776
177774
177772
177570
177570

000000
000001

```

.#CALL NEWTST, #NEWTEST, .#TYPE, .#TYPOC, .#TRAP
.#CALL .SETUP, STARS, PUSH, POP, SETUP, .EQUIV
.#CALL .#APTHDR, .#APTBL, .#ACT11
.#CALL .#CHTAG, .#EOP, .#READ
.#CALL .EQUAT
.#SBTTL BASIC DEFINITIONS
; *INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
STACK= 1100
ERROR=EMT
SCOPE=IOT
; *MISCELLANEOUS DEFINITIONS
HT= 11 ;:CODE FOR HORIZONTAL TAB
LF= 12 ;:CODE FOR LINE FEED
CR= 15 ;:CODE FOR CARRIAGE RETURN
CRLF= 200 ;:CODE FOR CARRIAGE RETURN-LINE FEED
PS= 177776 ;:PROCESSOR STATUS WORD
PSW=PS
STKLMT= 177774 ;:STACK LIMIT REGISTER
PIRQ= 177772 ;:PROGRAM INTERRUPT REQUEST REGISTER
DSWR= 177570 ;:HARDWARE SWITCH REGISTER
DDISP= 177570 ;:HARDWARE DISPLAY REGISTER
; *GENERAL PURPOSE REGISTER DEFINITIONS
R0= #0 ;:GENERAL REGISTER
R1= #1 ;:GENERAL REGISTER

```

```
000002 R2= #2 ; GENERAL REGISTER
000003 R3= #3 ; GENERAL REGISTER
000004 R4= #4 ; GENERAL REGISTER
000005 R5= #5 ; GENERAL REGISTER
000006 R6= #6 ; GENERAL REGISTER
000007 R7= #7 ; GENERAL REGISTER
000006 SP= #6 ; STACK POINTER
000007 PC= #7 ; PROGRAM COUNTER

; *PRIORITY LEVEL DEFINITIONS
000000 PR0= 0 ; PRIORITY LEVEL 0
000040 PR1= 40 ; PRIORITY LEVEL 1
000100 PR2= 100 ; PRIORITY LEVEL 2
000140 PR3= 140 ; PRIORITY LEVEL 3
000200 PR4= 200 ; PRIORITY LEVEL 4
000240 PR5= 240 ; PRIORITY LEVEL 5
000300 PR6= 300 ; PRIORITY LEVEL 6
000340 PR7= 340 ; PRIORITY LEVEL 7

; *"SWITCH REGISTER" SWITCH DEFINITIONS
100000 SW15= 100000
040000 SW14= 40000
020000 SW13= 20000
010000 SW12= 10000
004000 SW11= 4000
002000 SW10= 2000
001000 SW09= 1000
000400 SW08= 400
000200 SW07= 200
000100 SW06= 100
000040 SW05= 40
000020 SW04= 20
000010 SW03= 10
000004 SW02= 4
000002 SW01= 2
000001 SW00= 1
001000 SW9=SW09
000400 SW8=SW08
000200 SW7=SW07
000100 SW6=SW06
000040 SW5=SW05
000020 SW4=SW04
000010 SW3=SW03
000004 SW2=SW02
000002 SW1=SW01
000001 SW0=SW00

; *DATA BIT DEFINITIONS (BIT00 TO BIT15)
100000 BIT15= 100000
040000 BIT14= 40000
020000 BIT13= 20000
010000 BIT12= 10000
004000 BIT11= 4000
002000 BIT10= 2000
001000 BIT09= 1000
000400 BIT08= 400
000200 BIT07= 200
000100 BIT06= 100
000040 BIT05= 40
000020 BIT04= 20
```

```

000010 BIT03= 10
000004 BIT02= 4
000002 BIT01= 2
000001 BIT00= 1
001000 BIT9=BIT09
000400 BIT8=BIT08
000200 BIT7=BIT07
000100 BIT6=BIT06
000040 BIT5=BIT05
000020 BIT4=BIT04
000010 BIT3=BIT03
000004 BIT2=BIT02
000002 BIT1=BIT01
000001 BIT0=BIT00

; *BASIC "CPU" TRAP VECTOR ADDRESSES
000004 ERRVEC= 4 ; TIME OUT AND OTHER ERRORS
000010 RESVEC= 10 ; RESERVED AND ILLEGAL INSTRUCTIONS
000014 TBITVEC=14 ; "T" BIT
000014 TRTVEC= 14 ; TRACE TRAP
000014 BPTVEC= 14 ; BREAKPOINT TRAP (BPT)
000020 IOTVEC= 20 ; INPUT/OUTPUT TRAP (IOT) **SCOPE**
000024 PMRVEC= 24 ; POWER FAIL
000030 EMTVEC= 30 ; EMULATOR TRAP (EMT) **ERROR**
000034 TRAPVEC=34 ; "TRAP" TRAP
000060 TKVEC= 60 ; TTY KEYBOARD VECTOR
000064 TPVEC= 64 ; TTY PRINTER VECTOR
000240 PIRQVEC=240 ; PROGRAM INTERRUPT REQUEST VECTOR

1439 000007 MFPT=7
1440 176500 ABASE= 176500
1441 000300 AVECT1= 300
1442 000400 AUSWR= 400
1443 000001 $TN= 1
1444 161000 $SWR= 161000
1445 000003 BPT= 000003 ; THIS IS THE COMMAND FOR A TRAP
1446 ; THROUGH 14 (BPT TRAP)
1447
1448 000000 .=0
1449 ; *****
1450 ; *ALL UNUSED LOCATIONS FROM 4-776 CONTAIN A ".+2,BPT"
1451 ; *SEQUENCE TO CATCH ILLEGAL TRAPS & INTERRUPTS
1452 ; *LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
1453
1461
1462 000014 .=14 ; THE BPT TRAP VECTOR POINTS TO THE
1463 000014 014520 .WORD CATCH ; ILLEGAL TRAP HANDLER "CATCH"
1464 000016 000340 .WORD 340
1465
1466 000042 .= 42
1467 000042 000000 .WORD 0
1468
1469
1471
1472
1473 000174 .= 174
1474 000174 000000 DISPREG: .WORD 0
1475 000176 000000 SWREG: .WORD 0
1476
    
```

N2

1477		000200		.	=200			
1478	000200	000137	003046	JMP	START	:	DO	INTERFACE TEST
1479	000204	000137	017600	JMP	ECHO	:	DO	ECHO TEST
1480	000210	000137	020020	JMP	OUTTST	:	DO	OUTPUT TEST TO TERMINAL

1485

```

.SBTTL APT PARAMETER BLOCK
;*****
;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
;*****
      000500      .IX.      ;;SAVE CURRENT LOCATION
000024 000024      .-24     ;;SET POWER FAIL TO POINT TO START OF PROGRAM
      000200      200      ;;FOR APT START UP
000044 000044      .-44     ;;POINT TO APT INDIRECT ADDRESS PNTR.
      000500      $APTHDR  ;;POINT TO APT HEADER BLOCK
      000500      .-.IX    ;;RESET LOCATION COUNTER
;*****
;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
;INTERFACE SPEC.
000500 $APTHD:
000500 000000 $HIBTS: .WORD 0      ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
000502 001066 $MBAOR: .WORD $MAIL  ;;ADDRESS OF APT MAILBOX (BITS 0-15)
000504 000050 $TSTM: .WORD 50     ;;RUN TIM OF LONGEST TEST
000506 000060 $PASTH: .WORD 60    ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
000510 000055 $UNITH: .WORD 55    ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
000512 000030      .WORD $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)

```



```

001114 000000      $CPUOP: .WORD  ACPUOP  ;;CPU TYPE,OPTIONS
                    ;*      BITS 15-11=CPU TYPE
                    ;*      11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
                    ;*      11/70=06,PDQ=07,Q=10
                    ;*      BIT 10=REAL TIME CLOCK
                    ;*      BIT 9=FLOATING POINT PROCESSOR
                    ;*      BIT 8=MEMORY MANAGEMENT
001116      000      $MMS1: .BYTE  AMMS1  ;;HIGH ADDRESS,M.S. BYTE
001117      000      $MTP1: .BYTE  AMTP1  ;;MEM. TYPE,BLK#1
                    ;*      MEM.TYPE BYTE -- (HIGH BYTE)
                    ;*      900 NSEC CORE=001
                    ;*      300 NSEC BIPOLAR=002
                    ;*      500 NSEC MOS=003
001120 000000      $MADR1: .WORD  AMADR1 ;;HIGH ADDRESS,BLK#1
                    ;*      MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
001122      000      $MMS2: .BYTE  AMMS2  ;;HIGH ADDRESS,M.S. BYTE
001123      000      $MTP2: .BYTE  AMTP2  ;;MEM. TYPE,BLK#2
001124 000000      $MADR2: .WORD  AMADR2 ;;MEM.LAST ADDRESS,BLK#2
001126      000      $MMS3: .BYTE  AMMS3  ;;HIGH ADDRESS,M.S.BYTE
001127      000      $MTP3: .BYTE  AMTP3  ;;MEM. TYPE,BLK#3
001130 000000      $MADR3: .WORD  AMADR3 ;;MEM.LAST ADDRESS,BLK#3
001132      000      $MMS4: .BYTE  AMMS4  ;;HIGH ADDRESS,M.S.BYTE
001133      000      $MTP4: .BYTE  AMTP4  ;;MEM. TYPE,BLK#4
001134 000000      $MADR4: .WORD  AMADR4 ;;MEM.LAST ADDRESS,BLK#4
001136 000300      $VECT1: .WORD  AVECT1 ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
001140 000000      $VECT2: .WORD  AVECT2 ;;INTERRUPT VECTOR#2BUS PRIORITY#2
001142 176500      $BASE: .WORD  ABASE  ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
001144 000000      $DEVH: .WORD  ADEVH  ;;DEVICE MAP
001146      $ETEND:
                    .MEXIT

```

```
.SBTTL ERROR POINTER TABLE
; *THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
; *THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
; *LOCATION #ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
; *NOTE1: IF #ITEMB IS 0 THE ONLY PERTINENT DATA IS (#ERRPC).
; *NOTE2: EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:
; *      EM      ; POINTS TO THE ERROR MESSAGE
; *      DM      ; POINTS TO THE DATA HEADER
; *      DT      ; POINTS TO THE DATA
; *      DF      ; POINTS TO THE DATA FORMAT
```

Index	ItemB	ItemA	Pointer	Description
1488	001146			
1489	001146	020134	EM1	;"CAN NOT ACCESS TCSR"
1490	001150	024315	DM1	;"TEST# ERR PC TCSR"
1491	001152	025076	DT1	;"TESTN,#ERRPC,TCSR"
1492	001154	000000	0	
1493				
1494	001156	020160	EM2	;"CAN NOT ACCESS TBUF"
1495	001160	024342	DM2	;"TEST# ERR PC TBUF"
1496	001162	025106	DT2	;"TESTN,#ERRPC,TBUF"
1497	001164	000000	0	
1498				
1499	001166	020204	EM3	;"TCSR DONE NOT CLEARED WITH TBUF FULL"
1500	001170	024315	DM1	;"TEST# ERR PC TCSR"
1501	001172	025076	DT1	;"TESTN,#ERRPC,TCSR"
1502	001174	000000	0	
1503				
1504	001176	020251	EM4	;"TCSR DONE NOT SET"
1505	001200	024315	DM1	;"TEST# ERR PC TCSR"
1506	001202	025076	DT1	;"TESTN,#ERRPC,TCSR"
1507	001204	000000	0	
1508				
1509	001206	020273	EM5	;"TCSR DONE NOT SET WITH RESET"
1510	001210	024315	DM1	;"TEST# ERR PC TCSR"
1511	001212	025076	DT1	;"TESTN,#ERRPC,TCSR"
1512	001214	000000	0	
1513				
1514	001216	020330	EM6	;"CAN NOT ACCESS RCSR"
1515	001220	024367	DM6	;"TEST# ERR PC RCSR"
1516	001222	025116	DT6	;"TESTN,#ERRPC,RCSR"
1517	001224	000000	0	

1518	001226	020354	EM7	;"CAN NOT ACCESS RBUF"
1519	001230	024414	DH7	;"TEST# ERR PC RBUF"
1520	001232	025126	DT7	;"TESTN,#ERRPC,RBUF"
1521	001234	000000	0	
1522				
1523	001236	020400	EM10	;"CAN NOT ACCESS LKS"
1524	001240	024441	DH10	;"TEST# ERR PC LKS"
1525	001242	025136	DT10	;"TESTN,#ERRPC,LKS"
1526	001244	000000	0	
1527				
1528	001246	020423	EM11	;"BIT0 OF TCSR NOT CLEAR AFTER RESET"
1529	001250	024315	DH1	;"TEST# ERR PC TCSR"
1530	001252	025076	DT1	;"TESTN,#ERRPC,TCSR"
1531	001254	000000	0	
1532				
1533	001256	020466	EM12	;"CAN NOT SET BIT0 OF TCSR"
1534	001260	024315	DH1	;"TEST# ERR PC TCSR"
1535	001262	025076	DT1	;"TESTN,#ERRPC,TCSR"
1536	001264	000000	0	
1537				
1538	001266	020517	EM13	;"CAN NOT CLEAR BIT0 OF TCSR"
1539	001270	024315	DH1	;"TEST# ERR PC TCSR"
1540	001272	025076	DT1	;"TESTN,#ERRPC,TCSR"
1541	001274	000000	0	
1542				
1543	001276	020552	EM14	;"RESET DID NOT CLEAR BIT0 OF TCSR"
1544	001300	024315	DH1	;"TEST# ERR PC TCSR"
1545	001302	025076	DT1	;"TESTN,#ERRPC,TCSR"
1546	001304	000000	0	
1547				
1548	001306	020613	EM15	;"BIT2 OF TCSR NOT CLEAR AFTER RESET"
1549	001310	024315	DH1	;"TEST# ERR PC TCSR"
1550	001312	025076	DT1	;"TESTN,#ERRPC,TCSR"
1551	001314	000000	0	
1552				
1553	001316	020656	EM16	;"CAN NOT SET BIT2 OF TCSR"
1554	001320	024315	DH1	;"TEST# ERR PC TCSR"
1555	001322	025076	DT1	;"TESTN,#ERRPC,TCSR"
1556	001324	000000	0	
1557				
1558	001326	020707	EM17	;"CAN NOT CLEAR BIT2 OF TCSR"
1559	001330	024315	DH1	;"TEST# ERR PC TCSR"
1560	001332	025076	DT1	;"TESTN,#ERRPC,TCSR"
1561	001334	000000	0	

1562	001336	020742	EM20	;"RESET DID NOT CLEAR BIT2 OF TCSR"
1563	001340	024315	DH1	;"TEST# ERR PC TCSR"
1564	001342	025076	DT1	;"TESTN,#ERRPC,TCSR"
1565	001344	000000	0	
1566				
1567	001346	021003	EM21	;"BIT6 OF TCSR NOT CLEAR AFTER RESET2"
1568	001350	024315	DH1	;"TEST# ERR PC TCSR"
1569	001352	025076	DT1	;"TESTN,#ERRPC,TCSR"
1570	001354	000000	0	
1571				
1572	001356	021046	EM22	;"XMIT INTERRUPT WITH PRIORITY 7"
1573	001360	024315	DH1	;"TEST# ERR PC TCSR"
1574	001362	025076	DT1	;"TESTN,#ERRPC,TCSR"
1575	001364	000000	0	
1576				
1577	001366	021103	EM23	;"CAN NOT SET BIT6 OF TCSR"
1578	001370	024315	DH1	;"TEST# ERR PC TCSR"
1579	001372	025076	DT1	;"TESTN,#ERRPC,TCSR"
1580	001374	000000	0	
1581				
1582	001376	021133	EM24	;"CAN NOT CLEAR BIT6 OF TCSR"
1583	001400	024315	DH1	;"TEST# ERR PC TCSR"
1584	001402	025076	DT1	;"TESTN,#ERRPC,TCSR"
1585	001404	000000	0	
1586				
1587	001406	021167	EM25	;"RESET DID NOT CLEAR BIT6 OF TCSR"
1588	001410	024315	DH1	;"TEST# ERR PC TCSR"
1589	001412	025076	DT1	;"TESTN,#ERRPC,TCSR"
1590	001414	000000	0	
1591				
1592	001416	021236	EM26	;"BIT6 OF RCSR NOT CLEAR AFTER RESET"
1593	001420	024367	DH6	;"TEST# ERR PC RCSR"
1594	001422	025116	DT6	;"TESTN,#ERRPC,RCSR"
1595	001424	000000	0	
1596				
1597	001426	021273	EM27	;"RCVR INTERRUPT WITH PRIORITY 7"
1598	001430	024367	DH6	;"TEST# ERR PC RCSR"
1599	001432	025116	DT6	;"TESTN,#ERRPC,RCSR"
1600	001434	000000	0	
1601				
1602	001436	021332	EM30	;"CAN NOT SET BIT6 OF RCSR"
1603	001440	024367	DH6	;"TEST# ERR PC RCSR"
1604	001442	025116	DT6	;"TESTN,#ERRPC,RCSR"
1605	001444	000000	0	

1606	001446	021363	EM31	;"CAN NOT CLEAR BIT6 OF RCSR"
1607	001450	024367	DM6	;"TEST# ERR PC RCSR"
1608	001452	025116	DT6	;"TESTN,#ERRPC,RCSR"
1609	001454	000000	0	
1610				
1611	001456	021416	EM32	;"CAN NOT CLEAR BIT6 OF RCSR WITH RESET2"
1612	001460	024367	DM6	;"TEST# ERR PC RCSR"
1613	001462	025116	DT6	;"TESTN,#ERRPC,RCSR"
1614	001464	000000	0	
1615				
1616	001466	021464	EM33	;"BIT6 OF LKS NOT CLEAR AFTER RESET"
1617	001470	024441	DM10	;"TEST# ERR PC LKS"
1618	001472	025136	DT10	;"TESTN,#ERRPC,LKS"
1619	001474	000000	0	
1620				
1621	001476	021526	EM34	;"LKS INTERRUPT WITH PRIORITY 7"
1622	001500	024441	DM10	;"TEST# ERR PC LKS"
1623	001502	025136	DT10	;"TESTN,#ERRPC,LKS"
1624	001504	000000	0	
1625				
1626	001506	021564	EM35	;"CAN NOT SET BIT6 OF LKS"
1627	001510	024441	DM10	;"TEST# ERR PC LKS"
1628	001512	025136	DT10	;"TESTN,#ERRPC,LKS"
1629	001514	000000	0	
1630				
1631	001516	021614	EM36	;"CAN NOT CLEAR BIT6 OF LKS"
1632	001520	024441	DM10	;"TEST# ERR PC LKS"
1633	001522	025136	DT10	;"TESTN,#ERRPC,LKS"
1634	001524	000000	0	
1635				
1636	001526	021646	EM37	;"RESET DID NOT CLEAR BIT6 OF LKS"
1637	001530	024441	DM10	;"TEST# ERR PC LKS"
1638	001532	025136	DT10	;"TESTN,#ERRPC,LKS"
1639	001534	000000	0	
1640				
1641	001536	021706	EM40	;"DUAL ADDRESSING ERROR"
1642	001540	024465	DM40	;"TEST# ERR PC GOOD BAD"
1643	001542	025146	DT40	;"TESTN,#ERRPC,#GDADR,#BDCSR"
1644	001544	000000	0	
1645				
1646	001546	021734	EM41	;"BIT7 OF LKS NOT SET AFTER RESET2"
1647	001550	024441	DM10	;"TEST# ERR PC LKS"
1648	001552	025136	DT10	;"TESTN,#ERRPC,LKS"
1649	001554	000000	0	

1650	001556	021774	EM42	;"CAN NOT CLEAR BIT7 OF LKS"
1651	001560	024441	DH10	;"TEST# ERR PC LKS"
1652	001562	025136	DT10	;"TESTN,#ERRPC,LKS"
1653	001564	000000	0	
1654				
1655	001566	022026	EM43	;"BIT7 OF LKS DOES NOT SET"
1656	001570	024441	DH10	;"TEST# ERR PC LKS"
1657	001572	025136	DT10	;"TESTN,#ERRPC,LKS"
1658	001574	000000	0	
1659				
1660	001576	022057	EM44	;"RTC INTERRUPT AT PRIORITY 7"
1661	001600	024441	DH10	;"TEST# ERR PC LKS"
1662	001602	025136	DT10	;"TESTN,#ERRPC,LKS"
1663	001604	000000	0	
1664				
1665	001606	022113	EM45	;"RTC INTERRUPTS WHEN DISABLED"
1666	001610	024441	DH10	;"TEST# ERR PC LKS"
1667	001612	025136	DT10	;"TESTN,#ERRPC,LKS"
1668	001614	000000	0	
1669				
1670	001616	022150	EM46	;"RTC INTERRUPT DID NOT OCCUR"
1671	001620	024441	DH10	;"TEST# ERR PC LKS"
1672	001622	025136	DT10	;"TESTN,#ERRPC,LKS"
1673	001624	000000	0	
1674				
1675	001626	022150	EM47	;"RTC INTERRUPT DID NOT OCCUR"
1676	001630	024441	DH10	;"TEST# ERR PC LKS"
1677	001632	025136	DT10	;"TESTN,#ERRPC,LKS"
1678	001634	000000	0	
1679				
1680	001636	022204	EM50	;"RTC DOUBLE INTERRUPT"
1681	001640	024441	DH10	;"TEST# ERR PC LKS"
1682	001642	025136	DT10	;"TESTN,#ERRPC,LKS"
1683	001644	000000	0	
1684				
1685	001646	022231	EM51	;"RESET DID NOT CLEAR RTC INTERRUPT"
1686	001650	024441	DH10	;"TEST# ERR PC LKS"
1687	001652	025136	DT10	;"TESTN,#ERRPC,LKS"
1688	001654	000000	0	
1689				
1690	001656	022261	EM52	;"RTC INTERRUPT DID NOT CLEAR WITH BIT7 OF LKS"
1691	001660	024441	DH10	;"TEST# ERR PC LKS"
1692	001662	025136	DT10	;"TESTN,#ERRPC,LKS"
1693	001664	000000	0	

1694	001666	022336	EM53	
1695	001670	024521	DH53	; "TEST# ERR PC LKS CNT1 CNT2"
1696	001672	025160	DT53	; #TESTN, #ERRPC, LKS, FIRST, SECND
1697	001674	000000	0	
1698				
1699	001676	022362	EM54	; "XMIT INTERRUPTS WHEN DISABLED"
1700	001700	024315	DH1	; "TEST# ERR PC TCSR"
1701	001702	025076	DT1	; #TESTN, #ERRPC, TCSR
1702	001704	000000	0	
1703				
1704	001706	022520	EM55	; "XMIT DID NOT INTERRUPT"
1705	001710	024315	DH1	; "TEST# ERR PC TCSR"
1706	001712	025076	DT1	; #TESTN, #ERRPC, TCSR
1707	001714	000000	0	
1708				
1709	001716	022420	EM56	; "XMIT INTERRUPT AT PRIORITY 7"
1710	001720	024315	DH1	; "TEST# ERR PC TCSR"
1711	001722	025076	DT1	; #TESTN, #ERRPC, TCSR
1712	001724	000000	0	
1713				
1714	001726	022456	EM57	; "XMIT INTERRUPTS WITH ENABLE CLEAR"
1715	001730	024315	DH1	; "TEST# ERR PC TCSR"
1716	001732	025076	DT1	; #TESTN, #ERRPC, TCSR
1717	001734	000000	0	
1718				
1719	001736	022520	EM60	; "XMIT DID NOT INTERRUPT"
1720	001740	024315	DH1	; "TEST# ERR PC TCSR"
1721	001742	025076	DT1	; #TESTN, #ERRPC, TCSR
1722	001744	000000	0	
1723				
1724	001746	022547	EM61	; "XMIT RE-INTERRUPTED"
1725	001750	024315	DH1	; "TEST# ERR PC TCSR"
1726	001752	025076	DT1	; #TESTN, #ERRPC, TCSR
1727	001754	000000	0	
1728				
1729	001756	022573	EM62	; "LOADING TBUF DID NOT CLEAR INTERRUPT"
1730	001760	024315	DH1	; "TEST# ERR PC TCSR"
1731	001762	025076	DT1	; #TESTN, #ERRPC, TCSR
1732	001764	000000	0	
1733				
1734	001766	022640	EM63	; "RCVR ACTIVE NOT SET"
1735	001770	024367	DH6	; "TEST# ERR PC RCSR"
1736	001772	025116	DT6	; #TESTN, #ERRPC, RCSR
1737	001774	000000	0	

1738	001776	022664	EM64	;"RECEIVER DONE NEVER SET"
1739	002000	024367	DM6	;"TEST# ERR PC RCSR"
1740	002002	025116	DT6	;"TESTN,#ERRPC,RCSR"
1741	002004	000000	0	
1742				
1743	002006	022710	EM65	;"RCVR ACTIVE NOT CLEARED WITH DONE SET2"
1744	002010	024367	DM6	;"TEST# ERR PC RCSR"
1745	002012	025116	DT6	;"TESTN,#ERRPC,RCSR"
1746	002014	000000	0	
1747				
1748	002016	022756	EM66	;"RESET DID NOT CLEAR RCVR DONE"
1749	002020	024367	DM6	;"TEST# ERR PC RCSR"
1750	002022	025116	DT6	;"TESTN,#ERRPC,RCSR"
1751	002024	000000	0	
1752				
1753	002026	023014	EM67	;"RDR ENABLE SET DID NOT CLEAR RCVR DONE"
1754	002030	024367	DM6	;"TEST# ERR PC RCSR"
1755	002032	025116	DT6	;"TESTN,#ERRPC,RCSR"
1756	002034	000000	0	
1757				
1758	002036	023057	EM70	;"READING RBUF DID NOT CLEAR RCVR DONE"
1759	002040	024367	DM6	;"TEST# ERR PC RCSR"
1760	002042	025116	DT6	;"TESTN,#ERRPC,RCSR"
1761	002044	000000	0	
1762				
1763	002046	023124	EM71	;"RCVR INTERRUPTS WITH ENABLE CLEAR"
1764	002050	024367	DM6	;"TEST# ERR PC RCSR"
1765	002052	025116	DT6	;"TESTN,#ERRPC,RCSR"
1766	002054	000000	0	
1767				
1768	002056	023273	EM72	;"RCVR DID NOT INTERRUPT"
1769	002060	024367	DM6	;"TEST# ERR PC RCSR"
1770	002062	025116	DT6	;"TESTN,#ERRPC,RCSR"
1771	002064	000000	0	
1772				
1773	002066	023166	EM73	;"RCVR INTERRUPTS AT PRIORITY 7"
1774	002070	024367	DM6	;"TEST# ERR PC RCSR"
1775	002072	025116	DT6	;"TESTN,#ERRPC,RCSR"
1776	002074	000000	0	
1777				
1778	002076	023224	EM74	;"RCVR INTERRUPT REQUEST PASSED WITH ENABLE CLEAR"
1779	002100	024367	DM6	;"TEST# ERR PC RCSR"
1780	002102	025116	DT6	;"TESTN,#ERRPC,RCSR"
1781	002104	000000	0	

1782	002106	023273	EM75	;"RCVR DID NOT INTERRUPT"
1783	002110	024367	DM6	;"TEST# ERR PC RCSR"
1784	002112	025116	DT6	;\$TESTN,\$ERRPC,RCSR
1785	002114	000000	0	
1786	002116	023322	EM76	;"RECEIVER RE-INTERRUPTED"
1787	002120	024367	DM6	;"TEST# ERR PC RCSR"
1788	002122	025116	DT6	;\$TESTN,\$ERRPC,RCSR
1789	002124	000000	0	
1790				
1791	002126	023346	EM77	;"READING RBUF DID NOT CLEAR INTERRUPT"
1792	002130	024367	DM6	;"TEST# ERR PC RCSR"
1793	002132	025116	DT6	;\$TESTN,\$ERRPC,RCSR
1794	002134	000000	0	
1795				
1796	002136	023413	EM100	;"RESET DID NOT CLEAR RCVR INTERRUPT"
1797	002140	024367	DM6	;"TEST# ERR PC RCSR"
1798	002142	025116	DT6	;\$TESTN,\$ERRPC,RCSR
1799	002144	000000	0	
1800				
1801				
1802	002146	023456	EM101	;"'OR' FLAG DID NOT SET"
1803	002150	024367	DM6	
1804	002152	025116	DT6	;\$TESTN,\$ERRPC,RCSR
1805	002154	000000	0	
1806				
1807	002156	023504	EM102	;"'ERROR' NOT SET WITH 'OR' FLAG"
1808	002160	024367	DM6	;"TEST# ERR PC RCSR"
1809	002162	025116	DT6	;\$TESTN,\$ERRPC,RCSR
1810	002164	000000	0	
1811	002166	023543	EM103	;"BREAK DID NOT TRANSMIT ALL ZEROES"
1812	002170	024566	DM103	;"TEST# ERR PC RCSR DATA"
1813	002172	025174	DT103	;\$TESTN,\$ERRPC,RCSR,\$BDAT
1814	002174	000000	0	
1815				
1816	002176	023601	EM104	;"BREAK DID NOT SET FRAMING ERROR"
1817	002200	024367	DM6	;"TEST# ERR PC RCSR"
1818	002202	025116	DT6	;\$TESTN,\$ERRPC,RCSR
1819	002204	000000	0	
1820				
1821	002206	023636	EM105	;"DATA COMPARE ERROR"
1822	002210	024623	DM105	;"TEST# ERR PC RCSR GOOD BAD"
1823	002212	025206	DT105	;\$TESTN,\$ERRPC,RCSR,GD,BD
1824	002214	000000	0	

N3

1825	002216	023661	EM106	; "LOOP-BACK DATA COMPARE ERROR"
1826	002220	024623	DH105	; "TEST# ERR PC RCSR GOOD BAD"
1827	002222	025206	DT105	; #TESTN, #ERRPC, RCSR, GD, BD
1828	002224	000000	0	
1829				
1830	002226	023716	EM107	; "TIMEOUT IN EXERCISER TEST"
1831	002230	024367	DH6	; "TEST# ERR PC RCSR"
1832	002232	025116	DT6	; #TESTN, #ERRPC, RCSR
1833	002234	000000	0	
1834				
1835	002236	023750	EM110	; "INCORRECT RECEIVE COUNT"
1836	002240	024667	DH110	; "TEST# ERR PC RCSR TRANS RCV"
1837	002242	025222	DT110	; #TESTN, #ERRPC, RCSR, XMTCNT, RVCNT
1838	002244	000000	0	
1839				
1840	002246	024000	EM111	; "DATA COMPARE ERROR IN EXERCISER"
1841	002250	024623	DH105	; "TEST# ERR PC RCSR GOOD BAD"
1842	002252	025206	DT105	; #TESTN, #ERRPC, RCSR, GD, BD
1843	002254	000000	0	
1844				
1845	002256	024040	EM112	; "TRAP CATCHER"
1846	002260	024733	DH112	; "TEST# ERR PC RCSR OLDPC TRAP ADR"
1847	002262	025236	DT112	; #TESTN, #ERRPC, RCSR, OLDPC, BDVECT
1848	002264	000000	0	
1849				
1850	002266	024055	EM113	; "NO CLK INTERRUPT IN EXERCISER"
1851	002270	024441	DH10	; "TEST# ERR PC LKS"
1852	002272	025136	DT10	; #TESTN, #ERRPC, LKS
1853	002274	000000	0	
1854				
1855	002276	024113	EM114	; "'ERROR' NOT SET WITH 'FR' FLAG"
1856	002300	024367	DH6	; "TEST# ERR PC RCSR"
1857	002302	025116	DT6	; #TESTN, #ERRPC, RCSR
1858	002304	000000	0	
1859				
1860	002306	024152	EM115	; RCV ACTIVE NOT CLEAR WITH INIT
1861	002310	024367	DH6	; "TEST# ERR PC RCSR"
1862	002312	025116	DT6	; #TESTN, #ERRPC, RCSR
1863	002314	000000	0	
1864				
1865	002316	024211	EM116	; RCV ACTIVE WITHOUT "START" BIT
1866	002320	024367	DH6	; "TEST# ERR PC RCSR"
1867	002322	025116	DT6	; #TESTN, #ERRPC, RCSR
1868	002324	000000	0	
1869				
1870	002326	024250	EM117	; RDR ENABLE NOT CLEAR WITH RCV ACTIVE
1871	002330	024367	DH6	; "TEST# ERR PC RCSR"
1872	002332	025116	DT6	; #TESTN, #ERRPC, RCSR
1873	002334	000000	0	
1874				
1875				
1876	002336	000000		; STORAGE LOCATIONS
1877	002340	000000	FIRST: .WORD	0
1878	002342	000000	LOC1: .WORD	0
1879	002344	000000	LOC2: .WORD	0
1880	002346		SAVE0: .WORD	0
1881	002412		SAVLOC: .BLKW	22
			ENDSTK: .BLKW	10
				; TIMER LOOP COUNTER
				; TIMER LOOP COUNTER
				; STORAGE FOR LOCATIONS
				; THAT T/E USES
				; JIMS SPECIAL STACK (WRAP AROUND)

1882	002432	000000				JIMSTK: .WORD	0	
1883	002434	000000				SAVEPS: .WORD	0	;SAVE PSW AREA
1884	002436	000000				OLDSUM: .WORD	0	;CHECKSUM STORAGE
1885	002440	000000				RCVONT: .WORD	0	
1886	002442	000000				XMTONT: .WORD	0	
1887	002444	000000				CLKONT: .WORD	0	
1888	002446	000000				STPSW: .WORD	0	
1889	002450	000000				SPPSW: .WORD	0	
1890	002452	000000				SCPSW: .WORD	0	
1891	002454					BUF: .BLKW	44	
1892	002564	020	000			CNTLP: .ASCIZ	<20>	
1893	002566	136	120	000		PROMPT: .ASCII	/1P/<0><CR><LF>/CONSOLE/<CR><LF>/>>>/<377>	
1894	002610	124	057	101		TA: .ASCIZ	0T/A0<CR><LF>	
1895								
1896								
1897	002616	000000				SECD: .WORD	0	
1898	002620	000000				OLDPC: .WORD	0	
1899	002622	000000				BOVECT: .WORD	0	
1900								
1901								
1902								
1903								
1904								
1905	002624	000000				CTSTFL: .WORD	0	;CONSOLE UNDER TEST FLAG
1906	002626	000000				TMP1: .WORD	0	;TEMP LOCATION FOR TABLE OFFSETS
1907	002630	000000				TMP2: .WORD	0	;TEMP LOCATION FOR DEVICE COUNT
1908	002632	000000				TMP3: .WORD	0	;LOCATION FOR DEVICE MAP BIT TEST MASK
1909								;REGISTER AND VECTOR ADDRESSES FOR THE DL-11W UNDER TEST
1910								
1911	002634	000000				RCSR: .WORD	0	
1912	002636	000000				RBUF: .WORD	0	
1913	002640	000000				TCSR: .WORD	0	
1914	002642	000000				TBUF: .WORD	0	
1915	002644	000000				RVECT: .WORD	0	
1916	002646	000000				RPSW: .WORD	0	
1917	002650	000000				TVECT: .WORD	0	
1918	002652	000000				TPSW: .WORD	0	
1919								
1920								;CONSOLE REGISTER AND VECTOR ADDRESSES FOR THE DL-11W
1921								
1922	002654	177560				CRCSR: 177560		;ADDRESS OF RECEIVER COMMAND/STATUS REGISTER
1923	002656	177562				CRBUF: 177562		;ADDRESS OF RECEIVER BUFFER
1924	002660	177564				CTCSR: 177564		;ADDRESS OF TRANSMITTER COMMAND/STATUS REGISTER
1925	002662	177566				CTBUF: 177566		;ADDRESS OF TRANSMITTER BUFFER
1926	002664	000060				CRVECT: 60		;RECEIVER INTERRUPT VECTOR
1927	002666	000062				CRPSW: 62		
1928	002670	000064				CTVECT: 64		;TRANSMITTER INTERRUPT VECTOR
1929	002672	000066				CTPSW: 66		
1930								
1931								;REAL TIME CLOCK REGISTER AND VECTOR ADDRESSES
1932	002674	177546				LKS: .WORD	177546	
1933	002676	000100				RTCVT: .WORD	100	
1934	002700	000102				RTCPSW: .WORD	102	
1935								
1936	002702					ADRTBL: .BLKW	20	
1937	002742					VCTTBL: .BLKW	20	
1938	003002	000000				FLAG44: .WORD	0	

1939 003004 176500
 1940 003006 176502
 1941 003010 176504
 1942 003012 176506
 1943
 1944
 1945
 1946
 1947 003014 012702 002702
 1948 003020 013700 001142
 1949 003024 010001
 1950 003026 062701 000170
 1951 003032 010022
 1952 003034 062700 000010
 1953 003040 020001
 1954 003042 003773
 1955 003044 000207
 1956
 1957
 1958
 1959 003046 005037 001070
 1960 003052 005037 001066
 1961 003056 005037 001072
 1962 003062 005037 002624
 1963 003066 005037 001076
 1964 003072 005037 001100
 1965 003076 005037 003002
 1966 003102 005737 001112
 1967 003106 001003
 1968 003110 012737 000400 001112
 1969 003116 012737 000006 000004
 1970 003124 012737 000003 000006
 1971
 1972

TURCSR: .WORD 176500
 TURBUF: .WORD 176502
 TUTCSR: .WORD 176504
 TUTBUF: .WORD 176506

;SUBROUTINE TO GENERATE DEVICE ADDRESS TABLE

DEVADR: MOV #ADRtbl,R2 ;POINT R2 TO THE DEVICE ADDRESS TABLE
 MOV #BASE,R0 ;LOAD BASE DEVICE ADDRESS IN R0
 MOV R0,R1 ;
 ADD #170,R1 ;POINT R1 TO LAST DEVICE ADDRESS
 @@: MOV R0,(R2). ;MOVE DEVICE ADDRESS TO TABLE
 ADD #10,R0 ;POINT R0 TO NEXT DEVICE ADDRESS
 CMP R0,R1 ;FINISHED GENERATING TABLE?
 BLE @@ ;BR, IF LAST DEVICE ADDRESS NOT LOADED
 RTS PC

START: CLR #FATAL ;CLEAR ERROR NO.
 CLR #MSGTYP ;CLEAR MESSAGE TYPE
 CLR #TESTN ;CLEAR TEST NO.
 CLR CTSTFL ;CLEAR CONSOLE UNDER TEST FLAG
 CLR #DEVCT ;CLEAR DEVICE COUNT
 CLR #UNIT ;CLEAR UNIT NUMBER
 CLR FLAG44 ;** CLEAR 11/44 CPU FLAG
 TST #USMR ;IS #USMR LOADED?
 BNE @@ ;BR IF YES
 MOV #400,#USMR ;ELSE, DEFAULT TO #USMR=400
 @@: MOV #6,#@4 ;INITIALIZE TIMEOUT VECTORS TO TRAP
 MOV #3,#@6 ;CATCHER ROUTINE

.SBTTL INITIALIZE THE COMMON TAGS

;;CLEAR THE COMMON TAGS (#CHTAG) AREA
 MOV #CHTAG,R6 ;;FIRST LOCATION TO BE CLEARED
 CLR (R6). ;;CLEAR MEMORY LOCATION
 CMP #SMR,R6 ;;DONE?
 BNE .-6 ;;LOOP BACK IF NO
 MOV #1000,SP ;;SETUP THE STACK POINTER
 ;;INITIALIZE A FEW VECTORS
 MOV #SCOPE,#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
 MOV #340,#IOTVEC+2 ;;LEVEL 7
 MOV #ERROR,#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
 MOV #340,#EMTVEC+2 ;;LEVEL 7
 MOV #TRAP,#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
 MOV #340,#TRAPVEC+2 ;;LEVEL 7
 MOV #PWRDN,#PWRVEC ;;POWER FAILURE VECTOR
 MOV #340,#PWRVEC+2 ;;LEVEL 7
 MOV #ENDCT,#EOPCT ;;SETUP END-OF-PROGRAM COUNTER
 CLR #ESCAPE ;;CLEAR THE ESCAPE ON ERROR ADDRESS
 MOVB #1,#ERMAX ;;ALLOW ONE ERROR PER TEST
 MOV #.,#LPADR ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
 MOV #.,#LPERR ;;SETUP THE ERROR LOOP ADDRESS
 ;;SIZE FOR A HARDWARE SWITCH REGISTER, IF NOT FOUND OR IT IS
 ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
 MOV #ERRVEC,-(SP) ;;SAVE ERROR VECTOR

003132 012706 001000
 003136 005026
 003140 022706 001040
 003144 001374
 003146 012706 001000
 003152 012737 015374 000020
 003160 012737 000340 000022
 003166 012737 014544 000030
 003174 012737 000340 000032
 003202 012737 017522 000034
 003210 012737 000340 000036
 003216 012737 015212 000024
 003224 012737 000340 000026
 003232 013737 014356 014350
 003240 005037 001060
 003244 112737 000001 001015
 003252 012737 003252 001006
 003260 012737 003260 001010
 003266 013746 000004

```

003272 012737 003326 000004      MOV      #64, @ERRVEC      ;;SET UP ERROR VECTOR
003300 012737 177570 001040      MOV      @CSWR, SWR        ;;SETUP FOR A HARDWARE SWICH REGISTER
003306 012737 177570 001042      MOV      @DISP, DISPLAY    ;;AND A HARDWARE DISPLAY REGISTER
003314 022777 177777 175516      CMP      #-1, @SWR         ;;TRY TO REFERENCE HARDWARE SWR
003322 001012                               BNE      66;               ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                ;;AND THE HARDWARE SWR IS NOT - 1
                                ;;BRANCH IF NO TIMEOUT
                                ;;SET UP FOR TRAP RETURN
003324 000403                               BR       65;
003326 012716 003334 64;:      MOV      #65,.(SP)
003332 000002                               RTI
003334 012737 000176 001040 65;:  MOV      @SWREG, SWR        ;;POINT TO SOFTWARE SWR
003342 012737 000174 001042      MOV      @DISPREG, DISPLAY
003350 012637 000004 66;:      MOV      (SP), @ERRVEC     ;;RESTORE ERROR VECTOR
003354 005037 001074          CLR      #PASS             ;;CLEAR PASS COUNT
003360 132737 000200 001107      BITB    @APTSIZE, #ENVM    ;;TEST USER SIZE UNDER APT
003366 001403          BEQ     67;               ;;YES, USE NON-APT SWITCH
003370 012737 001110 001040      MOV      @SWREG, SWR      ;;NO, USE APT SWITCH REGISTER
003376                               67;:

1973
1974                               ;SIZE FOR 11/44 CPU
1975
1976 003376 013746 000010      MOV      @010, -(SP)      ;;** SAVE VECTOR
1977 003402 012737 003430 000010      MOV      @10, @010        ;;** SET UP FOR TRAP
1978 003410 000007          MFPT                       ;;** WHAT CPU??
1979 003412 122700 000001      CMPB    #1, R0            ;;** ARE WE A 11/44
1980 003416 001007          BNE     11;               ;;** NO GO RESET VECTOR
1981 003420 052737 000001 003002      BIS     #1, @FLAG44       ;;** YES SET FLAG
1982 003426 000403          BR     11;               ;;** SKIP RTI
1983 003430 012716 003436 10;:      MOV      @11,.(SP)        ;;** SET STACK RETURN
1984 003434 000002          RTI                       ;;** RETURN
1985 003436 012637 000010 11;:      MOV      (SP), @010        ;;** RESTORE VECTOR
1986 003442 032777 000020 175370      BIT     @BIT4, @SWR        ;;TEST CLOCK ONLY?
1987 003450 001404          BEQ     INIT              ;;BR IF NOT
1988 003452 005237 002624          INC     CTSTFL            ;;ELSE, SET CONSOLE TEST FLAG TO ENABLE CLOCK TESTS
1989 003456 000137 004620          JMP     ID                 ;; AND JUMP TO TYPE PROGRAM ID
1990 003462 132737 000001 001106 INIT:  BITB    @BIT0, #ENV        ;;CHECK IF ON APT
1991 003470 001404          BEQ     MANL              ;;BR IF NOT APT
1992 003472 132737 000200 001107      BITB    @BIT7, #ENVM      ;;DID APT SIZE
1993 003500 001056          BNE     APTSZD            ;;BR, IF APT SIZED
1994 003502 032777 000040 175330 MANL:  BIT     @BIT5, @SWR        ;;HAS "#DEVH" MANUALLY SET?
1995 003510 001052          BNE     APTSZD            ;;IF YES, SKIP SELF-SIZING
1996
1997 003512 004737 003014  SIZE:  JSR     PC, DEVADR         ;;GENERATE DEVICE ADDRESS TABLE
1998 003516 005037 002630          CLR     TMP2              ;;CLR TEMP LOCATION TO KEEP DEVICE COUNT
1999 003522 005037 001144          CLR     #DEVH             ;;CLEAR DEVICE MAP
2000 003526 013703 000004          MOV     @04, R3           ;;SAVE TIMEOUT VECTOR
2001 003532 012737 003562 000004      MOV     @4, @04           ;;SET TIMEOUT POINTER
2002 003540 013700 001142          MOV     #BASE, R0         ;;LOAD BASE ADDRESS
2003 003544 062700 000160          ADD     #160, R0          ;;POINT R0 TO UNIT #15 (UNIT#0=CONSOLE)
2004 003550 005710 3;:      TST     (R0)              ;;CHECK FOR DEVICE EXISTANCE
2005 003552 005237 001144          INC     #DEVH             ;;INDICATE DEVICE EXISTANCE IN DEVICE MAP
2006 003556 005237 002630          INC     TMP2              ;;INCREMENT DEVICE COUNT
2007 003562 012706 001000 4;:      MOV     #1000, SP         ;;RESET STACK POINTER
2008 003566 006337 001144          ASL     #DEVH             ;;ADJUST DEVICE MAP FOR NEXT UNIT CHECK
2009 003572 162700 000010          SUB     #10, R0           ;;POINT R0 TO NEXT DEVICE NUMBER
2010 003576 023700 001142          CMP     #BASE, R0         ;;FINISHED SIZING?
2011 003602 003762          BLE     3;               ;;BR, IF BASE ADDRESS HAS NOT BEEN CHECKED
2012 003604 013700 002654          MOV     CRCSR, R0         ;;LOAD CONSOLE DEVICE ADDRESS

```

```

2013 003610 012737 003630 000004      MOV      #51,004      ;SET UP TIMEOUT POINTER
2014 003616 005710              TST      (R0)         ;TEST FOR CONSOLE EXISTANCE
2015 003620 005237 001144      INC      #DEVH       ;INDICATE CONSOLE EXISTANCE IN DEVICE MAP
2016 003624 005237 002630      INC      TMP2        ;INCREMENT DEVICE COUNT
2017 003630 010337 000004      S1:     MOV      R3,004 ;RESTORE TIMEOUT VECTOR
2018
2019 003634 000415              BR       VCTADR      ;BR TO GENERATE VECTOR ADDRESS TABLE
2020
2021 003636 005037 002630      APTSZD: CLR     TMP2      ;CLEAR TEMP LOCATION TO KEEP DEVICE CNT
2022 003642 013702 001144      MOV      #DEVH,R2   ;MOVE DEVICE MAP TO R2
2023 003646 005702              TSTDVM: TST     R2     ;TEST MSB OF DEVICE MAP
2024 003650 100002              BPL     10          ;BR, IF MSB IS ZERO
2025 003652 005237 002630      INC      TMP2        ;INCREMENT DEVICE COUNT, IF MSB=1
2026 003656 006302      10:     ASL     R2     ;SHIFT NEXT BIT INTO MSB POSITION
2027 003660 001401              BEQ     DVADT       ;BR, IF NO OTHER BITS ARE SET IN #DEVH
2028 003662 000771              BR      TSTDVM      ;CONTINUE CHECKING #DEVH, IF MORE BITS SET
2029 003664 004737 003014      DVADT:  JSR     PC,DEVADR ;GENERATE DEVICE ADDRESS TABLE
2030
2031              ;GENERATE VECTOR ADDRESS TABLE
2032
2033 003670 012702 002742      VCTADR: MOV     #VCTTEL,R2 ;GET LOCATION OF VECTOR TABLE
2034 003674 013700 001136      MOV     @#VECT1,R0   ;COPY BASE VECTOR
2035 003700 042700 177000      BIC     #177000,R0  ;CLEAR UPPER BITS
2036 003704 010001              MOV     R0,R1       ;
2037 003706 062701 000170      ADD     #170,R1     ;POINT R1 TO LAST DEVICE VECTOR
2038 003712 010022      10:     MOV     R0,(R2)+ ;PUT VECTOR ADDRESS IN TABLE
2039 003714 062700 000010      ADD     #10,R0     ;POINT R0 TO NEXT VECTOR ADDRESS
2040 003720 020001              CMP     R0,R1       ;FINISHED GENERATING VECTOR TABLE?
2041 003722 003773              BLE     10          ;BR, IF LAST VECTOR IS NOT LOADED
2042
2043              ;MOVE DEVICE COUNT INTO DEVICE COUNT MESSAGE
2044
2045 003724 013700 002630      MOV     TMP2,R0     ;COPY DEVICE COUNT INTO R0
2046 003730 005001              CLR     R1          ;CLEAR AUXILIARY REGISTER
2047 003732 000300              SWAB   R0           ;PUT DEVICE COUNT IN UPPER BYTE OF R0
2048 003734 006300              ASL     R0          ;MOVE MSB OF COUNT INTO
2049 003736 006300              ASL     R0          ;MSB OF R0
2050 003740 006300      SHIFT: ASL     R0          ;PUT MSB OF COUNT INTO CARRY
2051 003742 106101              ROLB   R1           ;MOVE MSB OF COUNT INTO R1
2052 003744 006300              ASL     R0          ;MOVE NEXT BIT TO CARRY
2053 003746 106101              ROLB   R1           ;MOVE INTO R1
2054 003750 006300              ASL     R0          ;MOVE LAST BIT OF DIGIT
2055 003752 106101              ROLB   R1           ;INTO R1
2056 003754 062701 000060      ADD     #60,R1     ;CONVERT DIGIT TO ASCII
2057 003760 000301              SWAB   R1           ;MOVE DIGIT TO UPPER BYTE
2058 003762 032701 000020      BIT     #BIT4,R1   ;HAVE BOTH DIGITS BEEN MOVED TO R1?
2059 003766 001764              BEQ     SHIFT      ;BR, IF NOT
2060 003770 010137 025046      MOV     R1,M2A     ;MOVE DEVICE COUNT TO OUTPUT MESSAGE
2061
2062
2063 003774 052737 000002 002632 BEGIN: BIS     #BIT1,TMP3 ;SET UP BIT MASK TO TEST #DEVH FOR DEVICES EXCEPT CONSOLE
2064 004002 005037 002626      CLR     TMP1        ;CLEAR LOCATION TO STORE TABLE OFFSETS
2065 004006 032737 000001 001144      BIT     #BIT0,#DEVH ;IS CONSOLE TO BE TESTED?
2066 004014 001001              BNE     TCONS      ;BR, IF CONSOLE IS TO BE TESTED
2067 004016 000414              BR      TSTDEV     ;BR, TO TEST OTHER DEVICES
2068 004020 005237 002624      TCONS: INC     CTSTFL ;INDICATE CONSOLE UNDER TEST
2069 004024 012700 002654      MOV     #CRCRSR,R0 ;SET UP CONSOLE DEVICE ADDRESSES

```

```

2070 004030 012701 002634          MOV    #RCSR,R1      ;POINT R1 TO UUT ADDRESS TABLE
2071 004034 012021          18:   MOV    (R0),.(R1). ;TRANSFER CONSOLE ADDRESSES
2072 004036 022701 002652          CMP    #TPSW,R1     ;FINISHED TRANSFER?
2073 004042 002374          BGE   18            ;BR, IF NOT
2074 004044 000137 004172          JMP    TST1         ;GO TEST CONSOLE INTERFACE
2075
2076          ;PREPARE ADDRESSES AND VECTORS FOR UUT
2077 004050 033737 002632 001144 TSTDEV: BIT    TMP3,#DEVM ;CHECK TO SEE IF DEVICE IS TO BE TESTED
2078 004056 001010          BNE   SETADR       ;BR, IF YES
2079 004060 006337 002632          ASL   TMP3         ;SHIFT MASK TO CHECK NEXT #DEVM BIT
2080 004064 062737 000002 002626 ADD    #2,TMP1      ;INCREMENT TABLE INDEX
2081 004072 005237 001100          INC   #UNIT        ;INCREMENT UNIT NUMBER
2082 004076 000764          BR    TSTDEV       ;GO TEST NEXT BIT OF DEVICE MAP
2083
2084 004100 005237 001100          SETADR: INC   #UNIT ;UPDATE UNIT NUMBER
2085 004104 006337 002632          ASL   TMP3         ;UPDATE DEVICE MAP TEST MASK
2086 004110 013702 002626          MOV   TMP1,R2      ;MOVE TABLE OFFSET TO R2
2087 004114 062737 000002 002626 ADD    #2,TMP1      ;UPDATE TABLE OFFSET FOR NEXT DEVICE
2088 004122 016200 002707          MOV   ADRTBL(R2),R0 ;PUT UUT ADDRESS INTO R0
2089 004126 012701 002634          MOV   #RCSR,R1     ;POINT R1 TO STORAGE AREA FOR UUT ADDRESSES
2090 004132 010021          ADR:  MOV   R0,(R1). ;TRANSFER UUT ADDRESS
2091 004134 062700 000002          ADD   #2,R0        ;POINT TO NEXT UUT REGISTER
2092 004140 030027 000006          BIT   R0,#6        ;FINISHED TRANSFER?
2093 004144 001372          BNE   ADR          ;BR, IF NOT
2094
2095 004146 016200 002742          VECT: MOV   VCTTBL(R2),R0 ;PUT UUT VECTOR INTO R0
2096 004152 010021          MOV   R0,(R1).     ;TRANSFER UUT VECTORS TO ACTIVE TABLE AREA
2097 004154 062700 000002          ADD   #2,R0        ;POINT TO NEXT VECTOR
2098 004160 030027 000006          BIT   R0,#6        ;FINISHED TRANSFER?
2099 004164 001372          BNE   VECT         ;BR, IF NOT
2100 004166 000137 004172          JMP    TST1         ;GO TEST DEVICE
    
```

2101

.SBTTL TEST # 1 - TEST ABILITY TO REFERENCE TCSR

 ;*TEST 1 TEST ABILITY TO REFERENCE TCSR

2102	004172	000004			TST1: SCOPE	
	004174	013703	070004		MOV B#4,R3	;SAVE TIMEOUT VECTOR
2103	004200	012737	004214	000004	MOV #18,B#4	;SET UP TIMEOUT VECTOR
2104	004206	005777	176426		TST @TCSR	;REFERENCE THE XMIT COMMAND/STATUS REG.
2105	004212	000412			BR 4#	; GO TO END OF TEST
2106						
2107	004214	022626			1#: CMP (SP)+,(SP)+	;RESTORE SP AFTER TIMEOUT
2108	004216	005737	002624		TST CTSTFL	;CHECK IF DEVICE IS CONSOLE
2109	004222	001002			BNE 2#	;IF YES, SKIP ERROR TYPEOUT
2110	004224	104001			ERROR +1	;REPORT ERROR TO APT & TTY
2111	004226	000404			BR 4#	;BR TO END OF TEST
2112	004230				2#:	
	004230	004737	015714		JSR PC,#ATY4	;;ONLY REPORT A FATAL ERROR
	004234	000001			1	;;THE ERROR NUMBER (FROM APT LIST)
2113	004236	000000			3#: HALT	
2114	004240	010337	000004		4#: MOV R3,B#4	;RESTORE TIMEOUT VECTOR
2115						
2116						
2117						

2118

```
.SBTTL TEST # 2 - TEST ABILITY TO REFERENCE TBUF
;*****
;TEST 2 TEST ABILITY TO REFERENCE TBUF
;*****
TST2: SCOPE
      MOV     R3,R3          ;SAVE TIMEOUT VECTOR
      MOV     R1,R4          ;SET UP TIMEOUT VECTOR
      TST     @TBUF          ;REFERENCE THE XMIT BUFFER
      BR     4;              ;GO TO END OF TEST

      1;:    CMP     (SP), (SP) ;RESTORE SP AFTER TIMEOUT
           TST     CTSTFL      ;CHECK IF DEVICE IS CONSOLE
           BNE     2;          ;IF YES, SKIP ERROR TYPEOUT
           ERROR  +2          ;REPORT ERROR TO APT & TTY
           BR     4;          ;BR TO END OF TEST

      2;:    JSR     PC,@ATY4      ;ONLY REPORT A FATAL ERROR
           2;              ;THE ERROR NUMBER (FROM APT LIST)

      3;:    HALT

      4;:    MOV     R3,R4          ;RESTORE TIMEOUT VECTOR
```

```
004244 000004
2119 004246 013703 000004
2120 004252 012737 004266 000004
2121 004260 005777 176356
2122 004264 000412
2123
2124 004266 022626
2125 004270 005737 002624
2126 004274 001002
2127 004276 104002
2128 004300 000404
2129 004302
      004302 004737 015714
      004306 000002
2130 004310 000000
2131 004312 010337 000004
```

2132

.SBTTL TEST # 3 - TEST THAT BIT2(MAINT. BIT) CAN BE SET & RESET

 ;*TEST 3 TEST THAT BIT2(MAINT. BIT) CAN BE SET & RESET

2133	004316	000004				TST3:	SCOPE		
	004320	000005					RESET		;CLEAR EVERYTHING
2134	004322	032777	000004	176310			BIT	#BIT2,0TCSR	;TEST FOR BIT2 OF TCSR CLEAR
2135	004330	001411					BEQ	30	;BR IF CLEAR
2136									
2137	004332	005737	002624				TST	CTSTFL	;CHECK IF DEVICE IS CONSOLE
2138	004336	001002					BNE	10	;IF YES, SKIP ERROR TYPEOUT
2139	004340	104015					ERROR	+15	;BIT2 OF TCSR NOT CLEAR AFTER RESET
2140	004342	000404					BR	30	
2141									
2142	004344				10:				
	004344	004737	015714				JSR	PC,0ATY4	;;ONLY REPORT A FATAL ERROR
	004350	000015					15		;;THE ERROR NUMBER (FROM APT LIST)
2143	004352	000000			20:		HALT		
2144									
2145	004354	052777	000004	176256	30:		BIS	#BIT2,0TCSR	;SET BIT2 OF TCSR
2146	004362	032777	000004	176250			BIT	#BIT2,0TCSR	;TEST FOR BIT2 SET
2147	004370	001001					BNE	40	;BR IF SET
2148									
2149	004372	104016					ERROR	+16	;BIT2 OF TCSR WILL NOT SET
2150									
2151	004374	042777	000004	176236	40:		BIC	#BIT2,0TCSR	;CLEAR BIT2 OF TCSR
2152	004402	032777	000004	176230			BIT	#BIT2,0TCSR	;TEST BIT2 CLEAR
2153	004410	001411					BEQ	70	;BR IF CLEAR
2154									
2155	004412	005737	002624				TST	CTSTFL	;CHECK IF DEVICE IS CONSOLE
2156	004416	001002					BNE	50	;IF YES, SKIP ERROR TYPEOUT
2157	004420	104017					ERROR	+17	
2158	004422	000404					BR	70	
2159	004424				50:				
	004424	004737	015714				JSR	PC,0ATY4	;;ONLY REPORT A FATAL ERROR
	004430	000017					17		;;THE ERROR NUMBER (FROM APT LIST)
2160	004432	000000			60:		HALT		;BIT0 OF TCSR WILL NOT CLEAR
2161									
2162	004434	052777	000004	176176	70:		BIS	#BIT2,0TCSR	;SET BIT2 OF TCSR
2163	004442	000005					RESET		;CLEAR BIT2 WITH RESET
2164	004444	032777	000004	176166			BIT	#BIT2,0TCSR	;TEST FOR BIT2 CLEAR
2165	004452	001404					BEQ	100	;** IF CLEAR, GO TO NEXT TEST
2166									
2167	004454	042777	000004	176156			BIC	#BIT2,0TCSR	;CLEAR BIT2, TO PRINT ERROR
2168	004462	104020					ERROR	+20	
2169									;RESET DID NOT CLEAR BIT2 OF TCSR
2170									
2171	004464	000240			100:		NOP		;**

2172

.SBTTL TEST # 4 - TEST THAT TCSR BIT7(DONE) CLEARS WHEN XBUF IS LOADED

 ;*TEST 4 TEST THAT TCSR BIT7(DONE) CLEARS WHEN XBUF IS LOADED
 ;*****

2173	004466	000004			TST4: SCOPE		
2173	004470	052777	000004	176142	BIS	#BIT2,BTCSR	;** ENABLE MAINT. WRAP
2174	004476	005077	176140		CLR	BTBUF	;LOAD XBUF
2175	004502	105777	176132		TSTB	BTCSR	;CHECK DONE
2176	004506	100021			BPL	3#	;BR IF CLEAR
2177							;FILL SECOND BUFFER, BECAUSE REFRESH COULD CAUSE
2178							; FIRST TEST TO FAIL
2179	004510	005077	176126		CLR	BTBUF	;FILL DOUBLE BUFFER
2180	004514	105777	176120		TSTB	BTCSR	;CHECK DONE
2181	004520	100014			BPL	3#	;BR IF CLEAR
2182							
2183	004522	005737	002624		TST	CTSTFL	;CHECK IF DEVICE IS CONSOLE
2184	004526	001005			BNE	1#	;IF YES, SKIP ERROR TYPEOUT
2185	004530	042777	000004	176122	BIC	#BIT2,BCTCSR	;** DISABLE MAINTENANCE MODE FOR
							;** CONSOLE TO ALLOW FOR COMMUNICATION
							;** WITH TERMINAL.
2186	004536	104003			ERROR	+3	;DONE NOT CLEARED WITH TBUF FULL
2187	004540	000404			BR	3#	;BR TO END OF TEST
2188	004542				1#:		
	004542	004737	015714		JSR	PC,#ATY4	;; ONLY REPORT A FATAL ERROR
	004546	000003			3		;; THE ERROR NUMBER (FROM APT LIST)
2189	004550	000000			2#:	HALT	;TCSR "DONE" NOT CLEARED WITH TBUF FULL
2190	004552	005000			3#:	CLR	;CLEAR TIMER
2191	004554	105777	176060		4#:	TSTB	;CHECK FOR XMIT DONE
2192	004560	100417				BMI	;IF DONE SETS, BR TO END OF TEST
2193	004562	005200				INC	;INCREMENT TIMER
2194	004564	001373				BNE	;BR IF TIMER NOT DONE
2195							
2196	004566	005737	002624		TST	CTSTFL	;CHECK IF DEVICE IS CONSOLE
2197	004572	001005			BNE	5#	
2198	004574	042777	000004	176056	BIC	#BIT2,BCTCSR	;** DISABLE MAINTENANCE MODE FOR
							;** CONSOLE TO ALLOW FOR COMMUNICATION
							;** WITH TERMINAL.
2199	004602	104004			ERROR	+4	;TCSR "DONE" DOES NOT SET
2200	004604	000405			BR	ID	;BR TO END OF TEST
2201	004606				5#:		
	004606	004737	015714		JSR	FC,#ATY4	;; ONLY REPORT A FATAL ERROR
	004612	000004			4		;; THE ERROR NUMBER (FROM APT LIST)
2202	004614	000000			HALT		
2203	004616	000427			BR	ENDB7	;** BR TO NEXT TEST,
2204							;** AND SKIP THE TYPEOUT THAT FOLLOWS
2205							;** BECAUSE OF THIS FAILURE
2206							
2207	004620				ID:		
2208	004620	042777	000004	176032	BIC	#BIT2,BCTCSR	;** DISABLE MAINTENANCE MODE FOR
							;** CONSOLE TO ALLOW FOR COMMUNICATION
							;** WITH TERMINAL.
2209	004626	023737	000042	000046	CMP	B#42,B#46	;UNDER ACT11?
2210	004634	001412			BEQ	6#	;IF YES, SKIP IDENT. TYPEOUT
2211	004636	005737	001074		TST	#PASS	;IS THIS THE FIRST PASS?
2212	004642	001007			BNE	6#	;IF NOT BR TO NEXT TEST & SKIP THE IDENTIFICATION TYPEOUTS
2213	004644	005737	001076		TST	#DEVCT	;IS THIS THE FIRST SUBPASS?
2214	004650	001004			BNE	6#	;IF NOT, BR TO NEXT TEST

2215	004652	104401				TYPE			;TYPE PROGRAM IDENTIFICATION
2216	004654	025004				M1			
2217	004656	104401				TYPE			;TYPE NUMBER OF DEVICES UNDER TEST
2218	004660	025044				M2			
2219	004662	032777	000020	174150	64:	BIT	#BIT4,RSWR		;CLOCK TEST ONLY?
2220	004670	001402				BEG	ENDB7		;** BR IF NOT
2221	004672	000137	005536			JMP	TCLOCK		;ELSE, JUMP TO TEST CLOCK
2222	004676				ENDB7:				
2223	004676	005000				CLR	R0		;** DELAY ENOUGH TIME TO ALLOW ANY CHARACTERS
	004700	012701	000002			MOV	#2,R1		;** THAT MIGHT BE IN THE PROCESS OF BEING
	004704	005300			404:	DEC	R0		;** TRANSMITTED TO FINISH BEFORE MAINTENANCE
	004706	001376				BNE	404		;** WRAP FOR THE UART UNDER TEST IS DISABLED.
	004710	005301				DEC	R1		;** THIS WILL INHIBIT ANY COMMUNICATION
	004712	001374				BNE	404		;** TO HARDWARE MEDIA SUCH AS TUSB THAT MIGHT
									;** BE ATTACHED TO UART UNDER TEST.

2224

.SBTTL TEST # 5 - TEST THAT TCSR "DONE" SETS WITH RESET

 ;*TEST 5 TEST THAT TCSR "DONE" SETS WITH RESET
 ;*****

004714 000004
 2225 004716 052777 000004 175714
 2226 004724 005077 175712
 2227 004730 105777 175704
 2228 004734 100375
 2229 004736 005077 175700
 2230 004742 000240
 2231 004744 000005
 2232 004746 105777 175666
 2233 004752 100401
 2234
 2235 004754 104005
 2236
 2237 004756 000240
 2238
 2239

TST5: SCOPE
 BIS @BIT2,@TCSR ;** ENABLE MAINT. WRAP
 CLR @TBUF ;LOAD TRANSMIT BUFFER
 10: TSTB @TCSR ;WAIT FOR DONE
 BPL 10
 CLR @TBUF ;LOAD SECOND BUFFER
 NOP
 RESET ;CLEAR DONE WITH RESET
 TSTB @TCSR ;CHECK FOR DONE SET
 BMI 10 ;** BR TO NEXT TEST IF DONE SET
 ERROR *5 ;TCSR "DONE" DOES NOT SET WITH RESET
 10: NOP ;**

2240

.SBTTL TEST # 6 - TEST ABILITY TO ACCESS RCSR
;*****
;*TEST 6 TEST ABILITY TO ACCESS RCSR
;*****
TST6: SCOPE
MOV @R3 ;SAVE TIMEOUT VECTOR
MOV @R4 ;SET UP TIMEOUT VECTOR
TST @RCSR ;ACCESS RCSR
BR 2 ;BR TO END OF TEST

1: CMP (SP)+,(SP)+ ;RESTORE SP AFTER TIMEOUT
ERROR +6 ;CAN NOT ACCESS RCSR
2: MOV R3,@R4 ;RESTORE TIMEOUT VECTOR

004760 000004
2241 004762 013703 000004
2242 004766 012737 005002 000004
2243 004774 005777 175634
2244 005000 000402
2245
2246 005002 022626
2247 005004 104006
2248 005006 010337 000004

2249

```
.SBTTL TEST # 7 - TEST ABILITY TO ACCESS RBUF  
;*****  
;*TEST 7 TEST ABILITY TO ACCESS RBUF  
;*****  
TST7: SCOPE  
MOV R4,R3 ;SAVE TIMEOUT VECTOR  
MOV #1,R4 ;SET UP TIMEOUT VECTOR  
TST RBUF ;ACCESS RBUF  
BR 2 ;BR TO END OF TEST  
  
1: CMP (SP)+,(SP)+ ;RESTORE SP AFTER TIMEOUT  
ERROR +7 ;CAN NOT ACCESS RBUF  
2: MOV R3,R4 ;RESTORE TIMEOUT VECTOR
```

```
005012 000004  
2250 005014 013703 000004  
2251 005020 012737 005034 000004  
2252 005026 005777 175604  
2253 005032 000402  
2254  
2255 005034 022626  
2256 005036 104007  
2257 005040 010337 000004
```


2303

.SBTTL TEST # 11 - TEST THAT BIT6(XMIT INT EN) CAN BE SET & RESET
 ;:TEST 11 TEST THAT BIT6(XMIT INT EN) CAN BE SET & RESET

```

TST11: SCOPE
        RESET                ;CLEAR EVERYTHING
        MOV      BTVECT,R3    ;SAVE XMIT VECTOR
        MOV      #16,BTVECT  ;SET UP INTERRUPT VECTOR FOR ERROR REPORT
        JSR     PC,WPSW      ;SET PSM TO PRIORITY=7
                                .WORD 340
        BIT     @BIT6,@TCSR   ;TEST BIT6 OF TCSR
        BEQ     24            ;BR IF ZERO
        ERROR   *21          ;BIT6 IN TCSR NOT CLEAR AFTER RESET
        BR      24
        CMP     (SP),.(SP).  ;RESTORE SP AFTER INTERRUPT
        ERROR   *22
                                ;XMIT INTERRUPT OCCURRED PRIO=7
        BIS     @BIT6,@TCSR   ;SET BIT6 OF TCSR
        BIT     @BIT6,@TCSR   ;TEST BIT6 OF TCSR
        BNE     34            ;BR, IF SET
        ERROR   *23
                                ;CANNOT SET BIT6 OF TCSR
        BIC     @BIT6,@TCSR   ;CLEAR BIT6 OF TCSR
        BIT     @BIT6,@TCSR   ;TEST BIT6 OF TCSR
        BEQ     44            ;BR IF CLEAR
        ERROR   *24
                                ;CANNOT CLEAR BIT6 OF TCSR
        BIS     @BIT6,@TCSR   ;SET BIT6 OF TCSR
        RESET   ;CLEAR BIT6 WITH RESET
        BIT     @BIT6,@TCSR   ;TEST BIT6 OF TCSR
        BEQ     54            ;BR IF CLEAR
        ERROR   *25
                                ;CANNOT CLEAR BIT6 OF TCSR WITH RESET
        MOV     R3,BTVECT    ;RESTORE XMIT VECTOR
    
```


2379

.SBTTL TEST # 13 - TEST ABILITY TO ACCESS LKS
 ;*****
 ;*TEST 13 TEST ABILITY TO ACCESS LKS
 ;*****

2380	005544	000004			TST13: SCOPE	
	005546	005737	002624		TST CTSTFL	;IS CONSOLE UNDER TEST?
	005552	001420			BEQ TST14	;IF NOT, SKIP THIS TEST
	005554	032777	000100	173256	BIT #BIT6,BSWR	;ARE LINE CLOCK TESTS INHIBITED?
	005562	001014			BNE TST14	;IF YES, SKIP THIS TEST
2381	005564	013703	000004		MOV #R3	;SAVE TIMEOUT VECTOR
2382	005570	012737	005604	000004	MOV #10,R04	;SET UP TIMEOUT VECTOR
2383	005576	005777	175072		TST BLKS	;ACCESS LKS
2384	005602	000402			BR 24	;NO TIMEOUT - BR TO END OF TEST
2385						
2386	005604	022626			14: CMP (SP),.(SP),	;RESTORE SP AFTER TIMEOUT
2387	005606	104010			ERROR +10	;CAN NOT ACCESS LKS
2388						
2389	005610	010337	000004		24: MOV R3,R04	;RESTORE TIMEOUT VECTOR
2390						

2428

.SBTTL TEST # 15 - TEST FOR DUAL ADDRESSING OF REGISTERS
 ;*****
 ;*TEST 15 TEST FOR DUAL ADDRESSING OF REGISTERS
 ;*****
 TST15: SCOPE

2429	005764	000004					MOV	R3,R3	;	SAVE TIMEOUT VECTOR
2430	005772	013704	000006				MOV	R4,R4	;	SAVE TIMEOUT PSW
2431	005776	012737	006130	000004			MOV	R4,R4	;	SET UP TIMEOUT VECTOR
2432	006004	012737	000340	000006			MOV	R340,R06	;	KEEP PRIO=7
2433	006012	000005					RESET		;	CLEAR EVERYTHING
2434	006014	012700	000002				MOV	RBIT1,R0	;	SET UP BIT MASK
2435	006020	032777	000020	173012			BIT	RBIT4,BSHR	;	CLOCK TEST ONLY?
2436	006026	001404					BEQ	1	;	BR IF NOT
2437	006030	013737	002674	001020			MOV	LKS,RGDADR	;	ELSE, MOVE GOOD LKS ADDRESS INTO RGDADR
2438	006036	000403					BR	2		
2439	006040	013737	002634	001020	1:		MOV	RCSR,RGDADR	;	MOVE GOOD RCSR ADDRESS INTO RGDADR
2440	006046	013737	001020	001022	2:		MOV	RGDADR,RBDADR	;	MOVE GOOD ADDRESS INTO TEST ADDRESS LOCATION
2441	006054	040037	001022				BIC	R0,RBDADR	;	CREATE BAD ADDRESS BY COMPLEMENTING ONE BIT
2442	006060	023737	001020	001022			CMR	RGDADR,RBDADR	;	ARE ADDRESSES IDENTICAL?
2443	006066	001002					BNE	3	;	IF NOT, TEST THIS ADDRESS
2444	006070	050037	001022				BIS	R0,RBDADR	;	ELSE, BIT SET THIS BIT POSITION TO GENERATE BAD ADDRESS
2445	006074	017737	172722	001024	3:		MOV	RBDADR,RGDADR	;	SAVE CONTENTS OF BAD ADDRESS IF IT EXISTS
2446	006102	052777	000100	172712			BIS	RBIT6,RBDADR	;	SET BIT6 USING BAD ADDRESS
2447	006110	032777	000100	172702			BIT	RBIT6,RGDADR	;	CHECK TO SEE IF GOOD ADDRESS CONTAINS BIT6
2448	006116	001011					BNE	6	;	BR IF SET ---> ERROR
2449	006120	013777	001024	172674			MOV	RGDADR,RBDADR	;	RESTORE ANY MEMORY LOCATION THAT WAS ALTERED
2450	006126	000401					BR	5	;	BR TO CONTINUE TEST
2451	006130	022626			4:		CMR	(RSP),RSP	;	RESTORE SP AFTER TIMEOUT
2452	006132	006300			5:		ASL	R0	;	SHIFT BIT MASK TO NEXT POSITION
2453	006134	105700					TSTB	R0	;	COMPLEMENTED ALL BITS FROM 1 - 7? ;:~
C	2454	006136	100343				BPL	2	;	BR, IF NOT. ;:~
C	2455	006140	000401				BR	7	;	BR TO NEXT TEST
	2456									
	2457	006142	104040		6:		ERROR	+40	;	DUAL ADDRESSING ERROR
	2458								;	RBDADR = DUAL ADDRESS
	2459								;	RGDADR = GOOD ADDRESS
	2460									
	2461	006144	010337	000004	7:		MOV	R3,R3	;	RESTORE TIMEOUT VECTOR
	2462	006150	010437	000006			MOV	R4,R4	;	RESTORE TIMEOUT PSW

2463

.SBTTL TEST # 16 - TEST THAT BIT7 OF LKS SETS & CAN BE CLEARED
 ;*****
 ;*TEST 16 TEST THAT BIT7 OF LKS SETS & CAN BE CLEARED
 ;*****

2464	006154	000004				TST16: SCOPE		
	006156	005737	002624			TST	CTSTFL	; IS CONSOLE UNDER TEST?
	006162	001437				BEQ	TST17	; IF NOT, SKIP THIS TEST
	006164	032777	000100	172646		BIT	@BIT6,BSWR	; ARE LINE CLOCK TESTS INHIBITED?
	006172	001033				BNE	TST17	; IF YES, SKIP THIS TEST
2465	006174	000005				PSET		; CLEAR EVERYTHING & SET BIT7 OF LKS
2466	006176	105777	174472		14:	" 0	BLKS	; TEST FOR BIT7 OF LKS
2467	006202	100401				BR	24	; BR IF SET
2468								
2469	006204	104041				ERROR	+41	; BIT7 OF LKS DID NOT SET WITH RESET
2470								
2471	006206	042777	000200	174460	24:	BIC	@BIT7,BLKS	; CLEAR BIT7 OF LKS
2472	006214	032777	000200	174452		BIT	@BIT7,BLKS	; TEST BIT7 OF LKS
2473	006222	001410				BEQ	34	
2474	006224	042777	000200	174442		BIC	@BIT7,BLKS	; TRY ONE MORE TIME BECAUSE THE CLOCK
2475	006232	032777	000200	174434		BIT	@BIT7,BLKS	; MAY HAVE SET IMMEDIATELY AFTER THE FIRST CLEAR
2476	006240	001401				BEQ	34	
2477								
2478	006242	104042				ERROR	+42	; CAN NOT CLEAR BIT7 OF LKS
2479								
2480	006244	005000			34:	CLR	RO	; CLEAR TIMER
2481	006246	105777	174422		CONT:	TSTB	BLKS	; TEST FOR BIT7 OF LKS
2482	006252	100403				BMI	TST17	; BR, IF SET
2483	006254	005200				INC	RO	; INCREMENT TIMER
2484	006256	001373				BNE	CONT	; CONTINUE UNTIL TIME EXPIRES
2485								
2486	006260	104043				ERROR	+43	; BIT7 OF LKS DOES NOT SET

```

2487 .SBTTL TEST # 17 - TEST THAT THE REAL TIME CLOCK INTERRUPTS PROPERLY
;*****
;TEST 17 TEST THAT THE REAL TIME CLOCK INTERRUPTS PROPERLY
;*****
TST17: SCOPE
2488 006262 000004          TST      CTSTFL      ;IS CONSOLE UNDER TEST?
006264 005737 002624      BEQ      TST20      ;IF NOT, SKIP THIS TEST
006270 001514              BIT      #BIT6,BSWR ;ARE LINE CLOCK TESTS INHIBITED?
006272 032777 000100 172540 BNE      TST20      ;IF YES, SKIP THIS TEST
006300 001110              JSR      PC,WRPSW   ;SET PSW TO PRIORITY 7
2489 006302 004737 014502      .WORD   340
2490 006306 000340          MOV      BRTCVT,R3  ;SAVE LINE CLOCK VECTOR
2491 006310 017703 174362      MOV      BRTCPSW,R4 ;SAVE LINE CLOCK PSW VECTOR
2492 006314 017704 174360      MOV      #3#,BRTCVT ;SET RTC INTERRUPT VECTOR TO ERROR REPORT
2493 006320 012777 006400 174350 MOV      #340,BRTCPSW ;KEEP PRIORITY AT 7
2494 006326 012777 000340 174344 CLR      BLKS       ;CLEAR DONE FLAG AND
2495 006334 005077 174334          MOV      #BIT6,BLKS ;SET CLOCK INTERRUPT ENABLE
2496 006340 012777 000100 174326 1#:      TSTB    BLKS       ;WAIT FOR RTC DONE(INTERRUPT REQUEST)
2497 006346 105777 174322          BPL     1#
2498 006352 100375              CLR     BLKS       ;CLEAR DONE FLAG AND
2499 006354 005077 174314          MOV      #BIT6,BLKS ;SET CLOCK INTERRUPT ENABLE
2500 006360 012777 000100 174306 2#:      TSTB    BLKS       ;WAIT FOR RTC DONE(INTERRUPT REQUEST)
2501 006366 105777 174302          BPL     2#
2502 006372 100375              NOP
2503 006374 000240          BR      4#
2504 006376 000402          .GIVE TIME FOR ANY INTERRUPTS
2505          .BR, IF NO INTERRUPT OCCURS
2506 006400 022626          3#:      CMP      (SP)+,(SP)+ ;RESTORE SP AFTER INTERRUPT
2507 006402 104044          ERROR   +44        ;RTC INTERRUPTS AT PRIORITY 7
2508
2509 006404 005077 174264          4#:      CLR      BLKS       ;DISABLE RTC INTERRUPTS & CLEAR DONE
2510 006410 012777 006440 174260 MOV      #5#,BRTCVT ;SET RTC INTERRUPT VECTOR FOR ERROR
2511 006416 004737 014502          JSR      PC,WRPSW   ;CHANGE PSW TO PRIORITY 5
2512 006422 000240          .WORD   240
2513 006424 105777 174244          20#:     TSTB    BLKS       ;WAIT FOR DONE (INTERRUPT REQUEST)
2514 006430 100375              BPL     20#
2515 006432 000240          NOP
2516 006434 000240          NOP
2517 006436 000402          BR      6#
2518
2519 006440 022626          5#:      CMP      (SP)+,(SP)+ ;RESTORE SP AFTER INTERRUPT
2520 006442 104045          ERROR   +45        ;RTC INTERRUPTS WITH INTERRUPTS DISABLED
2521
2522 006444 012777 006502 174224          6#:      MOV      #8#,BRTCVT ;POINT RTC VECTOR TO END OF TEST
2523 006452 042777 000200 174214 BIC      #BIT7,BLKS  ;CLEAR CLOCK DONE FLAG
2524 006460 052777 000100 174206 BIS      #BIT6,BLKS  ;ALLOW INTERRUPTS
2525 006466 105777 174202          7#:      TSTB    BLKS       ;WAIT FOR RTC DONE
2526 006472 100375              BPL     7#
2527 006474 000240          NOP
2528          .GIVE TIME FOR INTERRUPT
2529 006476 104046          ERROR   +46        ;RTC INTERRUPT DID NOT OCCUR
2530 006500 000401          BR      9#
2531 006502 022626          8#:      CMP      (SP)+,(SP)+ ;RESTORE SP AFTER INTERRUPT
2532 006504 042777 000100 174162          9#:      BIC      #BIT6,BLKS  ;DISABLE INTERRUPTS
2533 006512 010377 174160          MOV      R3,BRTCVT ;RESTORE LINE CLOCK VECTOR
2534 006516 010477 174156          MOV      R4,BRTCPSW ;RESTORE LINE CLOCK PSW VECTOR
    
```


2599

.SBTTL TEST # 22 - TEST THAT RTC INTERRUPT CLEARS BY CLEARING BIT7 OF LKS
 ;*****
 ;*TEST 22 TEST THAT RTC INTERRUPT CLEARS BY CLEARING BIT7 OF LKS
 ;*****

2600	007054	000004			TST22: SCOPE		
	007056	005737	002624		TST	CTSTFL	;IS CONSOLE UNDER TEST?
	007062	001457			BEQ	TST23	;IF NOT, SKIP THIS TEST
	007064	032777	000100	171746	BIT	#BIT6,BSWR	;ARE LINE CLOCK TESTS INHIBITED?
	007072	001053			BNE	TST23	;IF YES, SKIP THIS TEST
2601	007074	004737	014502		JSR	PC,WRPSW	;SET PRIORITY TO 7
2602	007100	000340				.WORD 340	
2603	007102	017703	173570		MOV	BRTCVT,R3	;SAVE LINE CLOCK VECTOR
2604	007106	012777	007204	173562	MOV	#34,BRTCVT	;POINT RTC VECTOR TO ERROR REPORT
2605	007114	005077	173554		CLR	BLKS	;CLEAR DONE FLAG AND
2606	007120	012777	000100	173546	MOV	#BIT6,BLKS	;SET CLOCK INTERRUPT ENABLE
2607	007126	105777	173542		14: TSTB	BLKS	;WAIT FOR DONE (INTERRUPT REQUEST)
2608	007132	100375			BPL	14	
2609	007134	005077	173534		CLR	BLKS	;CLEAR DONE FLAG AND
2610	007140	012777	000100	173526	MOV	#BIT6,BLKS	;SET CLOCK INTERRUPT ENABLE
2611	007146	105777	173522		24: TSTB	BLKS	;WAIT FOR DONE (INTERRUPT REQUEST)
2612	007152	100375			BPL	24	
2613	007154	000240			NOP		;GIVE TIME FOR ANY INTERRUPT
2614	007156	042777	000200	173510	BIC	#BIT7,BLKS	;CLEAR DONE & INTERRUPT
2615	007164	004737	014502		JSR	PC,WRPSW	;ALLOW INTERRUPTS
2616	007170	000240				.WORD 240	
2617	007172	000240			NOP		;GIVE TIME FOR ANY INTERRUPT
2618	007174	042777	000100	173472	BIC	#BIT6,BLKS	;DISALLOW INTERRUPTS
2619	007202	000402			BR	44	;BR TO END OF TEST
2620							
2621							
2622	007204	022626			34: CMP	(SP)+,(SP)+	;RESTORE SP AFTER INTERRUPT
2623	007206	104052			ERROR	+52	;CLEARING BIT7 OF LKS DID NOT CLEAR INTERRUPT
2624							
2625	007210	010377	173462		44: MOV	R3,BRTCVT	;RESTORE LINE CLOCK VECTOR
2626	007214	004737	014502		JSR	PC,WRPSW	;RESTORE PRIORITY TO 7
2627	007220	000340			.WORD	340	

2667

.SBTTL TEST # 24 - TEST THAT XMIT INTERRUPTS ONLY WHEN ENABLED
 ;*****
 ;*TEST 24 TEST THAT XMIT INTERRUPTS ONLY WHEN ENABLED
 ;*****

	007410	000004				TST24: SCOPE	
2668	007412	042777	000100	173220		BIC #BIT6,@TCSR	;CLEAR TRANSMIT INTERRUPT ENABLE
2669	007420	017703	173224			MOV @TVECT,R3	;SAVE XMIT VECTOR
2670	007424	012777	007450	173216		MOV #2,@TVECT	;POINT XMIT VECTOR TO ERROR REPORT
2671	007432	105177	173202		1: TSTB @TCSR		;WAIT FOR DONE
2672	007436	100375				BPL 1:	
2673	007440	004737	014502			JSR PC,WRPSW	;SET PSM TO PRIORITY 3
2674	007444	000140				.WORD 140	
2675	007446	000402				BR 3:	
2676							
2677	007450	022626			2: CMP (SP),.(SP),		;RESTORE SP AFTER INTERRUPT
2678	007452	104054				ERROR +54	
2679							;XMIT INTERRUPTS WITH INTERRUPT ENABLE CLEAR
2680	007454	012777	007504	173166	3: MOV #4,@TVECT		;SET XMIT VECTOR TO END OF TEST
2681	007462	052777	000100	173150		BIS #BIT6,@TCSR	;ENABLE INTERRUPTS
2682	007470	000240				NOP	;ALLOW
2683	007472	000240				NOP	;TIME
2684	007474	000240				NOP	;FOR
2685	007476	000240				NOP	;INTERRUPT
2686	007500	104055				ERROR +55	;XMIT DID NOT INTERRUPT
2687	007502	000401				BR 5:	;BRANCH OVER STACK CORRECTION
2688	007504	022626			4: CMP (SP),.(SP),		;RESTORE SP AFTER INTERRUPT
2689	007506	042777	000100	173124	5: BIC #BIT6,@TCSR		;DISABLE INTERRUPTS
2690	007514	010377	173130			MOV R3,@TVECT	;RESTORE XMIT VECTOR

2723

.SBTTL TEST # 26 - TEST TRANSMITTER FOR DOUBLE INTERRUPTS
 ;
 ;
 ;TEST 26 TEST TRANSMITTER FOR DOUBLE INTERRUPTS
 ;
 ;

2724	007646	000004				TST26: SCOPE		
2724	007650	042777	000100	172762		BIC	0BIT6,0TCSR	;CLEAR INTERRUPT ENABLE
2725	007656	017703	172766			MOV	0TVECT,R3	;SAVE XMIT VECTOR
2726	007662	017704	172764			MOV	0TPSW,R4	;SAVE XMIT PSW VECTOR
2727	007666	012777	007740	172754		MOV	020,0TVECT	;SET UP XMIT VECTOR
2728	007674	012777	000340	172750		MOV	0340,0TPSW	;SET PIO 7 AFTER INTERRUPT
2729	007702	004737	014502			JSR	PC,MRPSW	;SET PSW TO PRIORITY 3
2730	007706	000140					.WORD 140	
2731	007710	105777	172724		18:	TSTB	0TCSR	;WAIT FOR DONE
2732	007714	100375				BPL	18	
2733	007716	052777	000100	172714		BIS	0BIT6,0TCSR	;ENABLE INTERRUPTS
2734	007724	000240				NOP		;ALLOW
2735	007726	000240				NOP		;TIME
2736	007730	000240				NOP		;FOR
2737	007732	000240				NOP		;INTERRUPT
2738								
2739	007734	104060				ERROR	*60	;XMIT INTERRUPT DID NOT OCCUR
2740	007736	000401				BR	258	;BRANCH OVER STACK CORRECTION INTERRUPT
2741	007740	022626			24:	CMP	(SP), (SP)	;RESTORE SP AFTER INTERRUPT
2742	007742	012777	007776	172700	25:	MOV	048,0TVECT	;POINT XMIT VECTOR TO ERROR
2743	007750	004737	014502			JSR	PC,MRPSW	;SET PSW TO PRIORITY 3
2744	007754	000140					.WORD 140	
2745	007756	000240				NOP		;ALLOW
2746	007760	000240				NOP		;TIME
2747	007762	000240				NOP		;FOR
2748	007764	000240				NOP		;INTERRUPT
2749	007766	042777	000100	172644		BIC	0BIT6,0TCSR	;DISABLE INTERRUPTS
2750	007774	000402				BR	58	;BR TO END OF TEST
2751								
2752	007776	022626			44:	CMP	(SP), (SP)	;RESTORE SP AFTER INTERRUPT
2753	010000	104061				ERROR	*61	
2754								;XMIT RE-INTERRUPTED
2755	010002	010377	172642		54:	MOV	R3,0TVECT	;RESTORE XMIT VECTOR
2756	010006	010477	172640			MOV	R4,0TPSW	;RESTORE XMIT PSW VECTOR
2757								

2758

.SBTTL TEST # 27 - TEST THAT XMIT INTERRUPT CLEARS WITH LOADING TBUF

;;
 ;*TEST 27 TEST THAT XMIT INTERRUPT CLEARS WITH LOADING TBUF
 ;;

TST27: SCOPE

2759	010012	000004							
2760	010014	042777	000100	172616	BIC	#BIT6,#TCSR		;	DISABLE INTERRUPTS
2761	010022	004737	014502		JSR	PC,#RPSM		;	SET PSM TO PRIORITY 7
2762	010026	000340				.WORD	340		
2763	010030	017703	172614		MOV	#TVECT,R3		;	SAVE XMIT VECTOR
2764	010034	012777	010122	172606	MOV	#2,#TVECT		;	POINT XMIT VECTOR TO ERROR
2765	010042	052777	000004	172570	BIS	#BIT2,#TCSR		;	ENABLE MAINT. WRAP
2766	010050	052777	000100	172562	BIS	#BIT6,#TCSR		;	ENABLE INTERRUPTS
2767	010056	005077	172560		CLR	#TBUF		;	LOAD TBUF
2768	010062	105777	172552		14: TSTB	#TCSR		;	WAIT FOR DONE (INTERRUPT)
2769	010066	100375			BPL	14			
2770	010070	005077	172546		CLR	#TBUF		;	FILL SECOND BUFFER TO RESET INT.
2771	010074	004737	014502		JSR	PC,#RPSM		;	ALLOW INTERRUPTS
2772	010100	000140				.WORD	140		
2773	010102	000240			NOP			;	ALLOW
2774	010104	000240			NOP			;	TIME
2775	010106	000240			NOP			;	FOR
2776	010110	000240			NOP			;	INTERRUPT
2777	010112	042777	000100	172520	BIC	#BIT6,#TCSR		;	DISABLE INTERRUPTS
2778	010120	000405			BR	34		;	BR TO END OF TEST
2779	010122	022626			24: CMP	(SP), (SP)		;	RESTORE SP AFTER INTERRUPT
2780	010124	042777	000004	172526	BIC	#BIT2,#TCSR		;	DISABLE MAINTENANCE MODE FOR
								;	CONSOLE TO ALLOW FOR COMMUNICATION
								;	WITH TERMINAL.
2781	010132	104062			ERROR	+62			
2782									
2783	010134	010377	172510		34: MOV	R3,#TVECT		;	LOADING TBUF DID NOT CLEAR INTERRUPT.
2784								;	RESTORE XMIT VECTOR
2785	010140	005000			CLR	R0		;	DELAY ENOUGH TIME TO ALLOW ANY CHARACTERS
	010142	012701	000002		MOV	#2,R1		;	THAT MIGHT BE IN THE PROCESS OF BEING
	010146	005300			40: DEC	R0		;	TRANSMITTED TO FINISH BEFORE MAINTENANCE
	010150	001376			BNE	404		;	WRAP FOR THE UART UNDER TEST IS DISABLED.
	010152	005301			DEC	R1		;	THIS WILL INHIBIT ANY COMMUNICATION
	010154	001374			BNE	404		;	TO HARDWARE MEDIA SUCH AS TUSB THAT MIGHT
								;	BE ATTACHED TO UART UNDER TEST.

2820	010346	052777	000004	172264	RCVDON:	BIS	#BIT2,BTCR	;SET MAINTENCE WRAP
2821	010354	005000				CLR	RO	;CLEAR TIMER
2822	010356	005077	172260			CLR	#TBUF	;LOAD TRANSMIT BUFFER
2823	010362	105777	172246		WDONE:	TSTB	#RCSR	;CHECK FOR RECEIVER DONE
2824	010366	100406				BMI	5:	;BR, IF DONE
2825	010370	005200				INC	RO	;INCREMENT TIMER, IF NOT DONE
2826	010372	001373				BNE	WDONE	;CONTINUE WAIT IF TIME REMAINS
2827	010374	042777	000004	172256		BIC	#BIT2,BTCR	;** DISABLE MAINTENANCE MODE FOR
								;** CONSOLE TO ALLOW FOR COMMUNICATION
								;** WITH TERMINAL.
2828	010402	104064				ERROR	+64	
2829								;RECEIVER DONE NEVER SET
2830								
2831	010404	032737	000001	003002	5:	BIT	#BIT0,FLAG44	;** 11/44??
2832	010412	001010				BNE	6:	;** DO NOT EXECUTE THIS SECTION
2833	010414	032777	004000	172212		BIT	#BIT11,BRCSR	;CHECK FOR RCVR ACTIVE CLEAR
2834	010422	001404				BEQ	6:	;BR, IF CLEAR
2835	010424	042777	000004	172226		BIC	#BIT2,BTCR	;** DISABLE MAINTENANCE MODE FOR
								;** CONSOLE TO ALLOW FOR COMMUNICATION
								;** WITH TERMINAL.
2836	010432	104065				ERROR	+65	
2837								;RCVR ACTIVE DID NOT CLEAR WITH RCVR DONE
2838								
2839	010434	000005			6:	RESET		;CLEAR DONE WITH RESET
2840	010436	105777	172172			TSTB	#RCSR	;CHECK FOR DONE CLEAR
2841	010442	001404				BEQ	7:	
2842								
2843	010444	042777	000004	172206		BIC	#BIT2,BTCR	;** DISABLE MAINTENANCE MODE FOR
								;** CONSOLE TO ALLOW FOR COMMUNICATION
								;** WITH TERMINAL.
2844	010452	104066				ERROR	+66	
2845								;RESET DID NOT CLEAR RCVR DONE
2846								
2847	010454				7:			
2848	010454	042777	000004	172176		BIC	#BIT2,BTCR	;** DISABLE MAINTENANCE MODE FOR
								;** CONSOLE TO ALLOW FOR COMMUNICATION
								;** WITH TERMINAL.

2849

.SBTTL TEST # 31 - TEST THAT READING RBUF CLEARS RECEIVER DONE
 ;*****
 ;*TEST 31 TEST THAT READING RBUF CLEARS RECEIVER DONE
 ;*****
 TST31: SCOPE

2850 010462 000004
 2850 010464 000005
 2851 010466 052777 000004 172144
 2852 010474 005000
 010476 012701 000002

RESET
 BIS #BIT2,@TCSR
 CLR R0
 MOV #2,R1

;CLEAR EVERYTHING
 ;SET MAINTENANCE WRAP
 ;** DELAY ENOUGH TIME TO ALLOW ANY CHARACTERS
 ;** THAT MIGHT BE IN THE PROCESS OF BEING
 ;** RECEIVED TO FINISH AFTER MAINTENANCE
 ;** WRAP FOR THE UART UNDER TEST IS ENABLED.
 ;** READ TO CLEAR DONE

010502 105777 172130
 010506 005300
 010510 001374
 010512 005301
 010514 001372

42: TSTB BRBUF
 DEC R0
 BNE 42:
 DEC R1
 BNE 42:

;**
 ;**
 ;**
 ;**

2853 010516 005077 172120
 2854 010522 105777 172106
 2855 010526 100375
 2856 010530 017700 172102
 2857 010534 042777 000004 172116

1: TSTB BRCSR
 BPL 1:
 MOV BRBUF,R0
 BIC #BIT2,@TCSR

;LOAD TRANSMITTER
 ;WAIT FOR RECEIVER DONE
 ;READ RECEIVE BUFFER
 ;** DISABLE MAINTENANCE MODE FOR
 ;** CONSOLE TO ALLOW FOR COMMUNICATION
 ;** WITH TERMINAL.

2858 010542 105777 172066
 2859 010546 001401
 2860 010550 104070
 2861
 2862 010552 000240
 2863

TSTB BRCSR
 BEQ 10:
 ERROR +70
 10: NOP

;CHECK FOR RECEIVE DONE CLEAR
 ;** BR, IF CLEAR TO NEXT TEST
 ;READING RBUF DID NOT CLEAR RCVR DONE

2864

.SBTTL TEST # 32 - TEST THAT RDR ENABLE CLEARS RECEIVER DONE FLAG

 ;*TEST 32 TEST THAT RDR ENABLE CLEARS RECEIVER DONE FLAG
 ;*****

	010554	000004			TST32: SCOPE		
2865	010556	032737	000001	003002	BIT	#BIT0,FLAG44	;** 11/44 ??
2866	010564	001044			BNE	10#	;** YES DO NOT EXECUTE THIS TEST
2867	010566	000005			RESET		;CLEAR EVERYTHING
2868	010570	052777	000001	172036	BIS	#BIT0,BRCR	;SET RDR ENABLE
2869	010576	052777	000004	172034	BIS	#BIT2,BTCR	;SET MAINTENANCE WRAP
2870	010604	005000			CLR	R0	;** DELAY ENOUGH TIME TO ALLOW ANY CHARACTERS
	010606	012701	000002		MOV	#2,R1	;** THAT MIGHT BE IN THE PROCESS OF BEING
							;** RECEIVED TO FINISH AFTER MAINTENANCE
							;** WRAP FOR THE UART UNDER TEST IS ENABLED.
							;** READ TO CLEAR DONE
	010612	105777	172020		42#:	TSTB	BRBUF
	010616	005300				DEC	R0
	010620	001374				BNE	42#
	010622	005301				DEC	R1
	010624	001372				BNE	42#
2871	010626	005077	172010			CLR	BTBUF
2872	010632	105777	171776		1#:	TSTB	BRCR
2873	010636	100375				BPL	1#
2874	010640	032777	000001	171766		BIT	#BIT0,BRCR
2875	010646	001401				BEQ	2#
2876							
2877	010650	104117				ERROR	+117
2878							
2879	010652	052777	000001	171754	2#:	BIS	#BIT0,BRCR
2880	010660	105777	171750			TSTB	BRCR
2881	010664	001404				BEQ	10#
2882	010666	042777	000004	171764		BIC	#BIT2,BTCR
							;** DISABLE MAINTENANCE MODE FOR
							;** CONSOLE TO ALLOW FOR COMMUNICATION
							;** WITH TERMINAL.
2883	010674	104067				ERROR	+67
2884							
2885	010676	000240			10#:	NOP	
							;SETTING RDR ENABLE DID NOT CLEAR RCVR DONE
							;**

2886

.SBTTL TEST # 33 - TEST THAT RCVR INTERRUPTS ONLY WHEN ENABLED
 ;*****
 ;TEST 33 TEST THAT RCVR INTERRUPTS ONLY WHEN ENABLED
 ;*****
 TST33: SCOPE

2887	010700	000004								
2888	010702	042777	000100	171730		BIC	#BIT6,BTCR		;DISABLE TRANSMIT INTERRUPTS	
2889	010710	042777	000100	171716		BIC	#BIT6,BRCR		;DISABLE RECEIVER INTERRUPTS	
2890	010716	052777	000004	171714		BIS	#BIT2,BTCR		;SET MAINTENANCE WRAP	
	010724	005000				CLR	R0		;** DELAY ENOUGH TIME TO ALLOW ANY CHARACTERS	
	010726	012701	000002			MOV	#2,R1		;** THAT MIGHT BE IN THE PROCESS OF BEING	
									;** RECEIVED TO FINISH AFTER MAINTENANCE	
									;** WRAP FOR THE UART UNDER TEST IS ENABLED.	
									;** READ TO CLEAR DONE	
	010732	105777	171700		42:	TSTB	BRBUF			
	010736	005300				DEC	R0			
	010740	001374				BNE	42:			
	010742	005301				DEC	R1			
	010744	001372				BNE	42:			
2891	010746	017703	171672			MOV	BRVECT,R3		;SAVE RECEIVE VECTOR	
2892	010752	012777	011010	171664		MOV	#2,BRVECT		;POINT RCV VECTOR TO ERROR REPORT	
2893	010760	004737	014502			JSR	PC,WRPSW		;SET PSW TO PRIORITY 3	
2894	010764	000140					.WORD 140			
2895	010766	005077	171650			CLR	BTBUF		;SEND A CHARACTER	
2896	010772	105777	171636		1:	TSTB	BRCR		;WAIT FOR RECEIVER DONE	
2897	010776	100375				BPL	1:			
2898	011000	042777	000004	171652		BIC	#BIT2,BTCR		;** DISABLE MAINTENANCE MODE FOR	
									;** CONSOLE TO ALLOW FOR COMMUNICATION	
									;** WITH TERMINAL.	
2899	011006	000405				BR	3:		;CONTINUE TEST	
2900										
2901	011010				2:					
2902	011010	042777	000004	171642		BIC	#BIT2,BTCR		;** DISABLE MAINTENANCE MODE FOR	
									;** CONSOLE TO ALLOW FOR COMMUNICATION	
									;** WITH TERMINAL.	
2903	011016	022626				CHP	(SP)+,(SP)+		;RESTORE SP AFTER INTERRUPT	
2904	011020	104071				ERROR	+71		;RECEIVER INTERRUPTS WITH INT. ENABLE CLEAR	
2905										
2906	011022	012777	011060	171614	3:	MOV	#4,BRVECT		;POINT RCV VECTOR TO END OF TEST	
2907	011030	052777	000100	171576		BIS	#BIT6,BRCR		;ENABLE RCV INTERRUPTS	
2908	011036	000240				NOP			;** GIVE ANY INTERRUPTS TIME	
2909	011040	000240				NOP			;** GIVE ANY INTERRUPTS TIME	
2910	011042	000240				NOP			;** GIVE ANY INTERRUPTS TIME	
2911	011044	000240				NOP			;** GIVE ANY INTERRUPTS TIME	
2912	011046	042777	000004	171604		BIC	#BIT2,BTCR		;** DISABLE MAINTENANCE MODE FOR	
									;** CONSOLE TO ALLOW FOR COMMUNICATION	
									;** WITH TERMINAL.	
2913	011054	104072				ERROR	+72		;RCVR DID NOT INTERRUPT	
2914	011056	000401				BR	5:		;BRANCH OVER STACK CORRECTION	
2915	011060	022626			4:	CHP	(SP)+,(SP)+		;RESTORE SP AFTER INTERRUPT	
2916	011062	042777	000100	171544	5:	BIC	#BIT6,BRCR		;DISABLE INTERRUPTS	
2917	011070	042777	000004	171562		BIC	#BIT2,BTCR		;** DISABLE MAINTENANCE MODE FOR	
									;** CONSOLE TO ALLOW FOR COMMUNICATION	
									;** WITH TERMINAL.	
2918	011076	010377	171542			MOV	R3,BRVECT		;RESTORE RECEIVE VECTOR	

2919

.SBTTL TEST # 34 - TEST THAT RCVR INTERRUPTS DO NOT OCCUR WHEN DISABLED
;*****
;*TEST 34 TEST THAT RCVR INTERRUPTS DO NOT OCCUR WHEN DISABLED
;*****

```

TST34: SCOPE
        RESET                               ;CLEAR EVERYTHING
        JSR    PC,WRPSW                      ;SET PSW TO PRIORITY 7
        .WORD 340
        BRVECT,R3                          ;SAVE RECEIVE VECTOR
        MOV    #2, BRVECT                   ;POINT RCVR VECTOR TO ERROR REPORT
        BIS    #BIT2, BCTCSR               ;SET MAINTENANCE WRAP
        CLR    R0                           ;** DELAY ENOUGH TIME TO ALLOW ANY CHARACTERS
        MOV    #2, R1                       ;** THAT MIGHT BE IN THE PROCESS OF BEING
        ;** RECEIVED TO FINISH AFTER MAINTENANCE
        ;** WRAP FOR THE UART UNDER TEST IS ENABLED.
        ;** READ TO CLEAR DONE
        ;**

        42: TSTB  BRBUF                      ;TEST FOR XMIT READY
        DEC    R0                           ;LOOP IF NOT
        BNE    42:                           ;SEND A CHARACTER
        DEC    R1                           ;WAIT FOR RECEIVER DONE
        BNE    42:                           ;**

        6:  TSTB  BRTPS                      ;ENABLE INTERRUPTS
        BPL    6:                            ;** GIVE TIME FOR INTERRUPT
        CLR    BRBUF                        ;** GIVE TIME FOR INTERRUPT
        ;** GIVE TIME FOR INTERRUPT
        ;** GIVE TIME FOR INTERRUPT
        ;** GIVE TIME FOR INTERRUPT
        BR    3:                             ;CONTINUE TEST

        1:  TSTB  BRCSR                      ;** DISABLE MAINTENANCE MODE FOR
        BPL    1:                            ;** CONSOLE TO ALLOW FOR COMMUNICATION
        BIS    #BIT6, BRCSR                 ;** WITH TERMINAL.
        NOP                                     ;RESTORE SP AFTER INTERRUPT
        NOP                                     ;RCVR INTERRUPTS AT PRIORITY 7
        NOP
        NOP
        BR    3:

        2:  BIC    #BIT2, BCTCSR             ;** DISABLE MAINTENANCE MODE FOR
        ;** CONSOLE TO ALLOW FOR COMMUNICATION
        ;** WITH TERMINAL.
        ;RESTORE SP AFTER INTERRUPT
        ;RCVR INTERRUPTS AT PRIORITY 7

        3:  BIC    (SP), (SP)                ;CLEAR INTERRUPT ENABLE
        ERROR  +73                           ;POINT RCVR VECTOR TO ERROR REPORT
        ;SET PSW TO PRIORITY 3
        BIC    #BIT6, BRCSR                 ;GIVE TIME FOR ANY INTERRUPT
        MOV    #4, BRVECT                   ;GIVE TIME FOR ANY INTERRUPT
        JSR    PC,WRPSW                      ;** DISABLE MAINTENANCE MODE FOR
        .WORD 140                            ;** CONSOLE TO ALLOW FOR COMMUNICATION
        ;** WITH TERMINAL.
        NOP                                     ;BR TO END OF TEST, IF NO INTERRUPT
        NOP
        BIC    #BIT2, BCTCSR

        5:  BR    5:

        4:  BIC    #BIT2, BCTCSR             ;** DISABLE MAINTENANCE MODE FOR
        ;** CONSOLE TO ALLOW FOR COMMUNICATION
        ;** WITH TERMINAL.
        ;RESTORE SP AFTER INTERRUPT
        ;RCVR INTERRUPT REQUEST PASSED WITH BIT6 CLEAR
        ;RESTORE RECEIVE VECTOR

        5:  MOV    R3, BRVECT

```

2958

.SBTTL TEST # 35 - TEST RECEIVER FOR DOUBLE INTERRUPTS
 ;*****
 ;*TEST 35 TEST RECEIVER FOR DOUBLE INTERRUPTS
 ;*****
 TST35:

011304	000004										
2959 011306	000005										
2960 011310	017703	171330									
2961 011314	017704	171326									
2962 011320	012777	011434	171316								
2963 011326	012777	000340	171312								
2964 011334	004737	014502									
2965 011340	000140										
2966 011342	052777	000004	171270								
2967 011350	005000										
011352	012701	000002									
011356	105777	171254		42:	TSTB	BRBUF					
011362	005300				DEC	R0					
011364	001374				BNE	42:					
011366	005301				DEC	R1					
011370	001372				BNE	42:					
2968 011372	005077	171244			CLR	BTBUF					
2969 011376	105777	171232		1:	TSTB	BRCSR					
2970 011402	100375				BPL	1:					
2971 011404	042777	000004	171246		BIC	BRIT2, BRCSR					
2972 011412	052777	000100	171214		BIS	BRIT6, BRCSR					
2973 011420	000240				NOP						
2974 011422	000240				NOP						
2975 011424	000240				NOP						
2976 011426	000240				NOP						
2977											
2978 011430	104075				ERROR	:75					
2979 011432	000401				BR	25:					
2980 011434	022626			2:	CHP	(SP)+, (SP)+					
2981 011436	012777	011502	171200	25:	MOV	BR3, BRVECT					
2982 011444	004737	014502			JSR	PC, BRPSW					
2983 011450	000140				.WORD	140					
2984 011452	000240				NOP						
2985 011454	000240				NOP						
2986 011456	000240				NOP						
2987 011460	000240				NOP						
2988 011462	042777	000100	171144		BIC	BRIT6, BRCSR					
2989 011470	010377	171150			MOV	R3, BRVECT					
2990 011474	010477	171146			MOV	R4, BRPSW					
2991 011500	000402				BR	4:					
2992											
2993 011502	022626			3:	CHP	(SP)+, (SP)+					
2994 011504	104076				ERROR	:76					
2995											
2996 011506	010377	171132		4:	MOV	R3, BRVECT					

2997

.SBTTL TEST # 36 - TEST THAT RCVR INTERRUPT CLEARS BY READING RBUF

;*****
 ;*TEST 36 TEST THAT RCVR INTERRUPT CLEARS BY READING RBUF
 ;*****

2998	011512	000004									
2999	011514	000005									
3000	011516	004737	014502								
3001	011522	000340									
3002	011524	017703	171114								
3003	011530	012777	011646	171106							
3004	011536	052777	000100	171070							
3005	011544	052777	000004	171066							
3005	011552	005000									
	011554	012701	000002								
	011560	105777	171052								
	011564	005300									
	011566	001374									
	011570	005301									
	011572	001372									
3006	011574	005077	171042								
3007	011600	105777	171030								
3008	011604	100375									
3009	011606	042777	000004	171044							
3010	011614	005777	171016								
3011	011620	004737	014502								
3012	011624	000140									
3013	011626	000240									
3014	011630	000240									
3015	011632	000240									
3016	011634	000240									
3017	011636	042777	000100	170770							
3018	011644	000402									
3019											
3020	011646	022626									
3021	011650	104077									
3022											
3023	011652	010377	170766								

```

TST36: SCOPE
        RESET
        JSR    PC,WRPSW          ;CLEAR EVERYTHING
        .WORD 340                ;SET PSM PRIORITY TO 7
        MOV    BRVECT,R3        ;SAVE RECEIVE VECTOR
        BIS    #BIT6,BRCSR      ;POINT RCV VECTOR TO ERROR REPORT
        BIS    #BIT2,BTCR       ;SET RCVR INTERRUPT ENABLE
        CLR    R0                ;SET MAINTENANCE WRAP
        MOV    #2,R1            ;** DELAY ENOUGH TIME TO ALLOW ANY C.ARACTERS
        ;** THAT MIGHT BE IN THE PROCESS OF BEING
        ;** RECEIVED TO FINISH AFTER MAINTENANCE
        ;** WRAP FOR THE UART UNDER TEST IS ENABLED.
        ;** READ TO CLEAR DONE
        TSTB   BRBUF            ;**
        DEC    R0                ;**
        BNE    42:              ;**
        DEC    R1                ;**
        BNE    42:              ;**
        CLR    BRBUF            ;SEND A CHARACTER
        TSTB   BRCSR            ;WAIT FOR DONE (INTERRUPT)
        BPL    1:
        BIC    #BIT2,BTCR       ;** DISABLE MAINTENANCE MODE FOR
        ;** CONSOLE TO ALLOW FOR COMMUNICATION
        ;** WITH TERMINAL.
        TST    BRBUF            ;READ RBUF TO CLEAR PENDING INTERRUPT ;DPM002
        JSR    PC,WRPSW          ;SET PSM TO PRIORITY 3
        .WORD 140
        NOP
        NOP
        NOP
        NOP
        BIC    #BIT6,BRCSR      ;** ALLOW TIME FOR ANY ERRONEOUS INTERRUPT
        BR     3:                ;** ALLOW TIME FOR ANY ERRONEOUS INTERRUPT
        ;** ALLOW TIME FOR ANY ERRONEOUS INTERRUPT
        ;** ALLOW TIME FOR ANY ERRONEOUS INTERRUPT
        ;** NO INTERRUPT-CLEAR INT. ENABLE
        CMP    (SP)+,(SP)+      ;RESTORE SP AFTER INTERRUPT
        ERROR  +77              ;READING RBUF DID NOT CLEAR INTERRUPT
        MOV    R3,BRVECT        ;RESTORE RECEIVE VECTOR
    
```

3024

.SBTTL TEST # 37 - TEST THAT RESET CLEARS RECEIVE INTERRUPT
 ;*****
 ;*TEST 37 TEST THAT RESET CLEARS RECEIVE INTERRUPT
 ;*****
 TST37: SCOPE

3025	011656	000004						RESET			;	CLEAR EVERYTHING
3026	011660	000005						JSR	PC,WRPSW		;	SET PSM TO PRIORITY 7
3027	011662	004737	014502						.WORD 340			
3028	011666	000340						MOV	BRVECT,R3		;	SAVE RECEIVE VECTOR
3029	011670	017703	170750					MOV	#2, BRVECT		;	POINT RCV VECTOR TO ERROR REPORT
3030	011674	012777	012004	170742				BIS	#BIT6, BRCSR		;	SET RCV INTERRUPT ENABLE
3031	011702	052777	000100	170724				BIS	#BIT2, BTCSR		;	SET MAINTENANCE WRAP
3032	011710	052777	000004	170722				CLR	R0		;	** DELAY ENOUGH TIME TO ALLOW ANY CHARACTERS
	011720	005000						MOV	#2,R1		;	** THAT MIGHT BE IN THE PROCESS OF BEING
		012701	000002								;	** RECEIVED TO FINISH AFTER MAINTENANCE
	011724	105777	170706				42:	TSTB	BRBUF		;	** WRAP FOR THE UART UNDER TEST IS ENABLED.
	011730	005300						DEC	R0		;	** READ TO CLEAR DONE
	011732	001374						BNE	42:		;	**
	011734	005301						DEC	R1		;	**
	011736	001372						BNE	42:		;	**
3033	011740	012777	000377	170674				MOV	#377, BTBUF		;	SEND AN ALL 1'S CHARACTER
3034	011746	105777	170662				1:	TSTB	BRCSR		;	WAIT FOR RCV DONE
3035	011752	100375						BPL	1:			
3036	011754	000005						RESET			;	CLEAR RCV INTERRUPT & RBUF
3037	011756	004737	014502					JSR	PC,WRPSW		;	SET PSM TO PRIORITY 3
3038	011762	000140							.WORD 140			
3039	011764	000240						NOP			;	** ALLOW TIME FOR AN ERRONEOUS INTERRUPT
3040	011766	000240						NOP			;	** ALLOW TIME FOR AN ERRONEOUS INTERRUPT
3041	011770	000240						NOP			;	** ALLOW TIME FOR AN ERRONEOUS INTERRUPT
3042	011772	000240						NOP			;	** ALLOW TIME FOR AN ERRONEOUS INTERRUPT
3043	011774	042777	000100	170632				BIC	#BIT6, BRCSR		;	NO INTERRUPT-CLEAR INT. ENABLE
3044	012002	000402						BR	3:		;	CONTINUE TEST
3045												
3046												
3047	012004	022626					2:	CMP	(SP)+,(SP)+		;	RESTORE SP AFTER INTERRUPT
3048	012006	104100						ERROR	+100		;	RESET DID NOT CLEAR RCVR INTERRUPT
3049												
3050	012010	010377	170630				3:	MOV	R3, BRVECT		;	RESTORE RECEIVE VECTOR

3051

.SBTTL TEST # 40 - TEST THAT THE "OR" ERROR & "ERROR" CAN BE SET

 ;*TEST 40 TEST THAT THE "OR" ERROR & "ERROR" CAN BE SET

012014 000004
 3052 012016 032777 002000 167014
 3053 012024 001465
 3054 012026 032737 000001 003002
 3055 012034 001407
 3056 012036 005737 002624
 3057
 3058 012042 001404
 3059 012044 032777 000010 166766
 3060
 3061 012052 001452
 3062 012054 000005
 3063 012056 052777 000004 170554
 3064 012064 005000
 012066 012701 000002

 012072 105777 170540
 012076 005300
 012100 001374
 012102 005301
 012104 001372
 3065 012106 012700 000003
 3066 012112 005077 170524
 3067 012116 105777 170516
 3068 012122 100375
 3069 012124 005300
 3070 012126 001371
 3071 012130 042777 000004 170522

 3072 012136 032777 040000 170472
 3073 012144 001001
 3074 012146 104101
 3075
 3076
 3077 012150 032777 100000 170460
 3078 012156 001001
 3079 012160 104102
 3080
 3081 012162
 3082 012162 005000
 012164 012701 000002
 012170 005300
 012172 001376
 012174 005301
 012176 001374

TST40: SCOPE
 BIT #BIT10,BSWR
 BEQ TST41
 BIT #BIT0,FLAG44
 BEQ 9#
 TST CTSTFL

 BEQ 9#
 BIT #BIT03,BSWR

 BEQ TST41
 9#: RESET
 BIS #BIT2,@TCSR
 CLR R0
 MOV #2,R1

 42#: TSTB @RBUF
 DEC R0
 BNE 42#
 DEC R1
 BNE 42#
 MOV #3,R0
 1#: CLR @TBUF
 2#: TSTB @TCSR
 BPL 2#
 DEC R0
 BNE 1#
 BIC #BIT2,@TCSR

 BIT #BIT14,@RBUF
 BNE 3#
 ERROR +101

 3#: BIT #BIT15,@RBUF
 BNE 4#
 ERROR +102

 4#: CLR R0
 MOV #2,R1
 40#: DEC R0
 BNE 40#
 DEC R1
 BNE 40#

;IS THIS TEST ENABLED
 ;IF NOT ENABLED, BR TO NEXT TEST
 ;** IS THIS A 11/44
 ;** NO
 ;** YES THIS IS 11/44. IS THIS THE CONSOLE
 ;** SLU
 ;** NO
 ;** THIS IS THE CONSOLE SLU.SHOULD THE OVERRUN
 ;** ERROR TEST BE PERFORMED
 ;** NO
 ;CLEAR EVERYTHING
 ;SET MAINTENANCE WRAP
 ;** DELAY ENOUGH TIME TO ALLOW ANY CHARACTERS
 ;** THAT MIGHT BE IN THE PROCESS OF BEING
 ;** RECEIVED TO FINISH AFTER MAINTENANCE
 ;** WRAP FOR THE UART UNDER TEST IS ENABLED.
 ;** READ TO CLEAR DONE
 ;**
 ;**
 ;SET CHARACTER COUNT TO SEND 3 CHAR.
 ;LOAD TRANSMIT BUFFER
 ;WAIT FOR TRANSMIT DONE

 ;DECREMENT CHARACTER COUNT
 ;BR IF ALL CHARACTERS NOT TRANSMITTED
 ;** DISABLE MAINTENANCE MODE FOR
 ;** CONSOLE TO ALLOW FOR COMMUNICATION
 ;** WITH TERMINAL.
 ;TEST FOR "OR" ERROR FLAG
 ;BR, IF SET

 ;"OR" ERROR FLAG DID NOT SET

 ;TEST "ERROR" FLAG
 ;BR, IF SET

 ;"ERROR" FLAG DID NOT SET WITH "OR" FLAG

 ;** DELAY ENOUGH TIME TO ALLOW ANY CHARACTERS
 ;** THAT MIGHT BE IN THE PROCESS OF BEING
 ;** TRANSMITTED TO FINISH BEFORE MAINTENANCE
 ;** WRAP FOR THE UART UNDER TEST IS DISABLED.
 ;** THIS WILL INHIBIT ANY COMMUNICATION
 ;** TO HARDWARE MEDIA SUCH AS TU58 THAT MIGHT
 ;** BE ATTACHED TO UART UNDER TEST.

3121

.SBTTL TEST # 42 - TEST THAT "FR" ERROR CAN BE SET DURING BREAK

|||
 |||TEST 42 TEST THAT "FR" ERROR CAN BE SET DURING BREAK
 |||

3122	012414	000004				TST42	SCOPE	
3122	012416	032777	002000	166414			BIT #BIT10,BSMR	;IS THE "TEST ERROR FLAGS" BIT SET
3123	012424	001464					BEQ TST43	;OR TO NEXT TEST, IF NOT SET
3124	012426	032777	000400	166404			BIT #BIT8,BSMR	;IS BREAK FUNCTION ENABLED
3125	012434	001460					BEQ TST43	;OR TO NEXT TEST, IF NOT SET
3126	012436	032737	000001	003002			BIT #BIT0,FLAG44	;** IS THIS A 11/44
3127	012444	001407					BEQ 91	;** NO
3128	012446	005737	002624				TST CTSTFL	;** YES THIS IS 11/44. IS THIS THE CONSOLE
3129								;** SLU
3130	012452	001404					BEQ 91	;** NO
3131	012454	032777	000010	166356			BIT #BIT03,BSMR	;** THIS IS THE CONSOLE SLU.SHOULD THE FRAME
3132								;** ERROR TEST BE PERFORMED
3133	012462	001445					BEQ TST43	;** NO
3134	012464	000005			91:		RESET	;CLEAR EVERYTHING
3135	012466	032777	000004	170144			BIS #BIT2,BTCSR	;SET MAINTENANCE WRAP
3136	012474	005000					CLR R0	;** DELAY ENOUGH TIME TO ALLOW ANY CHARACTERS
	012476	012701	000002				MOV #2,R1	;** THAT MIGHT BE IN THE PROCESS OF BEING
								;** RECEIVED TO FINISH AFTER MAINTENANCE
								;** WRAP FOR THE UART UNDER TEST IS ENABLED.
	012502	105777	170130		42:		TSTB #RBUF	;** READ TO CLEAR DONE
	012506	005300					R0	;**
	012510	001374					BNE 421	;**
	012512	005301					DEC R1	;**
	012514	001372					BNE 421	;**
3137	012516	032777	000001	170114			BIS #BIT0,BTCSR	;SEND BREAK
3138	012524	005077	170112				CLR #RBUF	;TRANSMIT A CHARACTER TO TIME BREAK
3139	012530	105777	170100		16:		TSTB #BTCSR	;WAIT FOR RCVR DONE
3140	012534	100375					BPL 11	
3141	012536	042777	000001	170074			BIC #BIT0,BTCSR	;CLEAR BREAK BIT
3142	012544	042777	000004	170106			BIC #BIT2,BTCSR	;** DISABLE MAINTENANCE MODE FOR
								;** CONSOLE TO ALLOW FOR COMMUNICATION
								;** WITH TERMINAL.
3143	012552	032777	020000	170056			BIT #BIT13,#RBUF	;CHECK FOR FRAMING ERROR FLAG
3144	012560	001001					BNE 21	;OR, IF SET
3145								
3146	012562	104104					ERROR +104	
3147								
3148	012564	032777	100000	170044	21:		BIT #BIT15,#RBUF	;BREAK DID NOT SET FRAMING ERROR
3149	012572	001001					BNE 31	;TEST "ERROR" FLAG
3150								;OR, IF SET
3151	012574	104114					ERROR +114	
3152								
3153	012576				31:			; "ERROR" FLAG DID NOT SET WITH "OR" FLAG

3190

.SBTTL TEST # 44 - TEST DATA PATHS USING LOOP-BACK CONNECTOR
 ;*****
 ;TEST 44 TEST DATA PATHS USING LOOP-BACK CONNECTOR
 ;*****

3191	012744	000004				TST44: SCOPE		
3191	012746	032777	000200	166064		BIT #BIT7,BSWR		;IS THIS TEST ENABLED
3192	012754	001442				BEQ TST45		;BR, IF NOT
3193	012756	000005				RESET		;CLEAR EVERYTHING
3194	012760	005001				CLR R1		;CLEAR REGISTER FOR TEST DATA
3195								;TRANSMIT A BINARY COUNT PATTERN - UP
3196								;TO THE BIT POSITION INDICATED BY THE
3197								;CONTENTS OF LOCATION "{USMR"
3198	012762	105201			1#:	INCB R1		;INCREMENT THE TEST DATA
3199	012764	032737	000001	003002		BIT #BIT0,FLAG44		;11/44 CPU
3200	012772	001404				BEQ 5#		;TEST ALL DATA
3201	012774	023727	002634	177560		CHP RCSR,#177560		;IS THIS 11/44 CONSOLE TERMINAL SLU?
3202	013002	001427				BEQ TST45		;YES, SKIP THIS TEST
3203	013004	010177	167632		5#:	MOV R1,#TBUF		;XMIT A CHARACTER
3204	013010	005000				CLR R0		;CLEAR A TIMER
3205	013012	105777	167616		2#:	TSTB BRCSR		;WAIT FOR RECEIVER DONE
3206	013016	100403				BMI 3#		;BR IF DONE
3207	013020	005200				INC R0		;INCREMENT TIMER IF NOT
3208	013022	001373				BNE 2#		;BR IF TIME REMAINS
3209								
3210	013024	104064				ERROR +64		;RECEIVER DONE NOT SET
3211								
3212	013026	017702	167604		3#:	MOV BRBUF,R2		;GET RECEIVED CHARACTER
3213	013032	043701	001112			BIC #USMR,R1		;CLEAR LOWEST UNUSED DATA BIT POSITON IN TEST DATA
3214	013036	020102				CHP R1,R2		;COMPARE DATA
3215	013040	001003				BNE 4#		;BR, IF NON-COMPARE
3216	013042	105701				TSTB R1		;TEST XMIT DATA FOR ZERO
3217	013044	001406				BEQ TST45		;BR, IF FINISHED
3218	013046	000745				BR 1#		;CONTINUE IF NOT
3219	013050	010137	001024		4#:	MOV R1,#GDAT		;STORE EXPECTED DATA
3220	013054	010237	001026			MOV R2,#DDAT		;STORE RECEIVED DATA
3221								
3222	013060	104106				ERROR +106		;DATA COMPARE ERROR USING LOOP-BACK CONNECTOR

3223

.SBTTL TEST # 45 - TEST DL11-W LOGIC BY EXERCISING THE XMIT, REC, & CLOCK
 ;*****
 ;*TEST 45 TEST DL11-W LOGIC BY EXERCISING THE XMIT, REC, & CLOCK
 ;*****
 TST45: SCOPE

3224	013062	000004				RESET		;CLEAR EVERYTHING
3225	013064	000005				CLR	CLKCNT	;INITIALIZE CLOCK COUNT TO ZERO
3226	013072	004737	002444			JSR	PC,MRPSW	;SET PRIORITY TO 7
3227	013076	000340				.WORD	340	
3228	013100	017703	167544			MOV	@TVECT,R3	;SAVE XMIT VECTOR
3229	013104	017704	167534			MOV	@RVECT,R4	;SAVE RECEIVE VECTOR
3230	013110	017705	167562			MOV	@RTCVT,R5	;SAVE CLOCK VECTOR
3231	013114	017737	167532	002446		MOV	@TPSW,STPSW	;SAVE XMIT PSW VECTOR
3232	013122	017737	167520	002450		MOV	@RPSW,SRPSW	;SAVE RECEIVE PSW VECTOR
3233	013130	017737	167544	002452		MOV	@RTCPSW,SCPSW	;SAVE CLOCK PSW VECTOR
3234	013136	012777	013542	167504		MOV	@XMIT,@TVECT	;POINT TRANSMIT VECTOR TO TRANSMIT ROUTINE
3235	013144	012777	000200	167500		MOV	@200,@TPSW	;NO MULTIPLE INTERRUPTS ALLOWED
3236	013152	012777	013614	167464		MOV	@RCV,@RVECT	;POINT RECEIVE VECTOR TO RECEIVE ROUTINE
3237	013160	012777	000200	167460		MOV	@200,@RPSW	;NO MULT INTERRUPTS
3238	013166	005737	002624			TST	CTSTFL	;IS CONSOLE UNDER TEST?
3239	013172	001415				BEQ	1#	;IF NOT SKIP CLOCK SET UP
3240	013174	032777	000100	165636		BIT	@BIT6,BSMR	;IF YES, ARE CLOCK TEST DISABLED?
3241	013202	001011				BNE	1#	;IF YES, SKIP CLOCK SET UP
3242	013204	012777	013630	167464		MOV	@CLK,@RTCVT	;POINT VECTOR TO CLOCK INTERRUPT ROUTINE
3243	013212	012777	000300	167460		MOV	@300,@RTCPSW	;NO MULT INTERRUPTS
3244	013220	052777	000100	167446		BIS	@BIT6,BLKS	;ENABLE CLOCK INTERRUPTS
3245	013226	052777	000004	167404	1#:	BIS	@BIT2,@TCSR	;SET MAINTENANCE WRAP
3246	013234	005000				CLR	R0	;** DELAY ENOUGH TIME TO ALLOW ANY CHARACTERS
	013236	012701	000002			MOV	@2,R1	;** THAT MIGHT BE IN THE PROCESS OF BEING
								;** RECEIVED TO FINISH AFTER MAINTENANCE
								;** WRAP FOR THE UART UNDER TEST IS ENABLED.
								;** READ TO CLEAR DONE
								;**
	013242	105777	167370		4#:	TSTB	@RBUF	
	013246	005300				DEC	R0	
	013250	001374				BNE	4#	
	013252	005301				DEC	R1	
	013254	001372				BNE	4#	
3247	013256	052777	000100	167354		BIS	@BIT6,@TCSR	;ENABLE TRANSMIT INTERRUPTS
3248	013264	052777	000100	167342		BIS	@BIT6,@RCSR	;ENABLE RECEIVE INTERRUPTS
3249	013272	005037	002442			CLR	XMITCNT	;CLEAR XMIT INTERRUPT COUNTER
3250	013276	005037	002440			CLR	RCVCNT	;CLEAR RCV INTERRUPT COUNTER
3251	013302	005001				CLR	R1	;CLEAR A REGISTER FOR TEST DATA USE
3252	013304	005000				CLR	R0	;CLEAR TIMER
3253	013306	012702	002454			MOV	@RBUF,R2	;POINT R2 TO RECEIVE DATA STORAGE
3254	013312	005077	167324			CLR	@TBUF	;SEND FIRST CHARACTER
3255	013316	004737	014502			JSR	PC,MRPSW	;SET PSW TO PRIORITY 3
3256	013322	000140				.WORD	140	
3257								;WAIT FOR INTERRUPTS
3258	013324	000240			2#:	NOP		;STALL
3259	013326	000240				NOP		;STALL
3260	013330	062700	000000			ADD	@0,R0	;ADD INSTRUCTIONS ARE USED TO LENGTHEN LOOP TIME
3261	013334	062700	000001			ADD	@1,R0	; TO COVER THE SLOWEST BAUD RATE ON THE FASTEST CPU
3262	013340	001371				BNE	2#	
3263	013342	032777	000100	167270		BIT	@BIT6,@TCSR	;FINISHED ENTIRE TRANSMISSION
3264	013350	001402				BEQ	3#	;OR, IF INTERRUPTS ARE DISABLED (FINISHED)
3265	013352	000005				RESET		;CLEAR EVERYTHING
3266								
3267	013354	104107				ERROR	*107	;TRANSMIT INTERRUPT TIMEOUT IN MAIN. DATA TEST

3268										
3269	013356	023737	002442	002440	34:	CMP	XMTCNT,RCVCNT			;COMPARE THE NUMBER OF INTERRUPTS
3270	013364	001402				BEQ	44			;BR, IF EQUAL
3271	013366	000005				RESET				;CLEAR EVERYTHING
3272										
3273	013370	104110				ERROR	+110			;RECEIVER DID NOT GET FULL TRANSMISSION
3274										; IF RCVCNT=0, NO DATA RECEIVED
3275										; IF RCVCNT=7, THEN (XMTCNT-RCVCNT)
3276										; EQUALS THE NO. OF INTERRUPTS LOST.
3277	013372	005737	002624		44:	TST	CTSTFL			;IS CONSOLE UNDER TEST?
3278	013376	001411				BEQ	54			;IF NOT, SKIP CLOCK COUNT CHECK
3279	013400	032777	000100	165432		BIT	#BIT6,#SWR			;IF YES, ARE CLOCK TESTS DISABLED?
3280	013406	001005				BNE	54			;IF YES, SKIP CLOCK COUNT CHECK
3281	013410	005737	002444			TST	CLKCNT			;CHECK FOR AT LEAST ONE CLOCK INTERRUPT
3282	013414	001002				BNE	54			;BR IF INTERRUPTS OCCURRED
3283	013416	000005				RESET				;CLEAR MAINTENANCE WRAPAROUND
3284	013420	104113				ERROR	+113			;NO CLOCK INTERRUPTS IN EXERCISER
3285										
3286	013422	000005			54:	RESET				;CLEAR EVERYTHING
3287	013424	012700	002454			MOV	#BUF,R0			;LOAD RECEIVED DATA POINTER TO R0
3288	013430	005001				CLR	R1			;SET UP REGISTER FOR COMPARISON
3289	013432	022001			COMP:	CMP	(R0)+,R1			;COMPARE XMIT & RCV DATA
3290	013434	001014				BNE	64			;BR, IF NOT EQUAL
3291	013436	105201			94:	INCB	R1			;INCREMENT COMPARE DATA
3292	013440	032737	000001	003002		BIT	#BIT0,FLAG44			;11/44 CPU?
3293	013446	001403				BEQ	84			;YES, SKIP
3294	013450	122701	000020			CMPSB	#20,R1			;SKIP 20 DATA IF 11/44
3295	013454	001770				BEQ	94			;SKIP 20
3296	013456	032701	000040		84:	BIT	#BIT5,R1			;FINISHED CHECKING RECEIVED DATA?
3297	013462	001763				BEQ	COMP			;BR, IF NOT FINISHED
3298	013464	000405				BR	74			;BR TO END OF TEST
3299										
3300	013466	014037	001026		64:	MOV	-(R0),#BDDAT			;STORE BAD DATA FOR ERROR REPORT
3301	013472	010137	001024			MOV	R1,#GDDAT			;STORE GOOD DATA FOR ERROR REPORT
3302	013476	104111				ERROR	+111			;DATA COMPARE ERROR IN EXERCISER
3303										
3304	013500	010377	167144		74:	MOV	R3,#TVECT			;RESTORE XMIT VECTOR
3305	013504	010477	167134			MOV	R4,#RVECT			;RESTORE RECEIVE VECTOR
3306	013510	010577	167162			MOV	R5,#RTCVT			;RESTORE CLOCK VECTOR
3307	013514	013777	002446	167130		MOV	STPSW,#TPSW			;RESTORE XMIT PSW VECTOR
3308	013522	013777	002450	167116		MOV	SRPSW,#RPSW			;RESTORE RECEIVE PSW VECTOR
3309	013530	013777	002452	167142		MOV	SCPSW,#RTCPSW			;RESTORE CLOCK PSW VECTOR
3310	013536	000137	014272			JMP	ENDEV			;GOTO NEXT TEST
3311										
3312	013542	005237	002442		XMIT:	INC	XMTCNT			;INCREMENT XMIT INTERRUPT COUNTER
3313	013546	105201			14:	INCB	R1			;INCREMENT TEST DATA
3314	013550	032737	000001	003002		BIT	#BIT0,FLAG44			;11/44 CPU
3315	013556	001403				BEQ	24			;TEST ALL DATA
3316	013560	122701	000020			CMPSB	#20,R1			;CHECK DATA FOR ↑P
3317	013564	001770				BEQ	14			;DO NOT XMIT ↑P
3318	013566	032701	000040		24:	BIT	#BIT5,R1			;SEND DATA PATTERN FROM 00 --> 37
3319	013572	001404				BEQ	XCONT			;BR, IF MORE DATA TO BE SENT
3320	013574	042777	000100	167036		BIC	#BIT6,#TCSR			;CLEAR XMIT INTERRUPT ENABLE
3321	013602	000402				BR	XRET			;RETURN, WITHOUT SENDING ANY MORE DATA
3322	013604	110177	167032		XCONT:	MOVB	R1,#TBUF			;SEND NEW CHARACTER
3323	013610	005000			XRET:	CLR	R0			;CLEAR TIMER
3324	013612	000002				RTI				;RETURN

```
3325
3326 013614 017722 167016      RCV:  MOV  SRBUF,(R2).  ;STORE RECEIVED DATA
3327 013620 005237 002440      INC  RVCNT  ;INCREMENT RCV INTERRUPT COUNTER
3328 013624 005000              CLR  R0     ;CLEAR TIMER
3329 013626 000002              RTI                    ;RETURN
3330
3331 013630 005237 002444      CLK:  INC  CLKCNT  ;INCREMENT CLOCK INTERRUPT COUNT
3332 013634 000002              RTI                    ;RETURN
```

3333

.SBTTL TEST # 46 - TEST CONSOLE WITH WRAP AROUND

TEST 46 TEST CONSOLE WITH WRAP AROUND

TST46: SCOPE

WRAPAROUND TESTING- AUTO INITIATION OF T/A CONSOLE TEST

013636 000004

3334

3335

3336

3337

3338

3339

3340

3341

3342

3343

3344

3345

3346

3347

3348

3349

3350

3351

3352

3353

3354

3355

3356

3357

3358

3359

3360

3361

3362

3363

3364

3365

3366

3367

3368

3369

3370

3371

3372

3373

3374

3375

3376

3377

3378

3379

3380

3381

3382

3383

3384

3385

176500
 176502
 176504
 176506
 003014
 025252

RCSR58 = 176500
 RBUF58 = 176502
 TCSR58 = 176504
 TBUF58 = 176506
 BGNADD = DEVADR
 ENDADD = ENDADR

MAIN LINE TEST

THIS TEST PUTS COMMANDS OUT OVER THE SERIAL LINE WHICH IS WRAPPED
 AROUND TO THE CONSOLE AND RECEIVES THE RESPONSES

CONTROL P IS SEND TO GET THE CONSOLE'S ATTENTION AND THEN
 T/A(A FOR APT) IS SENT. SINCE THE CPU IS HALTED DURING THE TESTING
 THE PROGRAM CAN NOT SEE THE CHARS RETURNING, THUS THE PROGRAM WILL
 SIT IN A LOOP WAITING FOR A "B" TO BE PRINTED BY THE CONSOLE AS A
 SIGNAL TO APT THAT THE TESTING IS DONE. ALSO SINCE THE TESTING
 INVOLVES WRITTING TO THE CPU'S MEMORY A CHECKSUM IS CALCULATED AND THE
 MEMORY TO BE USED INSIDE THIS PROGRAMS SPACE IS SAVE BEFORE TESTING
 AND RESTORED AFTER TEST WITH ANOTHER CHECKSUM CALCULATION

3362	013640	032777	000004	165172	WRAP:	BIT	@BIT2,BSWR	;RUN THIS TEST ONLY IF
3363	013646	001451				BEQ	104	;SW BIT 2 IS ON
3365	013650	013737	177776	002434		MOV	PSW,SAVEPS	;SAVE OLD PSW
3366	013656	012737	000340	177776		MOV	@340,PSW	;PUT IN NOW PSW
3367	013664	012706	002432			MOV	@JIMSTK,SP	;USE MY STACK (SO NO CLOBER)
3369	013670	005037	002340			CLR	LOC1	
3370	013674	012737	000100	002342		MOV	@100,LOC2	;TIMING LOOP COUNTERS
3372	013702	004537	014202			JSR	R5,SAVETE	;SAVE LOCATIONS T/A WRITES
3374	013706	004537	014010			JSR	R5,CHKSUM	;CALCULATE CHECKSUM
3375	013712	010437	002436			MOV	R4,OLDSUM	;SAVE OLD CHECKSUM
3377	013716	012700	002564			MOV	@CNTLP,R0	
3378	013722	004537	014030			JSR	R5,PUTLIN	;SEND OUT CONTROL P
3380	013726	012700	002566			MOV	@PROMPT,R0	
3381	013732	004537	014062			JSR	R5,GETLIN	;GET CRLF CONSOLE>>>
3383	013736	012700	002610			MOV	@TA,R0	
3384	013742	004537	014030			JSR	R5,PUTLIN	;SEND OUT T/A<CRLF>

```

3386 013746 004537 014126      JSR      R5,GETB      ;GET THE A FROM "-TESTB"
3387                               ;
3388 013752 004537 014236      JSR      R5,RESTTE    ;RESTORE LOCATIONS T/A WRITES
3389                               ;
3390 013756 004537 014010      JSR      R5,CHKSUM    ;RECALCULATE CHECKSUM
3391 013762 020437 002436      CMP      R4,OLDSUM
3392 013766 001401              BEQ      10$
3393 013770 000000              HALT
3394 013772 012706 001000      MOV      #1000,SP     ;RETURN THEIR SP
3395 013776 013737 002434      MOV      SAVEPS,PSW  ;RETURN PSW
3396 014004 000137 014326      JMP      #EOP         ;BR TO END OF PASS ROUTINE
3397
3398
3399                               ;*****
3400                               ;
3401                               ; ROUTINE TO CALCULATE CHECKSUM ON PROGRAM
3402                               ;
3403                               ; INPUT CONDITIONS
3404                               ;
3405                               ;     BGNADD = ADDRESS TO START CHECK SUM (INCLUSIVE)
3406                               ;     ENDADD  = ADDRESS TO END CHECK SUM (EXCLUSIVE)
3407                               ;
3408                               ; OUTPUT CONDITIONS
3409                               ;
3410                               ;     REG4 = CHECK SUM
3411                               ;
3412                               ;*****
3413
3414
3415 014010 012700 003014      CHKSUM:  MOV      #BGNADD,R0      ;GET STARTING ADDRESS
3416 014014 005004              CLR      R4                    ;RESET SUM
3417 014016 062004      1$:    ADD      (R0)+,R4            ;ADD WORD TO SUM
3418 014020 022700 025252      CMP      #ENDADD,R0          ;CHECK FOR END
3419 014024 001374              BNE     1$                     ;IF NOT DONE LOOP
3420 014026 000205              RTS      R5                    ;DONE RETURN
3421
3422
3423                               ;*****
3424                               ;
3425                               ; THIS ROUTINE OUTPUTS TO THE SERIAL LINE CHARS STARTING AT
3426                               ; THE ADDRESS IN REG0 UNTIL IT HITS A <377>
3427                               ;
3428                               ;*****
3429
3430
3431 014030 112001      PUTLIN:  MOVB     (R0)+,R1            ;GET DATA
3432 014032 001412      BEQ      10$                  ;END OF DATA
3433 014034 004537 014162      1$:    JSR      R5,TIMER            ;PROVIDE FOR TIMEOUT
3434 014040 032737 000200 176504  BIT      #BIT7,TCSR58        ;TEST FOR XMIT READY
3435 014046 001772      BEQ      1$                     ;WAIT FOR XMIT READY
3436 014050 110137 176506      MOVB     R1,TBUF58          ;OUTPUT CHAR
3437 014054 000137 014030      JMP      PUTLIN              ;REPETE
3438 014060 000205      10$:   RTS      R5                    ;RETURN
3439
3440                               ;*****
3441                               ;
3442                               ; THIS ROUTINE INPUTS A CHARS FROM THE SERIAL LINE AND COMPARES
    
```

```

3443 ; IT WITH THE EXPECTED VALUES POINTED TO BY REGO UNTIL NULL<00>
3444 ;
3445 ;*****
3446
3447
3448 014062 112001 GETLIN:      MOVB      (R0)+,R1      ;GET EXPECTED CHAR
3449 014064 105201          INCB      R1
3450 014066 001416          BEQ      104          ;IF NULL EXIT
3451 014070 105301          DECB      R1
3452 014072 004537 014162 14:      JSR      R5,TIMER     ;PROVIDE FOR TIMOUT
3453 014076 032737 000200 176500  BIT      @BIT7,RCSR58 ;TEST FOR REC READY
3454 014104 001772          BEQ      14          ;WAIT FOR REC READY
3455 014106 113702 176502      MOVB      RBUF58,R2
3456 014112 042702 000200      BIC      @BIT7,R2     ;STRIP PARITY
3457 014116 120102          CMPB      R1,R2       ;ARE THEY THE SAME
3458 014120 001760          BEQ      GETLIN      ;SAME GET MORE
3459 014122 000000          HALT
3460 014124 000205 104:      RTS      R5          ;NOT SAME HALT
3461 ; RETURN
3462
3463 ;*****
3464 ;
3465 ; THIS ROUTINE INPUTS CHARS UNTIL IT GETS THE CHAR "B"
3466 ; AND THEN RETURNS
3467 ;
3468 ;*****
3469
3470
3471 014126 004537 014162 GETB:      JSR      R5,TIMER     ;OUT TIMING LOOP OUT
3472 014132 032737 000200 176500  BIT      @BIT7,RCSR58 ;TEST OFR REC READY
3473 014140 001772          BEQ      GETB        ;NOT DONE YET
3474 014142 113702 176502      MOVB      RBUF58,R2
3475 014146 042702 000200      BIC      @BIT7,R2     ;STRIP PARITY
3476 014152 122702 000102      CMPB      #102,R2    ;CHECK FOR "B"
3477 014156 001363          BNE      GETB        ;NOT "B" REPETE
3478 014160 000205          RTS      R5          ;WAS "B" RETURN
3479 ;
3480 ;*****
3481 ;
3482 ; THIS ROUTINE IS USED AS A TIME OUT FEATURE
3483 ; WHEN THE TIMING LOOPS BOTH REACH 0 THIS ROUTINE WILL
3484 ; CAUSE A HALT
3485 ;
3486 ;*****
3487
3488
3489 014162 005337 002340 TIMER:      DEC      LOC1
3490 014166 001004          BNE      104          ;DECREMPNT TIMING LOOPS
3491 014170 005337 002342      DEC      LOC2
3492 014174 001001          BNE      104
3493 014176 000000          HALT
3494 014200 000205 104:      RTS      R5          ;IF ZERO THIS ROUTINE
3495 ; EXECUTED R3 TIMES
3496
3497 ;*****
3498 ;
3499 ; THIS ROUTINE SAVES THE LOCATIONS WRITTEN BY THE T/A CONSOLE TEST
    
```



```

3500 ; SO THEY MAY BE RESTORED LATER
3501 ;
3502 ;*****
3503
3504 014202 013737 000000 002344 SAVETE:      MOV      0,SAVE0      ;SAVE LOCATION 0
3505 014210 012702 000002                MOV      @2,R2        ;SET UP INDIRECT PNTER
3506 014214 012701 002346                MOV      @SAVLOC,R1   ;SET UP STORAGE LOC. PNTER
3507 014220 011221                1#:      MOV      (R2),(R1)+ ;GET WORD AND SAVE
3508 014222 000241                CLC                    ;DONT ROT IN ANY BITS
3509 014224 006102                ROL      R2            ;SET NEXT ADDRESS
3510 014226 020227 025252                CMP      R2,#ENDADD   ;SEE IF AT END
3511 014232 100772                BMI      1#           ;NO REPETE
3512 014234 000205                RTS      R5
3513
3514
3515 ;*****
3516 ;
3517 ; THIS ROUTINE RESTORES THE SAVED LOCATIONS THAT T/A WRITES INTO
3518 ;
3519 ;*****
3520
3521 014236 013737 002344 000000 RESTTE:    MOV      SAVE0,0
3522 014244 012702 000002                MOV      @2,R2        ;SET UP INDIRECT PNTER
3523 014250 012701 002346                MOV      @SAVLOC,R1   ;SET UP STORAGE LOC. PNTER
3524 014254 012112                1#:      MOV      (R1)+,(R2) ;GET WORD AND RESTORE
3525 014256 000241                CLC                    ;DONT ROT IN ANY BITS
3526 014260 006102                ROL      R2            ;SET NEXT ADDRESS
3527 014262 020227 025252                CMP      R2,#ENDADD   ;SEE IF AT END
3528 014266 100772                BMI      1#           ;NO REPETE
3529 014270 000205                RTS      R5
    
```

```
3530                                     ;END OF DEVICE PASS ROUTINE
3531 014272 005037 001002             ENDEV: CLR      $TSTNM      ;CLEAR TEST NO. COUNT FOR SCOPE ROUTINE
3532 014276 005237 001076             INC      $DEVCT      ;INCREMENT DEVICE COUNTER
3533 014302 023737 002630 001076     CMP      TMP2,$DEVCT ;ALL DEVICES TESTED
3534 014310 001002                     BNE     NOEOP        ;BR, IF NO
3535 014312 000137 013640             JMP     WRAP         ;EXECUTE WRAP AROUND AFTER ALL DEVICES
3536 014316 005037 002624             NOEOP: CLR     CTSTFL ;CLEAR CONSOLE UNDER TEST FLAG
3537 014322 000137 004050             JMP     TSTDEV      ;GO TEST NEXT DEVICE
```

3539

.SBTTL END OF PASS ROUTINE

;*INCREMENT THE PASS NUMBER (#PASS)

;*IF THERES A MONITOR GO TO IT

;*IF THERE ISN'T JUMP TO GOAGIN

\$EOP:

014326	000004				SCOPE		
014326	005037	001002			CLR	\$TSTNM	::ZERO THE TEST NUMBER
014330	005237	001074			INC	\$PASS	::INCREMENT THE PASS NUMBER
014340	042737	100000	001074		BIC	@100000,\$PASS	::DON'T ALLOW A NEG. NUMBER
014346	005327				DEC	(PC)+	::LOOP?
014350	000001			\$EOPCT:	.WORD	1	
014352	003015				BGT	\$DOAGN	::YES
014354	012737				MOV	(PC)+,\$(PC)+	::RESTORE COUNTER
014356	000001			\$ENDCT:	.WORD	1	
014360	014350				\$EOPCT		
014362	104401	014416			TYPE	,ENDMG	::TYPE "END PASS"
014366	013700	000042		\$GET42:	MOV	\$42,R0	::GET MONITOR ADDRESS
014372	001405				BEQ	\$DOAGN	::BRANCH IF NO MONITOR
014374	000005				RESET		::CLEAR THE WORLD
014376	004710			\$ENDAD:	JSR	PC,(R0)	::GO TO MONITOR
014400	000240				NOP		::SAVE ROOM
014402	000240				NOP		::FOR
014404	000240				NOP		::ACT11
014406				\$DOAGN:			
014406	000137				JMP	\$(PC)+	::RETURN
014410	014432			\$RTNAD:	.WORD	GOAGIN	
014412	377	377	000	\$ENULL:	.BYTE	-1,-1,0	::NULL CHARACTER STRING
					.EVEN		
3540	014416	015	012	105	ENDMG:	.ASCIZ	<CR><LF>/END PASS /

3542	014432	005026		GOAGIN:	CLR	-(SP)		;CLEAR ANOTHER LOCATION ON STACK
3543	014434	005216		18:	INC	(SP)		;INCREMENT STACK LOCATION
3544	014436	000240			NOP			;TAKE UP SOME MORE TIME
3545	014440	001375			BNE	18		;BRANCH BACK UNTIL ZERO AGAIN
3546	014442	062706	000002		ADD	#2,SP		;CLEAN THE LOCATION OFF THE STACK
3547	014446	005037	001076		CLR	#DEVCT		;CLEAR DEVICE COUNT
3548	014452	022737	000001	002630	CHP	#1,TMP2		;IS THERE ONLY ONE DEVICE UNDER TEST?
3549	014460	001004			BNE	RSTR		;BR, IF NOT
3550	014462	012706	001000		MOV	#1000,SP		;RESET STACK POINTER
3551	014466	000137	004172		JMP	TST1		;GO DO ANOTHER PASS
3552								
3553	014472	005037	001100		RSTR:	CLR	#UNIT	;CLEAR UNIT NUMBER
3554	014476	000137	003774			JMP	BEGIN	
3555								
3556	014502	011646			WRPSW:	MOV	(SP),-(SP)	;COPY RETURN PC
3557	014504	017666	000002	000002		MOV	#2(SP),2(SP)	;MOVE NEW PSW TO STACK
3558	014512	062716	000002			ADD	#2,(SP)	;ADJUST JSR RETURN OVER PRIORITY REQUESTED
3559	014516	000002				RTI		;POP RETURN PC & NEW PSW
3560								
3561								;SUBROUTINE TO REPORT UNEXPECTED OR ERRONEOUS TRAPS OR INTERRUPTS
3562								
3563	014520	012600			CATCH:	MOV	(SP),R0	;GET ADDRESS OF TRAP VECTOR + 4
3564	014522	162700	000004			SUB	#4,R0	;ADJUST TO POINT TO TRAP ADDRESS
3565	014526	010037	002622			MOV	R0,BDVCT	;STORE TRAP OR INTERRUPT ADDRESS
3566	014532	016637	000002	002620		MOV	2(SP),OLDPC	;GET PC WHERE TRAP OR INTERRUPT OCCURRED
3567	014540	104112				ERROR	+112	;REPORT ERROR
3568								
3569	014542	000000				HALT		;PROGRAM MUST BE RESTARTED AT THIS POINT

```

3571
3572
3573
3574
3575
3576
3577
3578
3579
3580
3581
3582
3583
3584 014544
3585 014544 105237 001003
3586 014550 001775
3587 014552 013777 001002 164262
3588 014560 005237 001012
3589 014564 011637 001016
3590 014570 162737 000002 001016
3591 014576 117737 164214 001014
3592 014604 032777 020000 164226
3593 014612 001004
3594 014614 004737 014726
3595 014620 104401 001063
3596 014624
3597 014624 122737 000001 001106
3598 014632 001007
3599 014634 113737 001014 014646
3600 014642 004737 0157'4
3601 014646 000
3602 014647 000
3603 014650 000777
3604 014652 005777 164162
3605 014656 100001
3606 014660 000000
3607 014662 104406
3608 014664 032777 001000 164146
3609 014672 001402
3610 014674 013716 001010
3611 014700 005737 001060
3612 014704 001402
3613 014706 013716 001060
3614 014712
3615 014712 022737 014376 000042
3616 014720 001001
3617 014722 000000
3618 014724
3619 014724 000002
3620

```

```

;*****
;THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
;SAVE THE ERROR ITEM NUMBER AND ADDRESS OF THE ERROR CALL
;AND GO TO $ERRTYP ON ERROR
;THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
;SW15=1      HALT ON ERROR
;SW13=1      INHIBIT ERROR TYPEOUTS
;SM09=1      LOOP IN ERROR
;CALL
;      ERROR  *N      ;ERROR=ENT AND N=ERROR ITEM NUMBER
;*****

$ERROR:
7:      INCB      $ERFLG      ;SET THE ERROR FLAG
      BEQ      7:      ;DON'T LET FLAG GO TO ZERO
      MOV      $TSTM,$DISPLAY ;DISPLAY TEST NUMBER AND ERROR FLAG
      INC      $ERTTL      ;INCREMENT ERROR COUNT
      MOV      (SP),$ERRPC    ;GET ADDRESS OF ERROR INSTRUCTION
      SUB      $2,$ERRPC
      MOVB    $ERRPC,$ITEMB   ;STRIP AND SAVE THE ERROR ITEM CODE
      BIT      $BIT13,$SMR    ;SKIP TYPEOUT IF SET
      BNE     20:           ;SKIP TYPEOUTS
      JSR     PC,$ERRTYP     ;GO TO USER ERROR ROUTINE
      TYPE    , $CRLF

20:
      CNPB    $APTENV,$ENV   ;RUNNING IN APT MODE
      BNE     21:           ;NO, SKIP APT ERROR REPORT
      MOVB    $ITEMB,$21:    ;SET ITEM NUMBER AS ERROR NUMBER
      JSR     PC,$ATY4      ;REPORT FATAL ERROR TO APT

21:
      .BYTE  0
      .BYTE  0

22:
      BR      22:           ;APT ERROR LOOP

2:      TST     $SMR
      BPL     3:           ;SKIP IF CONTINUE
      HALT    ;HALT ON ERROR!

3:      CKSMR
      BIT     $BIT09,$SMR   ;TEST FOR CHANGE IN SOFT-SMR
      BEQ     4:           ;LOOP ON ERROR SWITCH SET?
      MOV     $LPERA,(SP)   ;FUDGE RETURN FOR LOOPING
      TST     $ESCAPE
      BEQ     5:           ;CHECK FOR AN ESCAPE ADDRESS
      MOV     $ESCAPE,(SP) ;FUDGE RETURN ADDRESS FOR ESCAPE

5:      CNP     $ENDAD,$042 ;ACT-11 AUTO-ACCEPT?
      BNE     6:           ;BR IF NO
      HALT   ;YES

6:      RTI
      ;RETURN

```

3622
 3623

.SBTTL ERROR MESSAGE TYPEOUT ROUTINE

;;*****
 ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" (IITEMB) TO DETERMINE WHICH
 ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" (ERRTB),
 ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.

```

014726                                $ERRTYP:
014726 104401 001063                TYPE      ,%CRLF          ;; "CARRIAGE RETURN" & "LINE FEED"
014732 010046                        MOV        RO,-(SP)      ;; SAVE RO
014734 005000                        CLR        RO          ;; PICKUP THE ITEM INDEX
014736 153700 001014                BISB      @IITEMB,RO
014742 001004                        BNE       1%          ;; IF ITEM NUMBER IS ZERO, JUST
                                ;; TYPE THE PC OF THE ERROR
014744 013746 001016                MOV        $ERRPC,-(SP) ;; SAVE $ERRPC FOR TYPEOUT
                                ;; ERROR ADDRESS
014750 104402                        TYPCC
014752 000434                        BR         6%          ;; GO TYPE- OCTAL ASCII(ALL DIGITS)
J14754 122700 000177                1%:      CHPB      @177,RO  ;; GET OUT
014760 001003                        BNE       100%       ;; IS THIS THE POWER MONITOR BIT CALL      ;DPM001
014762 012700 015076                MOV        @PFECWS,RO ;; BRANCH IF NOT                          ;DPM001
014766 000406                        BR         110%      ;; MOVE ADDR OF PWR MONITOR BIT ERR TO RO ;DPM001
014770 005300                        100%:    DEC        RO          ;; BRANCH TO CALL THE ERROR                ;DPM001
014772 006300                        ASL       RO          ;; ADJUST THE INDEX SO THAT IT WILL
014774 006300                        ASL       RO          ;; WORK FOR THE ERROR TABLE
014776 006300                        ASL       RO
015000 062700 001146                ADD        @ERRTB,RO  ;; FORM TABLE POINTER
015004 012037 015014                110%:   MOV        (RO)+,2% ;; PICKUP "ERROR MESSAGE" POINTER
015010 001404                        BEQ       3%          ;; SKIP TYPEOUT IF NO POINTER
015012 104401                        TYPE
015014 000000                        2%:      .WORD      0      ;; TYPE THE "ERROR MESSAGE"
015016 104401 001063                TYPE      ,%CRLF          ;; "ERROR MESSAGE" POINTER GOES HERE
015022 012037 015032                3%:      MOV        (RO)+,4% ;; "CARRIAGE RETURN" & "LINE FEED"
015026 001404                        BEQ       5%          ;; PICKUP "DATA HEADER" POINTER
015030 104401                        TYPE
015032 000000                        4%:      .WORD      0      ;; SKIP TYPEOUT IF 0
015034 104401 001063                TYPE      ,%CRLF          ;; TYPE THE "DATA HEADER"
015040 011000                        5%:      MOV        (RO),RO     ;; "DATA HEADER" POINTER GOES HERE
015042 001004                        BNE       7%          ;; "CARRIAGE RETURN" & "LINE FEED"
015044 012600                        6%:      MOV        (SP)+,RO    ;; PICKUP "DATA TABLE" POINTER
015046 104401 001063                TYPE      ,%CRLF          ;; GO TYPE THE DATA
015052 000207                        RTS       PC           ;; RESTORE RO
015054                                7%:      ;; "CARRIAGE RETURN" & "LINE FEED"
                                ;; RETURN
    
```

```
015054 013046      MOV      B(R0),-(SP)    ;;SAVE B(R0). FOR TYPEOUT
015056 104402      TYPOC                      ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
015060 005710      TST      (R0)          ;;IS THERE ANOTHER NUMBER?
015062 001770      BEQ      6$           ;;BR IF NO
015064 104401 015072  TYPE     .8$          ;;TYPE TWO(2) SPACES
015070 000771      BR       7$           ;;LOOP
015072   040     040     000  8$:  .ASCIZ  / /           ;;TWO(2) SPACES
                                .EVEN
015076 015106 015146 015176 PFECWS: .WORD  PFECEN,PFECDH,PFECDT,PFECDF
015106   120     117     127 PFECEN: .ASCIZ  ?POWER MONITOR BIT WAS FOUND SET?
015146   124     105     123 PFECDH: .ASCIZ  ?TESTNO ERR PC CPUERR?
                                .EVEN
015176 001072 001016 015674 PFECDT: .WORD  $TESTN,$ERRPC,CPSAVE,0
015206   000     000     000 PFECDF: .BYTE  0,0,0,0
```

3624

```

3626
3627
3628
3629
3630
3631 015212 012737 015356 000024
3632 015220 012737 000340 000026
3633 015226 010046
3634 015230 010146
3635 015232 010246
3636 015234 010346
3637 015236 010446
3638 015240 010546
3639 015242 017746 163572
3640 015246 010637 015362
3641 015252 012737 015264 000024
3642 015260 000000
3643 015262 000776
3644
3645
3646
3647
3648
3649 015264 012737 015356 000024
3650 015272 013706 015362
3651 015276 012677 163536
3652 015302 012605
3653 015304 012604
3654 015306 012603
3655 015310 012602
3656 015312 012601
3657 015314 012600
3658 015316 012737 015212 000024
3659 015324 012737 000340 000026
3660 015332 005037 015362
3661 015336 005237 015362
3662 015342 001375
3663 015344 104401
3664 015346 015364
3665 015350 013716 001006
3666 015354 000002
3667 015356 000000
3668 015360 000776
3669 015362 000000
3670 015364 015 012 120
3671

```

```

.SBTTL POWER DOWN AND UP ROUTINES
;*****
;POWER DOWN ROUTINE
;*****
$PWRDN: MOV @ILLUP,@PWRVEC ;SET FOR FAST UP
MOV @340,@PWRVEC+2 ;PRIO:7
MOV R0,-(SP) ;PUSH R0 ON STACK
MOV R1,-(SP) ;PUSH R1 ON STACK
MOV R2,-(SP) ;PUSH R2 ON STACK
MOV R3,-(SP) ;PUSH R3 ON STACK
MOV R4,-(SP) ;PUSH R4 ON STACK
MOV R5,-(SP) ;PUSH R5 ON STACK
MOV @SMR,-(SP) ;PUSH @SMR ON STACK
MOV SP,$SAVR6 ;SAVE SP
MOV @PWRUP,@PWRVEC ;SET UP VECTOR
HALT
BR .-2 ;HANG UP

;*****
;POWER UP ROUTINE
;*****
$PWRUP: MOV @ILLUP,@PWRVEC ;SET FOR FAST DOWN
MOV $SAVR6,SP ;GET SP
MOV (SP)+,@SMR ;POP STACK INTO @SMR
MOV (SP)+,R5
MOV (SP)+,R4 ;POP STACK INTO R4
MOV (SP)+,R3 ;POP STACK INTO R3
MOV (SP)+,R2 ;POP STACK INTO R2
MOV (SP)+,R1 ;POP STACK INTO R1
MOV (SP)+,R0 ;POP STACK INTO R0
MOV @PWRDN,@PWRVEC ;SET UP THE POWER DOWN VECTOR
MOV @340,@PWRVEC+2 ;PRIO:7
CLR $SAVR6 ;WAIT LOOP FOR THE TTY
1$: INC $SAVR6 ;WAIT FOR THE INC
BNE 1$ ;OF WORD
TYPE ;REPORT THE POWER FAILURE
.PWRNG: .WORD $POWER ;POWER FAIL MESSAGE POINTER
MOV $LPADR,(SP) ;CHOCOLATE FUDGE RETURN TO BEGINNING OF TEST
RTI ;RETURN TO THERE
$ILLUP: HALT ;THE POWER UP SEQUENCE WAS STARTED
BR .-2 ; BEFORE THE POWER DOWN WAS COMPLETE
$SAVR6: 0 ;PUT THE SP HERE
$POWER: .ASCIZ <15><12>"POWER"

```


3673
3674

.SBTTL SCOPE HANDLER ROUTINE

```

;*****
;THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
;AND LOAD THE TEST NUMBER(ISTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
;AND LOAD THE ERROR FLAG (ERFLG) INTO DISPLAY<15:08>
;THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
;SW14=1      LOOP ON TEST
;SW09=1      LOOP ON ERROR
;CALL
;*          SCOPE          ;SCOPE=IOT

015374          ;SCOPE:
015374 104406          CKSMR          ;TEST FOR CHANGE IN SOFT-SMR
015376 032777 040000 163434 1: BIT #BIT14,BSMR ;LOOP ON PRESENT TEST?
015404 001125          BNE $OVER ;YES IF SW14=1

;*****START OF CODE FOR THE XOR TESTER*****
015406 000416          XTSTR: BR 6: ;IF RUNNING ON THE "XOR" TESTER CHANGE
;THIS INSTRUCTION TO A "NOP" (NOP=240)
015410 013746 000004          MOV $ERRVEC,-(SP) ;SAVE THE CONTENTS OF THE ERROR VECTOR
015414 012737 015434 000004          MOV #5,$ERRVEC ;SET FOR TIMEOUT
015422 003737 177060          TST $177060 ;TIME OUT ON XOR?
015426 012637 000004          MOV (SP)+,$ERRVEC ;RESTORE THE ERROR VECTOR
015432 000474          BR $SVLAD ;GO TO THE NEXT TEST
015434 022626          5: CMP (SP)+,(SP)+ ;CLEAR THE STACK AFTER A TIME OUT
015436 012637 000004          MOV (SP)+,$ERRVEC ;RESTORE THE ERROR VECTOR
015442 000462          BR 7: ;LOOP ON THE PRESENT TEST
015444          6: ;*****END OF CODE FOR THE XOR TESTER*****
015444 022737 177777 015674 2: CMP #-1,CPSAVE ;SEE IF TIMEOUT WAS PREVIOUSLY RECORDED ;DPH001
015452 001447          BEQ 2001: ;BRANCH AROUND ROUTINE IF SO ;DPH001
015454 005227 177777          INC #-1 ;TEST FOR 1ST TIME THROUGH ;DPH001
015460 001005          BNE 900: ;BRANCH IF NOT ;DPH001
015462 013746 000004          MOV 4,-(SP) ;SAVE TIMEOUT VECTOR AT 4 ;DPH001
015466 012737 015542 000004          MOV #1999,$4 ;SET VECTOR TO LOCATION BELOW ;DPH001
015474 013737 177766 015674 900: MOV 177766,CPSAVE ;MOVE CPU ERR REG VALUE TO LOC FOR TST ;DPH001
015502 032737 000001 015674          BIT #BIT00,CPSAVE ;SEE IF THE POWER MONITOR BIT IS ON ;DPH001
015510 001423          BEQ 2000: ;BRANCH TO CONTINUE ROUTINE IF CLEAR ;DPH001
015512 042737 000001 177766          BIC #BIT00,177766 ;CLEAR THE BIT FOUND TO BE SET ;DPH001
015520 017746 163314          MOV $SMR,-(SP) ;SAVE SMR VALUE ;DPH001
015524 042777 001000 163306          BIC #BIT09,$SMR ;DON'T ALLOW LOOP ON ERROR ON THIS ERROR ;DPH001
015532 104177          ENT +177 ;CALL SPECIAL POWER FAIL BIT ERROR CALL ;DPH001
015534 012677 163300          MOV (SP)+,$SMR ;RESTORE SMR TO ORIGINAL VALUE ;DPH001
015540 000407          BR 2000: ;BRANCH OVER TIMEOUT SECTION ;DPH001
015542 022626          1999: CMP (SP)+,(SP)+ ;CLEAN UP THE STACK AFTER TIMEOUT ;DPH001
015544 012737 177777 015674          MOV #-1,CPSAVE ;MOVE -1 TO CPSAVE - NO CPU ERR REG ;DPH001
015552 005237 015562          INC 2000+2 ;INCREMENT 1ST TIME THROUGH LOCATION ;DPH001
015556 000403          BR 910: ;BRANCH OVER 1ST TIME THROUGH TEST ;DPH001
015560 005227 177777          2000: INC #-1 ;TEST FOR 1ST TIME THROUGH ;DPH001
015564 001002          BNE 2001: ;BRANCH IF NOT ;DPH001
015566 012637 000004          910: MOV (SP)+,$4 ;RESTORE LOCATION 4 ;DPH001
015572 105737 001003          2001: TSTB $ERFLG ;HAS AN ERROR OCCURRED?
015576 001412          BEQ $SVLAD ;BR IF NO
015600 032777 001000 163232          BIT #BIT09,BSMR ;LOOP ON ERROR?
015606 001404          BEQ 4: ;BR IF NO
015610 013737 001010 001006 7: MOV $LPERR,$LPADR ;SET LOOP ADDRESS TO LAST SCOPE
015616 000420          BR $OVER

```

```
015620 105037 001003      4:   CLRB   $ERFLG      ;; ZERO THE ERROR FLAG
015624 105237 001002      $SVLAD: INCB  $TSTNM      ;; COUNT TEST NUMBERS
015630 113737 001002 001072      MOVB   $TSTNM,$TESTN  ;; SET TEST NUMBER IN APT MAILBOX
015636 011637 001006      MOV    (SP), $LPADR   ;; SAVE SCOPE LOOP ADDRESS
015642 011637 001010      MOV    (SP), $LPERR   ;; SAVE ERROR LOOP ADDRESS
015646 005037 001060      CLR    $ESCAPE       ;; CLEAR THE ESCAPE FROM ERROR ADDRESS
015652 112737 000001 001015      MOVB   @1,$ERMAX     ;; ONLY ALLOW ONE(1) ERROR ON NEXT TEST
015660 013777 001002 163154 $OVER:  MOV    $TSTNM,$DISPLAY ;; DISPLAY TEST NUMBER
015666 013716 001006      MOV    $LPADR,(SP)   ;; FUDGE RETURN ADDRESS
015672 000002      RTI                  ;; FIXES PS
015674 000000      CPSAVE: .WORD 0     ;; LOCATION TO SAVE CPU ERR REG CONTENTS ;DPH00:
```

3675

```

3677
3678
3679
3680
3681
3682 015676 112737 000001 016142 $ATY1: MOVB #1,$FFLG ;TO REPORT FATAL ERROR
3683 015704 112737 000001 016140 $ATY3: MOVB #1,$MFLG ;TO TYPE A MESSAGE
3684 015712 000403 BR $ATYC
3685 015714 112737 000001 016142 $ATY4: MOVB #1,$FFLG ;TO ONLY REPORT FATAL ERROR
3686 015722 $ATYC:
3687 015722 010046 MOV R0,-(SP) ;PUSH R0 ON STACK
3688 015724 010146 MOV R1,-(SP) ;PUSH R1 ON STACK
3689 015726 105737 016140 TSTB $MFLG ;SHOULD TYPE A MESSAGE?
3690 015732 001450 BEQ 5$ ;IF NOT: BR
3691 015734 122737 000001 001106 CMPB $APTENV,$ENV ;OPERATING UNDER APT?
3692 015742 001031 BNE 3$ ;IF NOT: BR
3693 015744 132737 000100 001107 BITB $APTSPOOL,$ENVM ;SHOULD SPOOL MESSAGE?
3694 015752 001425 BEQ 3$ ;IF NOT: BR
3695 015754 017600 000004 MOV $4(SP),R0 ;GET MESSAGE ADDRESS
3696 015760 062766 000002 000004 ADD #2,4(SP) ;BUMP RETURN ADDRESS
3697 015766 005737 001066 1$: TST $MSGTYPE ;SEE IF DONE W/ LAST XMISSION?
3698 015772 001375 BNE 1$ ;IF NOT: WAIT
3699 015774 010037 001102 MOV R0,$MSGAD ;PUT ADDRESS IN MAILBOX
3700 016000 105720 2$: TSTB (R0)+ ;FIND END OF MESSAGE
3701 016002 001376 BNE 2$
3702 016004 163700 001102 SUB $MSGAD,R0 ;SUB START OF MESSAGE
3703 016010 006200 ASR R0 ;GET MESSAGE LENGTH IN WORDS
3704 016012 010037 001104 MOV R0,$MSGLGT ;PUT LENGTH IN MAILBOX
3705 016016 012737 000004 001066 MOV #4,$MSGTYPE ;TELL APT TO TAKE MESSAGE
3706 016024 000413 BR 5$
3707 016026 017637 000004 016052 3$: MOV $4(SP),4$ ;PUT MSG ADDR IN JSR LINKAGE
3708 016034 062766 000002 000004 ADD #2,4(SP) ;BUMP RETURN ADDRESS
3709 016042 013746 177776 MOV 177776,-(SP) ;PUSH 177776 ON STACK
3710 016046 004737 016144 JSR PC,$TYPE ;CALL TYPE MACRO
3711 016052 000000 4$: .WORD 0
3712 016054 5$:
3713 016054 105737 016142 10$: TSTB $FFLG ;SHOULD REPORT FATAL ERROR?
3714 016060 001413 BEQ 12$ ;IF NOT: BR
3715 016062 005737 001106 TST $ENV ;RUNNING UNDER APT?
3716 016066 001410 BEQ 12$ ;IF NOT: BR
3717 016070 005737 001066 11$: TST $MSGTYPE ;FINISHED LAST MESSAGE?
3718 016074 001375 BNE 11$ ;IF NOT: WAIT
3719 016076 017637 000004 001070 MOV $4(SP),$FATAL ;GET ERROR #
3720 016104 005237 001066 INC $MSGTYPE ;TELL APT TO TAKE ERROR
3721 016110 062766 000002 000004 12$: ADD #2,4(SP) ;BUMP RETURN ADDRESS
3722 016116 105037 016142 CLRB $FFLG ;CLEAR FATAL FLAG
3723 016122 105037 016141 CLRB $LFLG ;CLEAR LOG FLAG
3724 016126 105037 016140 CLRB $MFLG ;CLEAR MESSAGE FLAG
3725 016132 012601 MOV (SP)+,R1 ;POP STACK INTO R1
3726 016134 012600 MOV (SP)+,R0 ;POP STACK INTO R1
3727 016136 000207 RTS PC ;RETURN
    
```

3728	016140	000	%MFLG: .BYTE	0	
3729	016141	000	%LFLG: .BYTE	0	;LOG FLAG
3730	016142	000	%FFLG: .BYTE	0	;FATAL FLAG
3731					
3732			.EVEN		
3733	000200		APTSIZE=200		
3734	000001		APTENV=001		
3735	000100		APTPOOL=100		
3736	000040		APTCSUP=040		
3737					

3739
3740

```

.SBTTL TYPE ROUTINE
;*****
;ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
;THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
;NOTE1:      #NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
;NOTE2:      #FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
;NOTE3:      #FILLC CONTAINS THE CHARACTER TO FILL AFTER.
;
;CALL:
;1) USING A TRAP INSTRUCTION
;*      TYPE      ,MESADR      ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
;OR
;*      TYPE
;*      MESADR
;
016144 105737 001057 $TYPE: TSTB  #TPFLG      ;;IS THERE A TERMINAL?
016150 100002          BPL  1#          ;;BR IF YES
016152 000000          HALT          ;;HALT HERE IF NO TERMINAL
016154 000430          BR  3#          ;;LEAVE
016156 010046          1#:  MOV  RO,-(SP)      ;;SAVE RO
016160 017600 000002  MOV  @2(SP),RO      ;;GET ADDRESS OF ASCIZ STRING
016164 122737 000001 001106  CMPB @APTENV,#ENV      ;;RUNNING IN APT MODE
016172 001011          BNE  62#          ;;NO,GO CHECK FOR APT CONSOLE
016174 132737 000100 001107  BITB @APTPOOL,#ENVH  ;;SPOOL MESSAGE TO APT
016202 001405          BEQ  62#          ;;NO,GO CHECK FOR CONSOLE
016204 010037 016214  MOV  RO,61#          ;;SETUP MESSAGE ADDRESS FOR APT
016210 004737 015704  JSR  PC,#ATY3      ;;SPOOL MESSAGE TO APT
016214 000000          61#: .WORD 0      ;;MESSAGE ADDRESS
016216 132737 000040 001107  62#: BITB @APTCSUP,#ENVH ;;APT CONSOLE SUPPRESSED
016224 001003          BNE  60#          ;;YES,SKIP TYPE OUT
016226 112046          2#:  MOVB (RO)+,-(SP)      ;;PUSH CHARACTER TO BE TYPED ONTO STACK
016230 001005          BNE  4#          ;;BR IF IT ISN'T THE TERMINATOR
016232 005726          TST  (SP)+          ;;IF TERMINATOR POP IT OFF THE STACK
016234 012600          60#:  MOV  (SP)+,RO      ;;RESTORE RO
016236 062716 000002  3#:  ADD  @2,(SP)      ;;ADJUST RETURN PC
016242 000002          RTI          ;;RETURN
016244 122716 000011  4#:  CMPB @H1,(SP)      ;;BRANCH IF <HT>
016250 001430          BEQ  8#          ;;BRANCH IF NOT <CRLF>
016252 122716 000200  CMPB @CRLF,(SP)
016256 001006          BNE  5#          ;;POP <CR><LF> EQUIV
016260 005726          TST  (SP)+          ;;TYPE A CR AND LF
016262 104401          TYPE
016264 001063          #CRLF
016266 105037 016504  CLRB #CHARCNT      ;;CLEAR CHARACTER COUNT
016272 000755          BR  2#          ;;GET NEXT CHARACTER
016274 004737 016356  5#:  JSR  PC,#TYPEC      ;;GO TYPE THIS CHARACTER
016300 123726 001056  6#:  CMPB #FILLC,(SP)+  ;;IS IT TIME FOR FILLER CHARS.?
016304 001350          BNE  2#          ;;IF NO GO GET NEXT CHAR.
016306 013746 001054  MOV  #NULL,-(SP)    ;;GET # OF FILLER CHARS. NEEDED
                                ;;AND THE NULL CHAR.
016312 105366 000001  7#:  DECB 1(SP)      ;;DOES A NULL NEED TO BE TYPED?
016316 002770          BLT  6#          ;;BR IF NO--GO POP THE NULL OFF OF STACK
016320 004737 016356  JSR  PC,#TYPEC      ;;GO TYPE A NULL
016324 105337 016504  DECB #CHARCNT      ;;DO NOT COUNT AS A COUNT
016330 000770          BR  7#          ;;LOOP
;HORIZONTAL TAB PROCESSOR

```

```

016332 112716 000040      8:  MOV  #' ,(SP)      ;;REPLACE TAB WITH SPACE
016336 004737 016356      9:  JSR  PC,$TYPEC    ;;TYPE A SPACE
016342 132737 000007 016504  BITS  #7,$CHARCNT  ;;BRANCH IF NOT AT
016350 001372          BNE  9:            ;;TAB STOP
016352 005726          TST  (SP)         ;;POP SPACE OFF STACK
016354 000724          BR   2:            ;;GET NEXT CHARACTER
016356          $TYPEC:
016356 105777 162462      TSTB  @TKS        ;;CHAR IN KYBD BUFFER?      ;MJD001
016362 100022          BPL  10:         ;;BR IF NOT                ;MJD001
016364 017746 162456      MOV   @TKB,-(SP)   ;;GET CHAR                  ;MJD001
016370 042716 177600      BIC   @177600,(SP) ;;STRIP EXTRANEIOUS BITS   ;MJD001
016374 122716 000023      CHPB  @XOFF,(SP)  ;;WAS CHAR XOFF            ;MJD001
016400 001012          BNE  102:        ;;BR IF NOT                ;MJD001
016402          101:
016402 105777 162436      TSTB  @TKS        ;;WAIT FOR CHAR             ;MJD001
016406 100375          BPL  101:        ;;BR IF NOT                ;MJD001
016410 117716 162432      MOVB  @TKB,(SP)   ;;GET CHAR                  ;MJD001
016414 042716 177600      BIC   @177600,(SP) ;;STRIP IT                  ;MJD001
016420 122716 000021      CHPB  @XON,(SP)   ;;WAS IT XON?              ;MJD001
016424 001366          BNE  101:        ;;BR IF NOT                ;MJD001
016426          102:
016426 005726          TST  (SP)         ;;FIX STACK                 ;MJD001
016430          10:
016430 105777 162414      TSTB  @TPS        ;;WAIT UNTIL PRINTER IS READY ;MJD001
016434 100375          BPL  10:         ;;BR IF NOT                ;MJD001
016436 126627 000002 000021  CHPB  2(SP),@XON  ;;IS CHARACTER A RANDOM XON? ;RAN001
016444 001420          BEQ  $TYPEX      ;;BRANCH IF YES           ;RAN001
016446 116677 000002 162376  MOVB  2(SP),@TPB  ;;LOAD CHAR TO BE TYPED INTO DATA REG.
016454 122766 000015 000002  CHPB  @CR,2(SP)   ;;IS CHARACTER A CARRIAGE RETURN?
016462 001003          BNE  1:         ;;BRANCH IF NO
016464 105037 016504      CLRB  $CHARCNT    ;;YES--CLEAR CHARACTER COUNT
016470 000406          BR   $TYPEX      ;;EXIT
016472 122766 000012 000002  1:  CHPB  @LF,2(SP)   ;;IS CHARACTER A LINE FEED?
016500 001402          BEQ  $TYPEX      ;;BRANCH IF YES
016502 105227          INCB (PC)       ;;COUNT THE CHARACTER
016504 000000          $CHARCNT: .WORD 0 ;;CHARACTER COUNT STORAGE
016506 000207          $TYPEX: RTS PC

```

3741

```

.SBTL  BINARY TO OCTAL (ASCII) AND TYPE
;*****
;THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
;OCTAL (ASCII) NUMBER AND TYPE IT.
;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
;CALL:
;*  MOV    NUM,-(SP)      ;;NUMBER TO BE TYPED
;*  TYPOS          ;;CALL FOR TYPEOUT
;*  .BYTE  N            ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
;*  .BYTE  M            ;;M=1 OR 0
;*                          ;;1=TYPE LEADING ZEROS
;*                          ;;0=SUPPRESS LEADING ZEROS
;*
;*$TYPON---ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
;*$TYPOS OR $TYPOC
;CALL:
;*  MOV    NUM,-(SP)      ;;NUMBER TO BE TYPED
;*  TYPON          ;;CALL FOR TYPEOUT
;*
;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER

```

```

; *CALL:
; *      MOV      NUM, -(SP)      ;; NUMBER TO BE TYPED
; *      TYP0C
; *      $TYP0S: MOV      8(SP), -(SP)  ;; CALL FOR TYPEOUT
; *              MOV8    1(SP), $0FILL  ;; PICKUP THE MODE
; *              MOV8    (SP)+, $0MODE+1  ;; LOAD ZERO FILL SWITCH
; *              ADD     #2, (SP)      ;; NUMBER OF DIGITS TO TYPE
; *              BR      $TYP0N      ;; ADJUST RETURN ADDRESS
016510 017646 000000      016733 $TYP0C: MOV8    #1, $0FILL      ;; SET THE ZERO FILL SWITCH
016514 116637 000001      016735      MOV8    #6, $0MODE+1      ;; SET FOR SIX(6) DIGITS
016522 112637 016735      $TYP0N: MOV8    #5, $0CNT      ;; SET THE ITERATION COUNT
016526 062716 000002      MOV      R3, -(SP)      ;; SAVE R3
016532 000406      MOV      R4, -(SP)      ;; SAVE R4
016534 112737 000001      MOV      R5, -(SP)      ;; SAVE R5
016542 112737 000006      MOV8    $0MODE+1, R4      ;; GET THE NUMBER OF DIGITS TO TYPE
016550 112737 000005      NEG     R4      ;; SUBTRACT IT FOR MAX. ALLOWED
016556 010346      ADD     #6, R4      ;; SAVE IT FOR USE
016560 010446      MOV8    R4, $0MODE      ;; GET THE ZERO FILL SWITCH
016562 010546      MOV8    $0FILL, R4      ;; PICKUP THE INPUT NUMBER
016564 113704 016735      MOV     12(SP), R5      ;; CLEAR THE OUTPUT WORD
016570 005404      CLR    R3      ;; ROTATE MSB INTO "C"
016572 062704 000006      1#:    ROL    R5      ;; GO DO MSB
016576 110437 016734      BR     3#      ;; FORM THIS DIGIT
016602 113704 016733      2#:    ROL    R5
016606 016605 000012      ROL    R5
016612 005003      MOV    R5, R3
016614 006105      3#:    ROL    R3      ;; GET LSB OF THIS DIGIT
016616 000404      DECB   $0MODE      ;; TYPE THIS DIGIT?
016620 006105      BPL    7#      ;; BR IF NO
016622 006105      BIC    #177770, R3      ;; GET RID OF JUNK
016624 006105      ONE   4#      ;; TEST FOR 0
016626 010503      TST   R4      ;; SUPPRESS THIS 0?
016630 006103      BEQ   5#      ;; BR IF YES
016632 105337 016734      4#:    INC    R4      ;; DON'T SUPPRESS ANYMORE 0'S
016636 100016      BIS   #'0, R3      ;; MAKE THIS DIGIT ASCII
016640 042703 177770      5#:    BIS   #' , R3      ;; MAKE ASCII IF NOT ALREADY
016644 001002      MOV8  R3, 8#      ;; SAVE FOR TYPING
016646 005704      TYPE  .8#      ;; GO TYPE THIS DIGIT
016650 001403      7#:    DECB  $0CNT      ;; COUNT BY 1
016652 005204      BGT   2#      ;; BR IF MORE TO DO
016654 052703 000060      BLT   6#      ;; BR IF DONE
016660 052703 000040      INC   R4      ;; INSURE LAST DIGIT ISN'T A BLANK
016664 110337 016730      BR    2#      ;; GO DO THE LAST DIGIT
016670 104401 016730      6#:    MOV   (SP)+, R5      ;; RESTORE R5
016674 105337 016732      MOV   (SP)+, R4      ;; RESTORE R4
016700 003347      MOV   (SP)+, R3      ;; RESTORE R3
016702 002402      MOV   2(SP), 4(SP)      ;; SET THE STACK FOR RETURNING
016704 005204      MOV   (SP)+, (SP)
016706 000744      RTI
016710 012605      8#:    .BYTE 0      ;; RETURN
016712 012604      ;; STORAGE FOR ASCII DIGIT
016714 012603      ;; TERMINATOR FOR TYPE ROUTINE
016716 016666 000002 000004      $0CNT: .BYTE 0      ;; OCTAL DIGIT COUNTER
016724 012616      $0FILL: .BYTE 0      ;; ZERO FILL SWITCH
016726 000002      $0MODE: .WORD 0      ;; NUMBER OF DIGITS TO TYPE
016730 000
016731 000
016732 000
016733 000
016734 000000
    
```

3744
 3745

```

.SBTTL TTY INPUT ROUTINE
;*****
.ENABL LSB
;*****
;SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
;ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
;SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
;WHEN OPERATING IN TTY FLAG MODE.
016736 022737 000176 001040 $CKSMR: CMP #SMREG,SMR ;;IS THE SOFT-SMR SELECTED?
016744 001074 BNE 15# ;;BRANCH IF NO
016746 105777 162072 TSTB @TKS ;;CHAR THERE?
016752 100071 BPL 15# ;;IF NO, DON'T WAIT AROUND
016754 117746 162066 MOVB @TKB,-(SP) ;;SAVE THE CHAR
016760 042716 177600 BIC @C177,(SP) ;;STRIP-OFF THE ASCII
016764 022726 000007 CMP #7,(SP)+ ;;IS IT A CONTROL G?
016770 001062 BNE 15# ;;NO, RETURN TO USER
016772 123727 001034 000001 CMPB $AUTOB,#1 ;;ARE WE RUNNING IN AUTO-MODE?
017000 001456 BEQ 15# ;;BRANCH IF YES
017002 104401 017473 TYPE ,#CNTLG ;;ECHO THE CONTROL-G (+G)
017006 104401 017500 $GTSMR: TYPE ,#MSMR ;;TYPE CURRENT CONTENTS
017012 013746 000176 MOV SMREG,-(SP) ;;SAVE SMREG FOR TYPEOUT
017016 104402 TYPOC ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
017020 104401 017511 TYPE ,#MNEW ;;PROMPT FOR NEW SMR
017024 005046 19#: CLR -(SP) ;;CLEAR COUNTER
017026 005046 CLR -(SP) ;;THE NEW SMR
017030 105777 162010 7#: TSTB @TKS ;;CHAR THERE?
017034 100375 BPL 7# ;;IF NOT TRY AGAIN
017036 117746 162004 MOVB @TKB,-(SP) ;;PICK UP CHAR
017042 042716 177600 BIC @C177,(SP) ;;MAKE IT 7-BIT ASCII
017046 021627 000025 9#: CMP (SP),#25 ;;IS IT A CONTROL-U?
017052 001005 BNE 10# ;;BRANCH IF NOT
017054 104401 017466 TYPE ,#CNTLU ;;YES, ECHO CONTROL-U (+U)
017060 062706 000006 20#: ADD #6,SP ;;IGNORE PREVIOUS INPUT
017064 000757 BR 19# ;;LET'S TRY IT AGAIN
017066 021627 000015 10#: CMP (SP),#15 ;;IS IT A <CR>?
017072 001022 BNE 16# ;;BRANCH IF NO
017074 005766 000004 TST 4(SP) ;;YES, IS IT THE FIRST CHAR?
017100 001403 BEQ 11# ;;BRANCH IF YES
017102 016677 000002 161730 MOV 2(SP),BSMR ;;SAVE NEW SMR
017110 062706 000006 11#: ADD #6,SP ;;CLEAR UP STACK
017114 104401 001063 14#: TYPE ,#CRLF ;;ECHO <CR> AND <LF>
017120 123727 001035 000001 CMPB $INTAG,#1 ;;RE-ENABLE TTY KBD INTERRUPTS?
017126 001003 BNE 15# ;;BRANCH IF NOT
017130 012777 000100 161706 MOV #100,@TKS ;;RE-ENABLE TTY KBD INTERRUPTS
017136 000002 15#: RTI ;;RETURN
017140 004737 016356 16#: JSR PC,#TYPEC ;;ECHO CHAR
017144 021627 000060 CMP (SP),#60 ;;CHAR < 0?
017150 002420 BLT 18# ;;BRANCH IF YES
017152 021627 000067 CMP (SP),#67 ;;CHAR > 7?
017156 003015 BGT 18# ;;BRANCH IF YES
017160 042726 000060 BIC #60,(SP)+ ;;STRIP-OFF ASCII
017164 005766 000002 TST 2(SP) ;;IS THIS THE FIRST CHAR
017170 001403 BEQ 17# ;;BRANCH IF YES
017172 006316 ASL (SP) ;;NO, SHIFT PRESENT
017174 006316 ASL (SP) ;; CHAR OVER TO MAKE
017176 006316 ASL (SP) ;; ROOM FOR NEW ONE.
    
```


TTY INPUT ROUTINE

```

017200 005266 000002      17:  INC      2(SP)      ;;KEEP COUNT OF CHAR
017204 056616 177776      BIS      -2(SP),(SP)  ;;SET IN NEW CHAR
017210 000707              BR       78          ;;GET THE NEXT ONE
017212 104401 001062      18:  TYPE    ,@QUES    ;;TYPE ?<CR><LF>
017216 000720              BR       208        ;;SIMULATE CONTROL-U
      .DSABL  LSB
      ;;*****
      ;;THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
      ;;CALL:
      ;;      RDCHR          ;;INPUT A SINGLE CHARACTER FROM THE TTY
      ;;      RETURN HERE   ;;CHARACTER IS ON THE STACK
      ;;      WITH PARITY BIT STRIPPED OFF
      ;;
017220 011646              RDCHR:  MOV      (SP),-(SP)  ;;PUSH DOWN THE PC
017222 016666 000004 000002  MOV      4(SP),2(SP)  ;;SAVE THE PS
017230 105777 161610      1:  TSTB    @TKS        ;;WAIT FOR
017234 100375              BPL      18          ;;A CHARACTER
017236 117766 161604 000004  MOVB    @TKB,4(SP)   ;;READ THE TTY
017244 042766 177600 000004  BIC     @C<177>,4(SP) ;;GET RID OF JUNK IF ANY
017252 026627 000004 000023  CMP     4(SP),@23    ;;IS IT A CONTROL-S?
017260 001013              BNE     38          ;;BRANCH IF NO
017262 105777 161556      2:  TSTB    @TKS        ;;WAIT FOR A CHARACTER
017266 100375              BPL      28          ;;LOOP UNTIL ITS THERE
017270 117746 161552      MOVB    @TKB,-(SP)   ;;GET CHARACTER
017274 042716 177600      BIC     @C177,(SP)  ;;MAKE IT 7-BIT ASCII
017300 022627 000021      CMP     (SP),@21    ;;IS IT A CONTROL-Q?
017304 001366              BNE     28          ;;IF NOT DISCARD IT
017306 000750              BR       18          ;;YES, RESUME
017310 026627 000004 000021  3:  CMP     4(SP),@XON  ;;IS IT A RANDOM XON?
017316 001744              BEQ     18          ;;BRANCH IF YES
017320 026627 000004 000140  CMP     4(SP),@140  ;;IS IT UPPER CASE?
017326 002407              BLT     48          ;;BRANCH IF YES
017330 026627 000004 000175  CMP     4(SP),@175  ;;IS IT A SPECIAL CHAR?
017336 003003              BGT     48          ;;BRANCH IF YES
017340 042766 000040 000004  BIC     @40,4(SP)   ;;MAKE IT UPPER CASE
017346 000002      4:  RTI          ;;GO BACK TO USER
      ;;*****
      ;;THIS ROUTINE WILL INPUT A STRING FROM THE TTY
      ;;CALL:
      ;;      RDLIN          ;;INPUT A STRING FROM THE TTY
      ;;      RETURN HERE   ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
      ;;      TERMINATOR WILL BE A BYTE OF ALL 0'S
      ;;
017350 010346              RDLIN:  MOV      R3,-(SP)  ;;SAVE R3
017352 012703 017456      1:  MOV     @TTYIN,R3   ;;GET ADDRESS
017356 022703 017466      2:  CMP     @TTYIN+8.,R3 ;;BUFFER FULL?
017362 101405              BLOS    48          ;;BR IF YES
017364 104407              RDCHR   ;;GO HEAD ONE CHARACTER FROM THE TTY
017366 112613              MOVB    (SP),,(R3)  ;;GET CHARACTER
017370 122713 000177      10:  CMPB   @177,(R3)   ;;IS IT A RUBOUT
017374 001003              BNE     38          ;;SKIP IF NOT
017376 104401 001062      4:  TYPE    ,@QUES    ;;TYPE A '?'
017402 000763              BR       18          ;;CLEAR THE BUFFER AND LOOP
017404 111337 017454      3:  MOVB    (R3),98    ;;ECHO THE CHARACTER
017410 104401 017454      TYPE    ,98
017414 122723 000015      CMPB   @15,(R3).   ;;CHECK FOR RETURN
017420 001356              BNE     28          ;;LOOP IF NOT RETURN
017422 105063 177777      CLRB   -1(R3)     ;;CLEAR RETURN (THE 15)

```

;RAN001
;RAN001

017426	104401	001064			TYPE	,BLF		;;TYPE A LINE FEED
017432	012603				MOV	(SP),R3		;;RESTORE R3
017434	011646				MOV	(SP),-(SP)		;;ADJUST THE STACK AND PUT ADDRESS OF THE
017436	016666	000004	000002		MOV	4(SP),2(SP)		;; FIRST ASCII CHARACTER ON IT
017444	012766	017456	000004		MOV	@TTYIN,4(SP)		
017452	000002				RTI			;;RETURN
017454	000			96:	.BYTE	0		;;STORAGE FOR ASCII CHAR. TO TYPE
017455	000				.BYTE	0		;;TERMINATOR
017456				\$TTYIN:	.BLKB	8.		;;RESERVE 8 BYTES FOR TTY INPUT
017466	136	125	015	\$CNTLU:	.ASCIZ	/TU/<15><12>		;;CONTROL "U"
017473	136	107	015	\$CNTLG:	.ASCIZ	/TG/<15><12>		;;CONTROL "G"
017500	015	012	123	\$MSMR:	.ASCIZ	<15><12>/SMR = /		
017511	040	040	116	\$MNEW:	.ASCIZ	/ NEW = /		

3746

3748
3749
3750

.SBTTL TRAP DECODER

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

017522 010046
017524 016600 000002
017530 005740
017532 111000
017534 006300
017536 016000 017556
017542 000200

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

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

017544 011646
017546 016666 000004 000002
017554 000002

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

.SBTTL TRAP TABLE

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

; ROUTINE

017556 017544
017560 016144
017562 016534
017564 016510
017566 016550
017570 017006
017572 016736
017574 017220
017576 017350

```
$TRPAD: .WORD  $TRAP2
        $TYPE  ;;CALL=TYPE      TRAP+1(104401) TTY TYPEOUT ROUTINE
        $TYPOC ;;CALL=TYPOC    TRAP+2(104402) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
        $TYPOS ;;CALL=TYPOS    TRAP+3(104403) TYPE OCTAL NUMBER (NO LEADING ZEROS)
        $TYPON ;;CALL=TYPON    TRAP+4(104404) TYPE OCTAL NUMBER (AS PER LAST CALL)
        $GTSWR ;;CALL=GTSWR    TRAP+5(104405) GET SOFT-SWR SETTING
        $CKSMR ;;CALL=CKSMR    TRAP+6(104406) TEST FOR CHANGE IN SOFT-SWR
        $RDCHR ;;CALL=RDCHR    TRAP+7(104407) TTY TYPEIN CHARACTER ROUTINE
        $RDLIN ;;CALL=RDLIN    TRAP+10(104410) TTY TYPEIN STRING ROUTINE
```

3751

```

3753          .SBTTL  ECHO TEST
3754          ;*****
3755          ;*THIS ROUTINE WILL ECHO ANY CHARACTER TYPED
3756          ;*AT THE CONSOLE
3757          ;*THE TEST IS HALTED BY TYPING A CONTROL-C
3758          ;*TEST CAN BE RESTARTED AFTER HALTING BY JUST CONTINUING
3759          ;*****
3760 017600    ECHO:
3761 017600    000005    RESET          ;CLEAR EVERYTHING
3762 017602    112777    000052    163052    MOVB      #'*,BCTBUF    ;TRANSMIT PROMPT "*"
3763 017610    105777    163040    2#:    TSTB      BCTCSR          ;WAIT FOR INPUT
3764 017614    100375          BPL          2#
3765 017616    117777    163034    163036    MOVB      BCRBUF,BCTBUF    ;ECHO INPUT
3766 017624    017700    163026    MOV        BCRBUF,RO      ;STORE INPUT
3767 017630    100023          BPL          5#
3768 017632    052701    010000    BIS        #BIT12,R1      ;BR IF "ERROR" NOT SET
3769 017636    030100          BIT          R1,RO        ;SET PARITY ERROR TEST MASK
3770 017640    001403          BEQ          3#
3771 017642    004737    017724    JSR        PC,MSG        ;CHECK FOR PARITY ERROR FLAG
3772 017646    017752          MPAR          ;BR IF NOT SET
3773 017650    006301          3#:    ASL          R1          ;REPORT PARITY ERROR
3774 017652    030100          BIT          R1,RO        ;SHIFT MASK TO TEST "FR" FLAG
3775 017654    001403          BEQ          4#
3776 017656    004737    017724    JSR        PC,MSG        ;TEST FOR FRAMING ERROR FLAG
3777 017662    017763          MFR          ;BR IF NOT SET
3778 017664    006301          4#:    ASL          R1          ;REPSORT FRAMING ERROR
3779 017666    030100          BIT          R1,RO        ;SHIFT MASK TO TEST "OR" FLAG
3780 017670    001403          BEQ          5#
3781 017672    004737    017724    JSR        PC,MSG        ;TEST FOR OVERFLOW ERROR
3782 017676    017775          MOR          ;BR IF NOT SET
3783 017700    042700    000200    5#:    BIC        #BIT7,RO      ;REPORT OVERFLOW ERROR
3784 017704    022700    000003    CMP        #3,RO          ;CLEAR ANY PARITY BIT
3785 017710    001337          BNE          2#
3786 017712    004737    017724    JSR        PC,MSG        ;WAS INPUT CONTROL-C
3787 017716    020010          MSTOP        ;BR IF NOT
3788 017720    000000          HALT          ;REPORT PROGRAM STOP
3789 017722    000726          BR          ECHO          ;END OF TEST HALT
3790
3791
3792 017724    013600          MSG:    MOV        B(SP)+,RO    ;PICK UP MESSAGE POINTER
3793 017726    062746    000002    ADD        #2,-(SP)      ;ADJUST RETURN PC
3794 017732    105777    162722    WAIT:    TSTB      BCTCSR          ;WAIT FOR XMIT DONE
3795 017736    100375          BPL          ;SEND CHARACTER
3796 017740    112077    162716    MOVB      (RO)+,BCTBUF    ;IS THIS END OF MESSAGE?
3797 017744    105710          TSTB      (RO)
3798 017746    001371          BNE        WAIT
3799 017750    000207          RTS         PC          ;BR IF NOT
3800
3801 017752          015          012          120    MPAR:    .ASCIZ  <CR><LF>/PARITY/
3802 017763          015          012          106    MFR:    .ASCIZ  <CR><LF>/FRAMING/
3803 017775          015          012          117    MOR:    .ASCIZ  <CR><LF>/OVERFLOW/
3804 020010          015          012          123    MSTOP:  .ASCIZ  <CR><LF>/STOP/

```

3806
3807
3808
3809
3810
3811
3812
3813
3814
3815
3816
3817
3818
3819
3820
3821
3822
3823
3824
3825
3826
3827
3828
3829
3830
3831
3832
3833
3834
3835
3836
3837
3838
3839
3840
3841
3842
3843
3844
3845
3846

```

.EVEN
.SBTTL  TERMINAL OUTPUT TEST
;*****
;THIS ROUTINE WILL OUTPUT ALL WRITABLE CHARACTERS FOR THE
;THE OCTAL CODE 040 --> 377
;32 CHARACTERS ARE PRINTED ON EACH LINE
;THE PATTERN IS REPEATED EVERY THREE LINES
;
;*****
OUTTST:  MOV    #40,R1          ;LOAD FIRST WRITABLE CHARACTER
1:      MOV    #40,R0          ;LOAD CHAR COUNT PER LINE
2:      TSTB   BCTCSR         ;WAIT FOR DONE
        BPL    2:             ;
        MOV    R1,BCTBUF      ;TRANSMIT A CHARACTER
        INCB   R1             ;INCREMENT CHARACTER CODE
        DEC    R0             ;DECREMENT CHAR COUNT
        BNE   2:             ;BR IF LINE NOT COMPLETE
        JSR   PC,MSG          ;ISSUE CR,LIN FEED
;CRLF
TSTB   BCRCSR                ;ANY CHARACTER RECEIVED?
BMI    4:                    ;BR IF YES
3:     BIT    #BIT7,R1        ;FINISHED ONE PASS OF WRITABLE CHARACTERS?
        BNE   OUTTST         ;BR IF YES
        BR    1:             ;IF NOT WRITE NEXT LINE
4:     MOVB  BCRBUF,BUFF      ;SAVE WHAT EVER IT WAS
        CMPB  #XON,BUFF      ;WAS IT AN XON?
        BEQ   3:             ;BR IF YES, CONTINUE PRINTING...
        CMPB  #XOFF,BUFF     ;WAS IT AN XOFF?
        BEQ   4:             ;BR IF YES, WAIT FOR XON...
        CLR   BCRBUF         ;CLEAR RECEIVER
        HALT                    ;STOP TEST
        BR    OUTTST         ;RESTART TEST IF CONTINUED

BUFF:   .WORD  0
    
```

020020 012701 000040
020024 012700 000040
020030 105777 162624
020034 100375
020036 010177 162620
020042 105201
020044 005300
020046 001370
020050 004737 017724

020054 001063
020056 105777 162572
020062 100404
020064 032701 000200
020070 001353
020072 000754

020074 117737 162556 020132
020102 122737 000021 020132
020110 001765
020112 122737 000023 020132
020120 001765
020122 005077 162530
020126 000000
020130 000733
020132 000000

3848						
3849						
3850 020134	103	101	116	EM1:	.ASCIZ	/CAN NOT ACCESS TCSR/
3851 020160	103	101	116	EM2:	.ASCIZ	/CAN NOT ACCESS TBUF/
3852 020204	124	103	123	EM3:	.ASCIZ	/TCSR DONE NOT CLEARED WITH TBUF FULL/
3853 020251	124	103	123	EM4:	.ASCIZ	/TCSR DONE NOT SET/
3854 020273	124	103	123	EM5:	.ASCIZ	/TCSR DONE NOT SET WITH RESET/
3855 020330	103	101	116	EM6:	.ASCIZ	/CAN NOT ACCESS RCSR/
3856 020354	103	101	116	EM7:	.ASCIZ	/CAN NOT ACCESS RBUF/
3857 020400	103	101	116	EM10:	.ASCIZ	/CAN NOT ACCESS LKS/
3858 020423	102	111	124	EM11:	.ASCIZ	/BIT0 OF TCSR NOT CLEAR AFTER RESET/
3859 020466	103	101	116	EM12:	.ASCIZ	/CAN NOT SET BIT0 OF TCSR/
3860 020517	103	101	116	EM13:	.ASCIZ	/CAN NOT CLEAR BIT0 OF TCSR/
3861 020552	122	105	123	EM14:	.ASCIZ	/RESET DID NOT CLEAR BIT0 OF TCSR/
3862 020613	102	111	124	EM15:	.ASCIZ	/BIT2 OF TCSR NOT CLEAR AFTER RESET/
3863 020656	103	101	116	EM16:	.ASCIZ	/CAN NOT SET BIT2 OF TCSR/
3864 020707	103	101	116	EM17:	.ASCIZ	/CAN NOT CLEAR BIT2 OF TCSR/
3865 020742	122	105	123	EM20:	.ASCIZ	/RESET DID NOT CLEAR BIT2 OF TCSR/
3866 021003	102	111	124	EM21:	.ASCIZ	/BIT6 OF TCSR NOT CLEAR AFTER RESET/
3867 021046	130	115	111	EM22:	.ASCIZ	/XMIT INTERRUPT AT PRIORITY 7/
3868 021103	103	101	116	EM23:	.ASCIZ	/CAN NOT SET BIT6 OF TCSR/
3869 021134	103	101	116	EM24:	.ASCIZ	/CAN NOT CLEAR BIT6 OF TCSR/
3870 021167	122	105	123	EM25:	.ASCIZ	/RESET DID NOT CLEAR BIT6 OF TCSR/
3871 021230	102	111	124	EM26:	.ASCIZ	/BIT6 OF RCSR NOT CLEAR AFTER RESET/
3872 021273	122	103	126	EM27:	.ASCIZ	/RCVR INTERRUPT WITH PRIORITY 7/
3873 021332	103	101	116	EM30:	.ASCIZ	/CAN NOT SET BIT6 OF RCSR/
3874 021363	103	101	116	EM31:	.ASCIZ	/CAN NOT CLEAR BIT6 OF RCSR/
3875 021416	103	101	116	EM32:	.ASCIZ	/CAN NOT CLEAR BIT6 OF RCSR WITH RESET/
3876 021464	102	111	124	EM33:	.ASCIZ	/BIT6 OF LKS NOT CLEAR AFTER RESET/
3877 021526	114	113	123	EM34:	.ASCIZ	/LKS INTERRUPT WITH PRIORITY 7/
3878 021564	103	101	116	EM35:	.ASCIZ	/CAN NOT SET BIT6 OF LKS/
3879 021614	103	101	116	EM36:	.ASCIZ	/CAN NOT CLEAR BIT6 OF LKS/
3880 021646	122	105	123	EM37:	.ASCIZ	/RESET DID NOT CLEAR BIT6 OF LKS/
3881 021706	104	125	101	EM40:	.ASCIZ	/DUAL ADDRESSING ERROR/
3882 021734	102	111	124	EM41:	.ASCIZ	/BIT6 OF LKS NOT SET AFTER RESET/
3883 021774	103	101	116	EM42:	.ASCIZ	/CAN NOT CLEAR BIT7 OF LKS/
3884 022026	102	111	124	EM43:	.ASCIZ	/BIT7 OF LKS DOES NOT SET/
3885 022057	122	124	103	EM44:	.ASCIZ	/RTC INTERRUPT AT PRIORITY 7/
3886 022113	122	124	103	EM45:	.ASCIZ	/RTC INTERRUPTS WHEN DISABLED/
3887 022150				EM47:		
3888 022150	122	124	103	EM46:	.ASCIZ	/RTC INTERRUPT DID NOT OCCUR/
3889 022204	122	124	103	EM50:	.ASCIZ	/RTC DOUBLE INTERRUPT/
3890 022231	122	105	123	EM51:	.ASCIZ	/RESET DID NOT INTERRUPT/
3891 022261	122	124	103	EM52:	.ASCIZ	/RTC INTERRUPT DID NOT CLEAR WITH BIT7 OF LKS/
3892 022336	103	114	117	EM53:	.ASCIZ	/CLOCK REPEATABILITY/
3893 022362	130	115	111	EM54:	.ASCIZ	/XMIT INTERRUPTS WHEN DISABLED/
3894 022420	130	115	111	EM56:	.ASCIZ	/XMIT INTERRUPTS AT PRIORITY 7/
3895 022456	130	115	111	EM57:	.ASCIZ	/XMIT INTERRUPTS WITH ENABLE CLEAR/
3896 022520				EM55:		
3897 022520	130	115	111	EM60:	.ASCIZ	/XMIT DID NOT INTERRUPT/
3898 022547	130	115	111	EM61:	.ASCIZ	/XMIT RE-INTERRUPTED/
3899 022573	114	117	101	EM62:	.ASCIZ	/LOADING TBUF DID NOT CLEAR INTERRUPT/
3900 022640	122	103	126	EM63:	.ASCIZ	/RCVR ACTIVE NOT SET/
3901 022664	122	103	126	EM64:	.ASCIZ	/RCVR DONE NEVER SET/
3902 022710	122	103	126	EM65:	.ASCIZ	/RCVR ACTIVE NOT CLEARED WITH DONE SET/
3903 022756	122	105	123	EM66:	.ASCIZ	/RESET DID NOT CLEAR RCVR DONE/
3904 023014	122	104	122	EM67:	.ASCIZ	/RDR ENABLE DID NOT CLEAR RCVR DONE/

TERMINAL OUTPUT TEST

```

3905 023057      122      105      101 EM70:  .ASCIZ  /READING RBUF DID NOT CLEAR RCVR DONE/
3906 023124      122      103      126 EM71:  .ASCIZ  /RCVR INTERRUPTS WITH ENABLE CLEAR/
3907 023166      122      103      126 EM73:  .ASCIZ  /RCVR INTERRUPTS AT PRIORITY 7/
3908 023224      122      103      126 EM74:  .ASCIZ  /RCVR INT RQST PASSED WITH ENABLE CLEAR/
3909 023273
3910 023273      122      103      126 EM75:  .ASCIZ  /RCVR DID NOT INTERRUPT/
3911 023322      122      103      126 EM76:  .ASCIZ  /RCVR RE-INTERRUPTED/
3912 023346      122      105      101 EM77:  .ASCIZ  /READING RBUF DID NOT CLEAR INTERRUPT/
3913 023413      122      105      123 EM100: .ASCIZ  /RESET DID NOT CLEAR RCVR INTERRUPT/
3914 023456      047      117      122 EM101: .ASCIZ  /'OR' FLAG DID NOT SET/
3915 023504      047      105      122 EM102: .ASCIZ  /'ERROR' NOT SET WITH 'OR' FLAG/
3916 023543      102      122      105 EM103: .ASCIZ  /BREAK DID NOT XMIT ALL ZEROES/
3917 023601      102      122      105 EM104: .ASCIZ  /BREAK DID NOT SET 'FR' ERROR/
3918 023636      104      101      124 EM105: .ASCIZ  /DATA COMPARE ERROR/
3919 023661      114      117      117 EM106: .ASCIZ  /LOOP-BACK DATA COMPARE ERROR/
3920 023716      124      111      115 EM107: .ASCIZ  /TIMEOUT IN EXERCISER TEST/
3921 023750      111      116      103 EM110: .ASCIZ  /INCORRECT RECEIVE COUNT/
3922 024000      104      101      124 EM111: .ASCIZ  /DATA COMPARE ERROR IN EXERCISER/
3923 024040      124      122      101 EM112: .ASCIZ  /TRAP CATCHER/
3924 024055      116      117      040 EM113: .ASCIZ  /NO CLK INTERRUPT IN EXERCISER/
3925 024113      042      105      122 EM114: .ASCIZ  /"ERROR" NOT SET WITH "FR" FLAG/
3926 024152      122      103      126 EM115: .ASCIZ  /RCV ACTIVE NOT CLEAR WITH INIT/
3927 024211      122      103      126 EM116: .ASCIZ  /RCV ACTIVE WITHOUT "START" BIT/
3928 024250      122      104      122 EM117: .ASCIZ  /RDR ENABLE NOT CLEAR WITH RCV ACTIVE/
3929
3930 024315      124      105      123 DM1:   .ASCIZ  /TEST#  ERROR#  TCSR/
3931 024342      124      105      123 DM2:   .ASCIZ  /TEST#  ERR PC  TBUF/
3932 024367      124      105      123 DM6:   .ASCIZ  /TEST#  ERR PC  RCSR/
3933 024414      124      105      123 DM7:   .ASCIZ  /TEST#  ERR PC  RBUF/
3934 024441      124      105      123 DM10:  .ASCIZ  /TEST#  ERR PC  LKS/
3935 024465      124      105      123 DM40:  .ASCIZ  /TEST#  ERR PC  GOOD    BAD/
3936 024521      124      105      123 DM53:  .ASCIZ  /TEST#  ERR PC  LKS      CNT1   CNT2/
3937 024566      124      105      123 DM103: .ASCIZ  /TEST#  ERR PC  RCSR    DATA/
3938 024623      124      105      123 DM105: .ASCIZ  /TEST#  ERR PC  RCSR    GOOD    BAD/
3939 024667      124      105      123 DM110: .ASCIZ  /TEST#  ERR PC  RCSR    TRANS   RCV/
3940 024733      124      105      123 DM112: .ASCIZ  /TEST#  ERR PC  RCSR    OLDPC   TRAP ADR/
3941
3942 025004      015      012      103 M1:    .ASCIZ  <CR><LF>?CZDLIO DL11-W 11/44 MFM SLU?
3943
3944 025044      015      012
3945 025046      040      040      040 M2:    .ASCII  <CR><LF>
M2A:   .ASCIZ  /  DEVICES UNDER TEST  /
3946
3947
3948 025076      001072  001016  002640 DT1:   .WORD   #TESTN, #ERRPC, TCSR, 0
3949 025106      001072  001016  002642 DT2:   .WORD   #TESTN, #ERRPC, TBUF, 0
3950 025116      001072  001016  002634 DT6:   .WORD   #TESTN, #ERRPC, RCSR, 0
3951 025126      001072  001016  002636 DT7:   .WORD   #TESTN, #ERRPC, RBUF, 0
3952 025136      001072  001016  002674 DT10:  .WORD   #TESTN, #ERRPC, LKS, 0

```

TERMINAL OUTPUT TEST

```
3953 025146 001072 001016 001020 DT40: .WORD #TESTN,#ERRPC,#GDADR,#BDADR,0
3954 025160 001072 001016 002674 DT53: .WORD #TESTN,#ERRPC,LKS,FIRST,SECND,0
3955 025174 001072 001016 002634 DT103: .WORD #TESTN,#ERRPC,RCSR,#BDDAT,0
3956 025206 001072 001016 002634 DT105: .WORD #TESTN,#ERRPC,RCSR,#GDADR,#BDDAT,0
3957 025222 001072 001016 002634 DT110: .WORD #TESTN,#ERRPC,RCSR,XMTCNT,RCVCNT,0
3958 025236 001072 001016 002634 DT112: .WORD #TESTN,#ERRPC,RCSR,OLDPC,BDVECT,0
3959 025252 000000 ENDADR: .WORD 0 ;END ADDRESS FOR CHECKSUM AND SAVE AREA
3960 ;OF WRAP AROUND CONSOLE TEST
3961
3962 000001 .END
```


ABASE	=	176500	BGNADD	=	003014	DH103	024566	EM25	021167	FLAG44	003002
ACDW1	=	000000	BIT0	=	000001	DH105	024623	EM26	021230	GETB	014126
ACDW2	=	000000	BIT00	=	000001	DH110	024667	EM27	021273	GETLIN	014062
ACPUOP	=	000000	BIT01	=	000002	DH112	024733	EM3	020204	GOAGIN	014432
ADWO	=	000000	BIT02	=	000004	DH2	024342	EM30	021332	GTSWR	= 104405
ADDW1	=	000000	BIT03	=	000010	DH40	024465	EM31	021363	HT	= 000011
ADDW10	=	000000	BIT04	=	000020	DH53	024521	EM32	021416	ID	004620
ADDW11	=	000000	BIT05	=	000040	DH6	024367	EM33	021464	INIT	003462
ADDW12	=	000000	BIT06	=	000100	DH7	024414	EM34	021526	IOTVEC	= 000020
ADDW13	=	000000	BIT07	=	000200	DISPLA	001042	EM35	021564	JIMSTK	002432
ADDW14	=	0000C0	BIT08	=	000400	DISPRE	000174	EM36	021614	LF	= 000012
ADDW15	=	000000	BIT09	=	001000	DSMR	= 177570	EM37	021646	LKS	002674
ADDW2	=	000000	BIT1	=	000002	DT1	025076	EM4	020251	LOC1	002340
ADDW3	=	000000	BIT10	=	002000	DT10	025136	EM40	021706	LOC2	002342
ADDW4	=	000000	BIT11	=	004000	DT103	025174	EM41	021734	MANL	003502
ADDW5	=	000000	BIT12	=	010000	DT105	025206	EM42	021774	MFPT	= 000007
ADDW6	=	000000	BIT13	=	020000	DT110	025222	EM43	022026	MFR	017763
ADDW7	=	000000	BIT14	=	040000	DT112	025236	EM44	022057	MOR	017775
ADDW8	=	000000	BIT15	=	100000	DT2	025106	EM45	022113	MPAR	017752
ADDW9	=	000000	BIT2	=	000004	DT40	025146	EM46	022150	MSG	017724
ADEVCT	=	000000	BIT3	=	000010	DT53	025160	EM47	022150	MSTOP	020010
ADEVH	=	000000	BIT4	=	000020	DT6	025116	EM5	020273	M1	025004
ADR	004132		BIT5	=	000040	DT7	025126	EM50	022204	M2	025044
ADRTBL	002702		BIT6	=	000100	DVADT	003664	EM51	022231	M2A	025046
AENV	=	000000	BIT7	=	000200	ECHO	017600	EM52	022261	NDEOP	014316
AENVH	=	000000	BIT8	=	000400	EMTVEC	= 000030	EM53	022336	OLDPC	002620
AFATAL	=	000000	BIT9	=	001000	EM1	020134	EM54	022362	OLDSUM	002436
AMADR1	=	000000	BPT	=	000003	EM10	020400	EM55	022520	OUTTST	020020
AMADR2	=	000000	BPTVEC	=	000014	EM100	023413	EM56	022420	PFECDF	015206
AMADR3	=	000000	BUF	002454		EM101	023456	EM57	022456	PFECDH	015146
AMADR4	=	000000	BUFF	020132		EM102	023504	EM6	020330	PFECDT	015176
AMMS1	=	000000	CATCH	014520		EM103	023543	EM60	022520	PFECM	015106
AMMS2	=	000000	CHKSUM	014010		EM104	023601	EM61	022547	PFECWS	015076
AMMS3	=	000000	CKSMR	= 104406		EM105	023636	EM62	022573	PIRQ	= 177772
AMMS4	=	000000	CLK	013630		EM106	023661	EM63	022640	PIRQVE	= 000240
AMSGAD	=	000000	CLKCNT	002444		EM107	023716	EM64	022664	PROMPT	002566
AMSGLG	=	000000	CMPCRE	007330		EM11	020423	EM65	022710	PRO	= 000000
AMSGTY	=	000000	CNTLP	002564		EM110	023750	EM66	022756	PR1	= 000040
AMTYP1	=	000000	COMP	013432		EM111	024000	EM67	023014	PR2	= 000100
AMTYP2	=	000000	CONT	006246		EM112	024040	EM7	020354	PR3	= 000140
AMTYP3	=	000000	CONT41	012354		EM113	024055	EM70	023057	PR4	= 000200
AMTYP4	=	000000	CPSAVE	015674		EM114	024113	EM71	023124	PR5	= 000240
APASS	=	000000	CR	= 000015		EM115	024152	EM72	023273	PR6	= 000300
APRIOR	=	000000	CRBUF	002656		EM116	024211	EM73	023166	PR7	= 000340
APTCSU	=	000040	CRCSR	002654		EM117	024250	EM74	023224	PS	= 177776
APTENV	=	000001	CRLF	= 000200		EM12	020466	EM75	023273	PSW	= 177776
APTSIZ	=	000200	CRPSW	002666		EM13	020517	EM76	023322	PUTLIN	014030
APTSPO	=	000100	CRVECT	002664		EM14	020552	EM77	023346	PWRVEC	= 000024
APTS7D	003636		CTBUF	002662		EM15	020613	ENDADD	= 025252	RBUF	002636
ASWREG	=	000000	CTCSR	002660		EM16	020656	ENDADR	025252	RBUF58	= 176502
ATESTN	=	000000	CTPSW	002672		EM17	020707	ENDB7	004676	RCSR	002634
AUNIT	=	000000	CTSTFL	002624		EM2	020160	ENDEV	014272	RCSR58	= 176500
AUSMR	=	000400	CTVECT	002670		EM20	020742	ENDMG	014416	RCV	013614
AVECT1	=	000300	DDISP	= 177570		EM21	021003	ENDSTK	002412	RCVCNT	002440
AVECT2	=	000000	DEVADR	003014		EM22	021046	ERROR	= 104000	RCVDON	010346
BDVECT	002622		DH1	024315		EM23	021103	ERRVEC	= 000004	RDCHR	= 104407
BEGIN	003774		DH10	024441		EM24	021134	FIRST	002336	RDLIN	= 104410

SYMBOL TABLE

RESTTE	014236	TBITVE	000014	TST46	013636	\$ERFLG	001003	\$PASS	001074
RESVEC	000010	TBUF	002642	TST5	004714	\$ERMAX	001015	\$PASTH	000506
RPSW	002646	TBUF58	176506	TST6	004760	\$ERROR	014544	\$POWER	015364
RSTR	014472	TLOCK	005536	TST7	005012	\$ERRPC	001016	\$PWRDN	015212
RTCPSW	002700	TCONS	004020	TURBUF	003006	\$ERRTB	001146	\$PWRMG	015346
RTCVT	002676	TCSR	002640	TURCSR	003004	\$ERRTY	014726	\$PWRUP	015264
RVECT	002644	TCSRS8	176504	TUTBUF	003012	\$ERTTL	001012	\$QUES	001062
R6	0000006	TIMER	014162	TUTCSR	003010	\$ESCAP	001060	\$RDCHR	017220
R7	0000007	TKVEC	000060	TVECT	002650	\$ETABL	001106	\$RDLIN	017350
SAVEPS	002434	TMP1	002626	TYPE	104401	\$ETEND	001146	\$RDSZ	000010
SAVETE	014202	TMP2	002630	TYPOC	104402	\$FATAL	001070	\$RTNAD	014410
SAVEO	002344	TMP3	002632	TYPON	104404	\$FFLG	016142	\$SAVR6	015362
SAVLOC	002346	TOLER	007342	TYPOS	104403	\$FILLC	001056	\$SCOPE	015374
SCOPE	000004	TPSW	002652	VCTADR	003670	\$FILS	001055	\$SETUP	000137
SCPSW	002452	TPVEC	000064	VCTTBL	002742	\$GDADR	001020	\$STUP	177777
SECND	002616	TRAPVE	000034	VECT	004152	\$GDDAT	001024	\$SVLAD	015624
SETADR	004100	TRTVEC	000014	WACTV	010230	\$GET42	014366	\$SVPC	000500
SHIFT	003740	TSTDEV	004050	WAIT	017732	\$GTSWR	017006	\$SWR	161000
SIZE	003512	TSTDVM	003646	WDone	010362	\$HIBTS	000500	\$SWREG	001110
SRPSW	002450	TST1	004172	WRAP	013640	\$ICNT	001004	\$SWRMK	000000
STACK	001100	TST10	005044	WRPSW	014502	\$ILLUP	015356	\$TESTN	001072
START	003046	TST11	005252	WT	010316	\$INTAG	001035	\$TKB	001046
STKLMT	177774	TST12	005404	XCONT	013604	\$ITEMB	001014	\$TKS	001044
STPSW	002446	TST13	005544	XMIT	013542	\$LF	001064	\$TN	000047
SMR	001040	TST14	005614	XMTCNT	002442	\$LFLG	016141	\$TPB	001052
SMREG	000176	TST15	005764	XRET	013610	\$LPADR	001006	\$TPFLG	001057
SMO	000001	TST16	006154	\$APTHD	000500	\$LPERR	001010	\$TPS	001050
SM00	000001	TST17	006262	\$ATYC	015722	\$MADR1	001120	\$TRAP	017522
SM01	000002	TST2	004244	\$ATY1	015676	\$MADR2	001124	\$TRAP2	017544
SM02	000004	TST20	006522	\$ATY3	015704	\$MADR3	001130	\$TRP	000011
SM03	000010	TST21	006720	\$ATY4	015714	\$MADR4	001134	\$TRPAD	017556
SM04	000020	TST22	007054	\$AUTOB	001034	\$MAIL	001066	\$TSTM	000504
SM05	000040	TST23	007222	\$BASE	001142	\$MMS1	001116	\$TSTNM	001002
SM06	000100	TST24	007410	\$BDADR	001022	\$MMS2	001122	\$TTYIN	017456
SM07	000200	TST25	007520	\$BDAT	001026	\$MMS3	001126	\$TYPE	016144
SM08	000400	TST26	007646	\$CHARC	016504	\$MMS4	001132	\$TYPEC	016356
SM09	001000	TST27	010012	\$CKSMR	016736	\$MBADR	000502	\$TYPEX	016506
SM1	000002	TST3	004316	\$CNTAG	001000	\$MFLG	016140	\$TYPOC	016534
SM10	002000	TST30	010156	\$CMS	000000	\$MNEW	017511	\$TYPON	016550
SM11	004000	TST31	010462	\$CNTLG	017473	\$MSGAD	001102	\$TYPOS	016510
SM12	010000	TST32	010554	\$CNTLU	017466	\$MSGLG	001104	\$UNIT	001100
SM13	020000	TST33	010700	\$CPUOP	001114	\$MSGTY	001066	\$UNITM	000510
SM14	040000	TST34	011102	\$CRLF	001063	\$MSMR	017500	\$USMR	001112
SM15	100000	TST35	011304	\$DEVCT	001076	\$MTYP1	001117	\$VECT1	001136
SM2	000004	TST36	011512	\$DEVM	001144	\$MTYP2	001123	\$VECT2	001140
SM3	000010	TST37	011656	\$DOAGN	014406	\$MTYP3	001127	\$XOFF	000023
SM4	000020	TST4	004466	\$ENDAD	014376	\$MTYP4	001133	\$XON	000021
SM5	000040	TST40	012014	\$ENDCT	014356	\$NULL	001054	\$XTSTR	015406
SM6	000100	TST41	012200	\$ENULL	014412	\$NMTST	000001	\$GET4	000000
SM7	000200	TST42	012414	\$ENV	001106	\$OCNT	016732	\$OFILL	016733
SM8	000400	TST43	012576	\$ENVH	001107	\$OMODE	016734	\$ERRT	001146
SM9	001000	TST44	012744	\$EOP	014326	\$OVER	015660	\$X	000500
TA	002610	TST45	013062	\$EOPCT	014350				

. ABS. 025254 000
 000000 001
 ERRORS DETECTED: 0

VIRTUAL MEMORY USED: 45360 WORDS (178 PAGES)
DYNAMIC MEMORY: 20060 WORDS (77 PAGES)
ELAPSED TIME: 00:02:34
CZDLIO.BIN.CZDLIO.SEQ/-SP=CZDLIO.MLB/ML.CZDLIO.P11