

KD11-Z,  
DL11-W

DL11-W/1144 MFM SLU  
CZDL DHO

AH-8529H-MC  
FICHE 1 OF 1

OCT 1981  
COPYRIGHT © 75-81  
MADE IN USA



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

.REM

IDENTIFICATION  
-----

PRODUCT CODE:    AC-852CH-MC  
PRODUCT NAME:    CZDL DHO 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 BY DIGITAL EQUIPMENT CORPORATION

THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION:

DIGITAL	PDP	UNIBUS	MASSBUS
DEC	DECUS	DECTAPE	

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  
85  
86  
87  
88  
89  
90  
91  
92  
93  
94  
95HISTORY SECTION  
-----

CZDLDDO 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 ^P 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 A\* ATTACHED TU58 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 TU58 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.
4. IT WAS FOUND THAT UPON POWERUP AND WITH THE TU58 ATTACHED, CHARACTERS WOULD BE SENT TO THE RECEIVER FROM THE TU58. 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 TU58 TO CONSOLE

96  
97

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

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 ^Q'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.

140  
141  
142  
143  
144  
145  
146  
147  
148  
149  
150  
151  
152  
153  
154  
155  
156  
157  
158  
159  
160  
161  
162  
163  
164  
165  
166  
167  
168  
169  
170  
171  
172  
173  
174  
175  
176  
177  
178  
179  
180  
181  
182  
183  
184  
185  
186  
187  
188  
189  
190  
191  
192  
193  
194  
195  
1961.0 GENERAL INFORMATION1.1 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 TU58 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 REQUIREMENTS1.2.1 EQUIPMENT

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

197  
198  
199  
200  
201  
202  
203  
204  
205  
206  
207

- 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

209 1.3 ASSUMPTIONS  
210 -----  
211 A. IF THE UNIT UNDER TEST (UUT) IS THE CONSOLE, THE PROGRAM WILL  
212 ASSUME THE REAL TIME CLOCK (RTC) IS ENABLED AND WILL TEST IT  
213 UNLESS THE TESTS ARE DISABLED BY BIT6 OF THE SWR.  
214  
215 B. IF THE UUT IS NOT THE CONSOLE, THE RTC IS NOT TESTED FOR THAT DEVICE.  
216  
217 C.  
218 FOR THE DL11-W:  
219  
220 THE PROGRAM WILL ASSUME THE ERROR FLAG BITS AND THE BREAK FUNCTION  
221 OF THE DL11-W ARE DISABLED AND WILL NOT TEST THESE FUNCTIONS  
222 UNLESS ENABLED BY BIT10 (FOR ERROR FLAGS) AND BIT8 (FOR BREAK)  
223 OF THE SWR. THE DEFAULT CHARACTER SIZE IS 8 BITS (SEE PARA 2.3.2).  
224  
225 FOR THE 11/44:  
226  
227 THE PROGRAM ASSUMES THAT THE ERROR FLAG BITS AND THE  
228 BREAK FUNCTION ARE DISABLED, AND WILL NOT TEST THESE FUNCTIONS  
229 HOWEVER, IF BIT 10 (FOR ERROR FLAGS) AND BIT 08 (FOR BREAK)  
230 OF SWR ARE SET, THEN ERROR FLAGS AND BREAK TESTS ARE PERFORMED  
231 FOR THE TU58 SLU ONLY. IF BIT03 OF SWR IS ALSO  
232 SET THEN THESE TESTS WILL BE ENABLED FOR THE CONSOLE.  
233  
234 READER ENABLE AND RECEIVER ACTIVE TESTS  
235 ARE NOT PERFORMED SINCE THE 11/44 MFM DOES NOT IMPLEMENT THESE  
236 FUNCTIONS.  
237  
238  
239 2.0 OPERATING INSTRUCTIONS  
240 -----  
241  
242 2.1 LOADING PROCEDURE  
243 -----  
244 USE STANDARD PROCEDURE FOR PDP-11 ABSOLUTE BINARY FORMATTED TAPES.  
245  
246  
247 2.2 STARTING PROCEDURE  
248 -----  
249 LOAD THE SWITCH REGISTER WITH SETTING  
250  
251  
252 NOTE: IF USING A CPU WITHOUT HARDWARE SWITCH REGISTER  
253 SOFTWARE SWITCH REGISTER LOCATION = 176.  
254 (FOR A 11/44 CPU USE MFM CONSOLE FOR DEPOSITING  
255 SWITCH REGISTER. TYPE ^P TO ENTER CONSOLE)  
256  
257 A. START AT 200.  
258 AFTER CHECKING THE TRANSMITTER, THE PROGRAM WILL PRINT ITS IDENTIFICATION  
259 AND REPORT THE NUMBER OF DEVICES UNDER TEST (NUMBER IS OCTAL).  
260 'END PASS' IS PRINTED AFTER A FULL PASS HAS BEEN MADE ON ALL DEVICES  
261 UNDER TEST.  
262  
263 B. START AT 204. \*\*\*\*\*NOTE\*\*\*\*\*  
264 THE 'ECHO' TEST WILL BE EXECUTED. AN '\*' IS PRINTED AT THE  
265 BEGINNING OF THE TEST. THE ECHO TEST READS A CHARACTER FROM



266  
267  
268  
269  
270  
271  
272  
273  
274  
275  
276  
277  
278  
279  
280  
281  
282  
283  
284  
285  
286  
287  
288  
289  
290  
291  
292

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)
@ABCDEFGHIJKLMN O PQRSTU VWXYZ[\]^_	(040 --> 077)
'ABCDEFGHIJKLMN O PQRSTU VWXYZ	(100 --> 137)
	(140 --> 177) [LOWER CASE ALPHA]

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

@ABCDEFGHIJKLMN O PQRSTU VWXYZ[\]                    [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.

:::++  
:::++

294  
295  
296  
297  
298  
299  
300  
301  
302  
303  
304  
305  
306  
307  
308  
309  
310  
311  
312  
313  
314  
315  
316  
317  
318  
319  
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

## 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 TYPEOUT
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 '\$DEVN' (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:

351  
352  
353  
354  
355  
356  
357  
358  
359  
360  
361  
362  
363  
364  
365  
366  
367  
368  
369  
370  
371  
372  
373  
374

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 '\$USWR' AS FOLLOWS:

CHAR. SIZE (# OF BITS)	'\$USWR' CONTENTS	
	(BINARY)	(OCTAL)
8	10000000	400
7	01000000	200
6	00100000	100
5	00010000	40

'\$USWR' 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 (\$STSM, \$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 \$DEVM=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 (\$ENVM=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

375  
376  
377  
378  
379  
380  
381  
382  
383  
384  
385  
386  
387  
388  
389  
390  
391  
392  
393  
394  
395  
396  
397  
398  
399  
400  
401  
402  
403  
404  
405  
406  
407  
408  
409  
410  
411  
412  
413  
414  
415  
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

	.....	E TABLES	.....
		A	B
	E-MODE/S-MODE (\$ENVM/\$ENV)	200/000	240/001
	SWITCH REGISTER 1 (\$SWREG)	002640	002404
	SWITCH REGISTER 2	000400	000400
	CPU TYPE/OPTIONS	00/0000	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 (\$DEVM)	000003	000003
	CTLR. SPECIFIC WORD 1	000000	000000
	CTLR. SPECIFIC WORD 2	000000	000000

457  
458  
459  
460  
461  
462  
463  
464  
465  
466  
467  
468  
469  
470  
471  
472  
473  
474  
475  
476  
477  
478  
479  
480  
481  
482  
483  
484  
485  
486  
487  
488  
489  
490  
491  
492  
493  
494  
495  
496  
497  
498  
499  
500  
501  
502  
503  
504  
505  
506  
507

#### 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  
CRVECT: ADDRESS OF RECEIVER VECTOR  
CRPSW: ADDRESS OF ASSOCIATED PSW  
CTVECT: 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)
		⋮
		↓
UNIT#15	-->	BASE ADDRESS + (160)
		BASE VECTOR + (160)

508  
509  
510  
511  
512  
513  
514  
515  
516  
517  
518  
519  
520  
521  
522  
523  
524  
525  
526  
527  
528  
529  
530  
531  
532  
533  
534

STARTING AT LOCATION 200 AND HAVING BIT5 OF SWR CLEAR, THE PROGRAM WILL SELF  
SIZE THE NUMBER OF DEVICES (STARTING AT THE BAS ADDRESS) AND  
STORE A BIT MAP AT '\$DEV M' (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 '\$DEV M' PRIOR TO STARTING THE PROGRAM  
SETTING BIT5 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

$DEV M: 13                    [BINARY - 0 000 000 000 001 011]

```

```

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

```

535  
536  
537  
538  
539  
540  
541  
542  
543  
544  
545  
546

2.4    EXECUTION TIMES -

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

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



547  
548  
549  
550  
551  
552  
553  
554  
555  
556  
557  
558  
559  
560  
561  
562  
563  
564  
565  
566  
567  
568  
569  
570  
571  
572  
573  
574  
575  
576  
577  
578  
579  
580  
581  
582  
583  
584

### 3.0 ERROR REPORTING

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

```
“(SOME ASCII MESSAGE)”  
TEST#  ERR PC  RCSR  [ANY APPLICABLE DAT 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 TYPEOUT. 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 TYPEOUT.

IF THE CPU THIS DIAGNOSTIC WILL BE RUN ON DOES NOT HAVE A CPU ERROR REGISTER (ADDRESS 17777766), 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.

585  
586  
587  
588  
589  
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

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.

623  
624  
625  
626  
627  
628  
629  
630  
631  
632  
633  
634  
635  
636  
637  
638  
639  
640  
641  
642  
643  
644  
645  
646  
647  
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

#### 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 '\$APTYPE' 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 CKSWR

-----  
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 TU58 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 SWR =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

680  
681  
682  
683  
684  
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

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

```
CZDLHD0 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 ....
```

DESCRIPTION: REFERRING TO THE ABOVE PRINTOUTS:

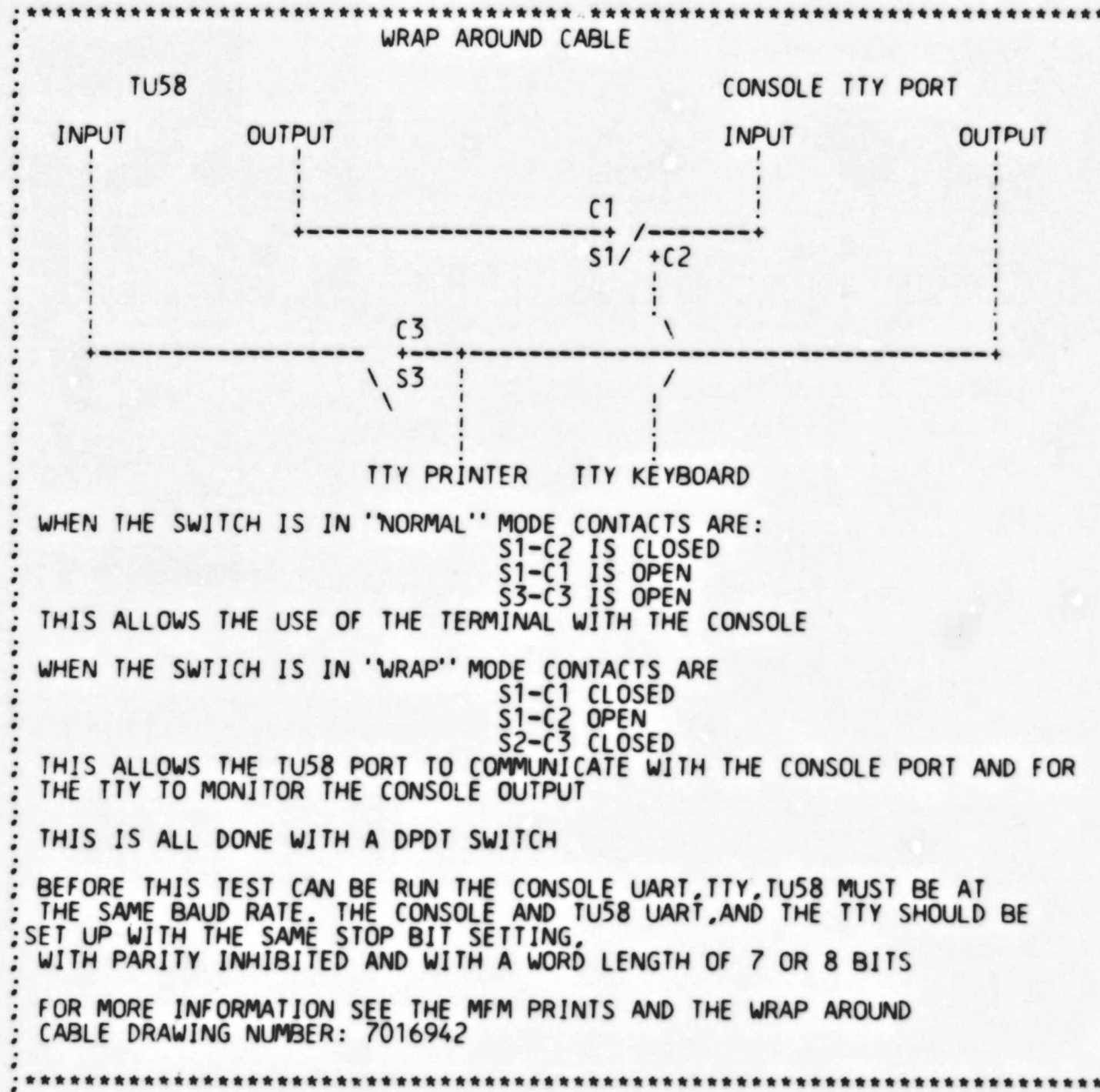
- A. THE DIAGNOSTIC IS LOADED WITH THE TITLE BEING PRINTED.
- B. TWO SLU DEVICES(CONSOLE,TU58) ARE SPECIFIED
- C. BOTH SLUS ARE TESTED COMPLETELY. AT THIS POINT THE THE DIAGNOSTIC ISSUES ^P TO THE WRAP CABLE. THE CONSOLE ENTERS CONSOLE MODE AND THE ^P TYPED ON THE TERMINAL IS THE MFM CONSOLE ECHO.
- 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.

723  
724  
725  
726  
727  
728  
729  
730  
731  
732  
733  
734  
735  
736  
737  
738  
739  
740  
741

2. IF 'E TABLE B' OF 2.3.3 WERE USED WITH TYPEOUTS SUPPRESSED (\$ENV=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^P .....ETC.
```

FIG. 1- WRAPAROUND CONFIGURATION - AUTO INITIATION OF 11/44 T/A CONSOLE TEST



743  
 744  
 745  
 746  
 747  
 748  
 749  
 750  
 751  
 752  
 753  
 754  
 755  
 756  
 757  
 758  
 759  
 760  
 761  
 762  
 763  
 764  
 765  
 766  
 767  
 768  
 769  
 770  
 771  
 772  
 773  
 774  
 775  
 776  
 777  
 778  
 779  
 780  
 781  
 782  
 783  
 784  
 785  
 786  
 787  
 788  
 789  
 790  
 791  
 792  
 793  
 794

803  
804  
805  
806  
807  
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  
1251  
1252  
1253  
1254  
1255  
1256  
1257  
1258  
1259  
1260  
1261  
1262  
1263  
1264  
1265  
1266

```

:*****
:*****
:***** . $ERROR *****
:
: $ERROR IS THE ERROR HANDLER
:
: ARGUMENTS:
:
: 1) ADDRESS -- 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 CONJUNCTION 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

```

1267  
1268  
1269  
1270  
1271  
1272  
1273  
1274  
1275  
1276  
1277  
1278  
1279  
1280  
1281  
1282  
1283  
1284  
1285  
1286  
1287  
1288  
1377  
1381  
1385  
1389  
1390  
1397  
1398  
1404  
1414  
1426  
1427  
1428  
1429  
1430  
1431  
1432  
1433

WITH '0' FOR OCTAL AND '1' FOR DECIMAL.

: ROUTINES REQUIRED:

- : 1) . \$TYPOCT TYPE AN OCTAL NUMBER
- : 2) . \$STYPDEC 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

```

: \*\*\*\*\*

```

.MCALL NEWST,$$NEWTEST,.$TYPE,.$TYPOC,.$STRAP
.MCALL .SETUP,STARS,PUSH,POP,SETUP,.EQUIV
.MCALL . $APTHDR,.$APTBL,.$ACT11
.MCALL . $CMTAG,.$EOP,.$READ
.MCALL .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

```

001100  
104000  
000004  
  
000011  
000012  
000015  
000200  
177776  
177776  
177774  
177772  
177570  
177570  
  
000000  
000001



```
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
SW9=SW09
SW8=SW08
SW7=SW07
SW6=SW06
SW5=SW05
SW4=SW04
SW3=SW03
SW2=SW02
SW1=SW01
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
```

```
00001C 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
ERRVEC= 4 ;:TIME OUT AND OTHER ERRORS
RESVEC= 10 ;:RESERVED AND ILLEGAL INSTRUCTIONS
TBITVEC=14 ;: 'T' BIT
TRTVEC= 14 ;:TRACE TRAP
BPTVEC= 14 ;:BREAKPOINT TRAP (BPT)
IOTVEC= 20 ;:INPUT/OUTPUT TRAP (IOT) **SCOPE**
PWRVEC= 24 ;:POWER FAIL
EMTVEC= 30 ;:EMULATOR TRAP (EMT) **ERROR**
TRAPVEC=34 ;: 'TRAP' TRAP
TKVEC= 60 ;:TTY KEYBOARD VECTOR
IPVEC= 64 ;:TTY PRINTER VECTOR
PIRQVEC=240 ;:PROGRAM INTERRUPT REQUEST VECTOR

MFPT=7
ABASE= 176500
AVECT1= 300
AUSWR= 400
$TN= 1
$SWR= 161000
BPT= 000003 ;:THIS IS THE COMMAND FOR A TRAP
; THROUGH 14 (BPT TRAP)

.=0
;:*****
;*ALL UNUSED LOCATIONS FROM 4-776 CONTAIN A ".+2,BPT"
;*SEQUENCE TO CATCH ILLEGAL TRAPS & INTERRUPTS
;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS

.=14 ;:THE BPT TRAP VECTOR POINTS TO THE
.WORD CATCH ; ILLEGAL TRAP HANDLER "CATCH"
.WORD 340

.= 42
.WORD 0

.= 174
DISPREG:.WORD 0
SWREG: .WORD 0
```

BASIC DEFINITIONS

1472		000200	
1473	000200	000137	003046
1474	000204	000137	017600
1475	000210	000137	020020

. = 200

JMP	START
JMP	ECHO
JMP	OUTTST

;DO	INTERFACE TEST
;DO	ECHO TEST
;DO	OUTPUT TEST TO TERMINAL

BASIC DEFINITIONS

1477 000500  
1478

. = 500  
.SBTTL ACT11 HOOKS  
:\*\*\*\*\*  
:HOOKS REQUIRED BY ACT11

000046 000500  
014376 000046  
000052 014376  
000000 000052  
000500

\$SVPC =  
. = 46  
\$ENDAD  
. = 52  
.WORD 0  
.= \$SVPC

;SAVE PC  
::1)SET LOC.46 TO ADDRESS OF \$ENDAD IN .SEOP  
::2)SET LOC.52 TO ZERO  
::RESTORE PC

1480

000500  
000024 000024  
000024 000200  
000044 000044  
000044 000500  
000500  
000500 000000  
000502 001066  
000504 000050  
000506 000060  
000510 000055  
000512 000030

```
.SBTTL APT PARAMETER BLOCK
:*****
:SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
:*****
      .SX=      ;;SAVE CURRENT LOCATION
      =24      ;;SET POWER FAIL TO POINT TO START OF PROGRAM
      200      ;;FOR APT START UP
      =44      ;;POINT TO APT INDIRECT ADDRESS PNTR.
$APTHDR ;;POINT TO APT HEADER BLOCK
      =.SX     ;;RESET LOCATION COUNTER
:*****
:SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
:INTERFACE SPEC.
$APTHD:
$HIBTS: .WORD 0      ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
$MBADR: .WORD $MAIL  ;;ADDRESS OF APT MAILBOX (BITS 0-15)
$TSTM:  .WORD 50     ;;RUN TIM OF LONGEST TEST
$PASTM: .WORD 60     ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
$UNITM: .WORD 55     ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
      .WORD $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE (WORDS)
```

1482

001000- 001000  
001000 000000  
001002 000  
001003 000  
001004 000000  
001006 000000  
001010 000000  
001012 000000  
001014 000  
001015 001  
001016 000000  
001020 000000  
001022 000000  
001024 000000  
001026 000000  
001030 000000  
001032 000000  
001034 000  
001035 000  
001036 000000  
001040 177570  
001042 177570  
001044 177560  
001046 177562  
001050 177564  
001052 177566  
001054 000  
001055 002  
001056 012  
001057 000  
001060 000000  
001062 077  
001063 015  
001064 012 000

```
.SBTTL COMMON TAGS
:*****
:*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
:*USED IN THE PROGRAM.
.=1000
$CMTAG:                ;;START OF COMMON TAGS
      .WORD            0
$TSTNM: .BYTE          0      ;;CONTAINS THE TEST NUMBER
$ERFLG: .BYTE          0      ;;CONTAINS ERROR FLAG
$I CNT:  .WORD         0      ;;CONTAINS SUBTEST ITERATION COUNT
$LPADR: .WORD          0      ;;CONTAINS SCOPE LOOP ADDRESS
$LPERR: .WORD          0      ;;CONTAINS SCOPE RETURN FOR ERRORS
$ERTTL: .WORD          0      ;;CONTAINS TOTAL ERRORS DETECTED
$ITEMB: .BYTE          0      ;;CONTAINS ITEM CONTROL BYTE
$ERMAX: .BYTE          1      ;;CONTAINS MAX. ERRORS PER TEST
$ERRPC: .WORD          0      ;;CONTAINS PC OF LAST ERROR INSTRUCTION
$GDADR: .WORD          0      ;;CONTAINS ADDRESS OF 'GOOD' DATA
$BDADR: .WORD          0      ;;CONTAINS ADDRESS OF 'BAD' DATA
$GDDAT: .WORD          0      ;;CONTAINS 'GOOD' DATA
$BDDAT: .WORD          0      ;;CONTAINS 'BAD' DATA
      .WORD            0      ;;RESERVED--NOT TO BE USED
      .WORD            0
$AUTOB: .BYTE          0      ;;AUTOMATIC MODE INDICATOR
$INTAG: .BYTE          0      ;;INTERRUPT MODE INDICATOR
      .WORD            0
SWR:      .WORD        DSWR    ;;ADDRESS OF SWITCH REGISTER
DISPLAY:  .WORD        DDISP   ;;ADDRESS OF DISPLAY REGISTER
$TKS:    177560           ;;TTY KBD STATUS
$TKB:    177562           ;;TTY KBD BUFFER
$TPS:    177564           ;;TTY PRINTER STATUS REG. ADDRESS
$TPB:    177566           ;;TTY PRINTER BUFFER REG. ADDRESS
$NULL:   .BYTE          0      ;;CONTAINS NULL CHARACTER FOR FILLS
$FILLS:  .BYTE          2      ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
$FILLC:  .BYTE         12      ;;INSERT FILL CHARS. AFTER A 'LINE FEED'
$TPFLG:  .BYTE          0      ;;'TERMINAL AVAILABLE' FLAG (BIT<07>=0=YES)
$ESCAPE: 0               ;;ESCAPE ON ERROR ADDRESS
$QUES:   .ASCII        /?/    ;;QUESTION MARK
$CRLF:   .ASCII        <15>   ;;CARRIAGE RETURN
$LF:     .ASCII        <12>   ;;LINE FEED
:*****
```

```
.SBTTL APT MAILBOX-ETABLE
:*****
.EVEN
$MAIL:                ;;APT MAILBOX
$MSGTY: .WORD        AMSGTY   ;;MESSAGE TYPE CODE
$FATAL: .WORD        AFATAL   ;;FATAL ERROR NUMBER
$TESTN: .WORD        ATESTN   ;;TEST NUMBER
$PASS:  .WORD        APASS    ;;PASS COUNT
$DEVCT: .WORD        ADEVCT   ;;DEVICE COUNT
$UNIT:  .WORD        AUNIT    ;;I/O UNIT NUMBER
$MSGAD: .WORD        AMSGAD   ;;MESSAGE ADDRESS
$MSGLG: .WORD        AMSGLG   ;;MESSAGE LENGTH
$ETABLE:                ;;APT ENVIRONMENT TABLE
$ENV:   .BYTE        AENV     ;;ENVIRONMENT BYTE
$ENVM:  .BYTE        AENVM    ;;ENVIRONMENT MODE BITS
$SWREG: .WORD        ASWREG   ;;APT SWITCH REGISTER
$USWR:  .WORD        AUSWR    ;;USER SWITCHES
```

```

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 $MAMS1: .BYTE AMAMS1 ;;HIGH ADDRESS,M.S. BYTE
001117 000 $MTYP1: .BYTE AMTYP1 ;;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 $MAMS2: .BYTE AMAMS2 ;;HIGH ADDRESS,M.S. BYTE
001123 000 $MTYP2: .BYTE AMTYP2 ;;MEM. TYPE,BLK#2
001124 000000 $MADR2: .WORD AMADR2 ;;MEM.LAST ADDRESS,BLK#2
001126 000 $MAMS3: .BYTE AMAMS3 ;;HIGH ADDRESS,M.S.BYTE
001127 000 $MTYP3: .BYTE AMTYP3 ;;MEM. TYPE,BLK#3
001130 000000 $MADR3: .WORD AMADR3 ;;MEM.LAST ADDRESS,BLK#3
001132 000 $MAMS4: .BYTE AMAMS4 ;;HIGH ADDRESS,M.S.BYTE
001133 000 $MTYP4: .BYTE AMTYP4 ;;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 $DEV M: .WORD ADEV M ;;DEVICE MAP
001146 .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
:*      DH      ;;POINTS TO THE DATA HEADER
:*      DT      ;;POINTS TO THE DATA
:*      DF      ;;POINTS TO THE DATA FORMAT
```

001146				
1483	001146			
1484	001146	020104	EM1	;'CAN NOT ACCESS TCSR'
1485	001150	024265	DH1	;'TEST# ERR PC TCSR'
1486	001152	025046	DT1	;\$TESTN,\$ERRPC,TCSR
1487	001154	000000	0	
1488				
1489	001156	020130	EM2	;'CAN NOT ACCESS TBUF'
1490	001160	024312	DH2	;'TEST# ERR PC TBUF'
1491	001162	025056	DT2	;\$TESTN,\$ERRPC,TBUF
1492	001164	000000	0	
1493				
1494	001166	020154	EM3	;'TCSR DONE NOT CLEARED WITH TBUF FULL'
1495	001170	024265	DH1	;'TEST# ERR PC TCSR'
1496	001172	025046	DT1	;\$TESTN,\$ERRPC,TCSR
1497	001174	000000	0	
1498				
1499	001176	020221	EM4	;'TCSR DONE NOT SET'
1500	001200	024265	DH1	;'TEST# ERR PC TCSR'
1501	001202	025046	DT1	;\$TESTN,\$ERRPC,TCSR
1502	001204	000000	0	
1503				
1504	001206	020243	EM5	;'TCSR DONE NOT SET WITH RESET'
1505	001210	024265	DH1	;'TEST# ERR PC TCSR'
1506	001212	025046	DT1	;\$TESTN,\$ERRPC,TCSR
1507	001214	000000	0	
1508				
1509	001216	020300	EM6	;'CAN NOT ACCESS RCSR'
1510	001220	024337	DH6	;'TEST# ERR PC RCSR'
1511	001222	025066	DT6	;\$TESTN,\$ERRPC,RCSR
1512	001224	000000	0	



1513	001226	020324	EM7	;'CAN NOT ACCESS RBUF''
1514	001230	024364	DH7	;'TEST# ERR PC RBUF''
1515	001232	025076	DT7	;\$TESTN,\$ERRPC,RBUF
1516	001234	000000	0	
1517				
1518	001236	020350	EM10	;'CAN NOT ACCESS LKS''
1519	001240	024411	DH10	;'TEST# ERR PC LKS''
1520	001242	025106	DT10	;\$TESTN,\$ERRPC,LKS
1521	001244	000000	0	
1522				
1523	001246	020373	EM11	;'BIT0 OF TCSR NOT CLEAR AFTER RESET''
1524	001250	024265	DH1	;'TEST# ERR PC TCSR''
1525	001252	025046	DT1	;\$TESTN,\$ERRPC,TCSR
1526	001254	000000	0	
1527				
1528	001256	020436	EM12	;'CAN NOT SET BIT0 OF TCSR''
1529	00126J	024265	DH1	;'TEST# ERR PC TCSR''
1530	001262	025046	DT1	;\$TESTN,\$ERRPC,TCSR
1531	001264	000000	0	
1532				
1533	001266	020467	EM13	;'CAN NOT CLEAR BIT0 OF TCSR''
1534	001270	024265	DH1	;'TEST# ERR PC TCSR''
1535	001272	025046	DT1	;\$TESTN,\$ERRPC,TCSR
1536	001274	000000	0	
1537				
1538	001276	020522	EM14	;'RESET DID NOT CLEAR BIT0 OF TCSR''
1539	001300	024265	DH1	;'TEST# ERR PC TCSR''
1540	001302	025046	DT1	;\$TESTN,\$ERRPC,TCSR
1541	001304	000000	0	
1542				
1543	001306	020563	EM15	;'BIT2 OF TCSR NOT CLEAR AFTER RESET''
1544	001310	024265	DH1	;'TEST# ERR PC TCSR''
1545	001312	025046	DT1	;\$TESTN,\$ERRPC,TCSR
1546	001314	000000	0	
1547				
1548	001316	020626	EM16	;'CAN NOT SET BIT2 OF TCSR''
1549	001320	024265	DH1	;'TEST# ERR PC TCSR''
1550	001322	025046	DT1	;\$TESTN,\$ERRPC,TCSR
1551	001324	000000	0	
1552				
1553	001326	020657	EM17	;'CAN NOT CLEAR BIT2 OF TCSR''
1554	001330	024265	DH1	;'TEST# ERR PC TCSR''
1555	001332	025046	DT1	;\$TESTN,\$ERRPC,TCSR
1556	001334	000000	0	

1557	001336	020712	EM20	:'RESET DID NOT CLEAR BIT2 OF TCSR''
1558	001340	024265	DH1	:'TEST# ERR PC TCSR''
1559	001342	025046	DT1	:\$TESTN,\$ERRPC,TCSR
1560	001344	000000	0	
1561				
1562	001346	020753	EM21	:'BIT6 OF TCSR NOT CLEAR AFTER RESET2
1563	001350	024265	DH1	:'TEST# ERR PC TCSR''
1564	001352	025046	DT1	:\$TESTN,\$ERRPC,TCSR
1565	001354	000000	0	
1566				
1567	001356	021016	EM22	:'XMIT INTERRUPT WITH PRIORITY 7''
1568	001360	024265	DH1	:'TEST# ERR PC TCSR''
1569	001362	025046	DT1	:\$TESTN,\$ERRPC,TCSR
1570	001364	000000	0	
1571				
1572	001366	021053	EM23	:'CAN NOT SET BIT6 OF TCSR''
1573	001370	024265	DH1	:'TEST# ERR PC TCSR''
1574	001372	025046	DT1	:\$TESTN,\$ERRPC,TCSR
1575	001374	000000	0	
1576				
1577	001376	021104	EM24	:'CAN NOT CLEAR BIT6 OF TCSR''
1578	001400	024265	DH1	:'TEST# ERR PC TCSR''
1579	001402	025046	DT1	:\$TESTN,\$ERRPC,TCSR
1580	001404	000000	0	
1581				
1582	001406	021137	EM25	:'RESET DID NOT CLEAR BIT6 OF TCSR''
1583	001410	024265	DH1	:'TEST# ERR PC TCSR''
1584	001412	025046	DT1	:\$TESTN,\$ERRPC,TCSR
1585	001414	000000	0	
1586				
1587	001416	021200	EM26	:'BIT6 OF RCSR NOT CLEAR AFTER RESET''
1588	001420	024337	DH6	:'TEST# ERR PC RCSR''
1589	001422	025066	DT6	:\$TESTN,\$ERRPC,RCSR
1590	001424	000000	0	
1591				
1592	001426	021243	EM27	:'RCVR INTERRUPT WITH PRIORITY 7''
1593	001430	024337	DH6	:'TEST# ERR PC RCSR''
1594	001432	025066	DT6	:\$TESTN,\$ERRPC,RCSR
1595	001434	000000	0	
1596				
1597	001436	021302	EM30	:'CAN NOT SET BIT6 OF RCSR''
1598	001440	024337	DH6	:'TEST# ERR PC RCSR''
1599	001442	025066	DT6	:\$TESTN,\$ERRPC,RCSR
1600	001444	000000	0	

1601	001446	021333	EM31	:'CAN NOT CLEAR BIT6 OF RCSR''
1602	001450	024337	DH6	:'TEST# ERR PC RCSR''
1603	001452	025066	DT6	:\$TESTN,\$ERRPC,RCSR
1604	001454	000000	0	
1605				
1606	001456	021366	EM32	:'CAN NOT CLEAR BIT6 OF RCSR WITH RESET2
1607	001460	024337	DH6	:'TEST# ERR PC RCSR''
1608	001462	025066	DT6	:\$TESTN,\$ERRPC,RCSR
1609	001464	000000	0	
1610				
1611	001466	021434	EM33	:'BIT6 OF LKS NOT CLEAR AFTER RESET''
1612	001470	024411	DH10	:'TEST# ERR PC LKS''
1613	001472	025106	DT10	:\$TESTN,\$ERRPC,LKS
1614	001474	000000	0	
1615				
1616	001476	021476	EM34	:'LKS INTERRUPT WITH PRIORITY 7''
1617	001500	024411	DH10	:'TEST# ERR PC LKS''
1618	001502	025106	DT10	:\$TESTN,\$ERRPC,LKS
1619	001504	000000	0	
1620				
1621	001506	021534	EM35	:'CAN NOT SET BIT6 OF LKS''
1622	001510	024411	DH10	:'TEST# ERR PC LKS''
1623	001512	025106	DT10	:\$TESTN,\$ERRPC,LKS
1624	001514	000000	0	
1625				
1626	001516	021564	EM36	:'CAN NOT CLEAR BIT6 OF LKS''
1627	001520	024411	DH10	:'TEST# ERR PC LKS''
1628	001522	025106	DT10	:\$TESTN,\$ERRPC,LKS
1629	001524	000000	0	
1630				
1631	001526	021616	EM37	:'RESET DID NOT CLEAR BIT6 OF LKS''
1632	001530	024411	DH10	:'TEST# ERR PC LKS''
1633	001532	025106	DT10	:\$TESTN,\$ERRPC,LKS
1634	001534	000000	0	
1635				
1636	001536	021656	EM40	:'DUAL ADDRESSING ERROR''
1637	001540	024435	DH40	:'TEST# ERR PC GOOD BAD''
1638	001542	025116	DT40	:\$TESTN,\$ERRPC,\$GDADR,\$BDCSR
1639	001544	000000	0	
1640				
1641	001546	021704	EM41	:'BIT7 OF LKS NOT SET AFTER RESET2
1642	001550	024411	DH10	:'TEST# ERR PC LKS''
1643	001552	025106	DT10	:\$TESTN,\$ERRPC,LKS
1644	001554	000000	0	

1645	001556	021744	EM42	:'CAN NOT CLEAR BIT7 OF LKS''
1646	001560	024411	DH10	:'TEST# ERR PC LKS''
1647	001562	025106	DT10	:\$TESTN,\$ERRPC,LKS
1648	001564	000000	0	
1649				
1650	001566	021776	EM43	:'BIT7 OF LKS DOES NOT SET''
1651	001570	024411	DH10	:'TEST# ERR PC LKS''
1652	001572	025106	DT10	:\$TESTN,\$ERRPC,LKS
1653	001574	000000	0	
1654				
1655	001576	022027	EM44	:'RTC INTERRUPT AT PRIORITY 7''
1656	001600	024411	DH10	:'TEST# ERR PC LKS''
1657	001602	025106	DT10	:\$TESTN,\$ERRPC,LKS
1658	001604	000000	0	
1659				
1660	001606	022063	EM45	:'RTC INTERRUPTS WHEN DISABLED''
1661	001610	024411	DH10	:'TEST# ERR PC LKS''
1662	001612	025106	DT10	:\$TESTN,\$ERRPC,LKS
1663	001614	000000	0	
1664				
1665	001616	022120	EM46	:'RTC INTERRUPT DID NOT OCCUR''
1666	001620	024411	DH10	:'TEST# ERR PC LKS''
1667	001622	025106	DT10	:\$TESTN,\$ERRPC,LKS
1668	001624	000000	0	
1669				
1670	001626	022120	EM47	:'RTC INTERRUPT DID NOT OCCUR''
1671	001630	024411	DH10	:'TEST# ERR PC LKS''
1672	001632	025106	DT10	:\$TESTN,\$ERRPC,LKS
1673	001634	000000	0	
1674				
1675	001636	022154	EM50	:'RTC DOUBLE INTERRUPT''
1676	001640	024411	DH10	:'TEST# ERR PC LKS''
1677	001642	025106	DT10	:\$TESTN,\$ERRPC,LKS
1678	001644	000000	0	
1679				
1680	001646	022201	EM51	:'RESET DID NOT CLEAR RTC INTERRUPT''
1681	001650	024411	DH10	:'TEST# ERR PC LKS''
1682	001652	025106	DT10	:\$TESTN,\$ERRPC,LKS
1683	001654	000000	0	
1684				
1685	001656	022231	EM52	:'RTC INTERRUPT DID NOT CLEAR WITH BIT7 OF LKS''
1686	001660	024411	DH10	:'TEST# ERR PC LKS''
1687	001662	025106	DT10	:\$TESTN,\$ERRPC,LKS
1688	001664	000000	0	

1689	001666	022306	EM53	
1690	001670	024471	DH53	:'TEST# ERR PC LKS CNT1 CNT2''
1691	001672	025130	DT53	:\$TESTN,\$ERRPC,LKS,FIRST,SECND
1692	001674	000000	0	
1693				
1694	001676	022332	EM54	:'XMIT INTERRUPTS WHEN DISABLED''
1695	001700	024265	DH1	:'TEST# ERR PC TCSR''
1696	001702	025046	DT1	:\$TESTN,\$ERRPC,TCSR
1697	001704	000000	0	
1698				
1699	001706	022470	EM55	:'XMIT DID NOT INTERRUPT''
1700	001710	024265	DH1	:'TEST# ERR PC TCSR''
1701	001712	025046	DT1	:\$TESTN,\$ERRPC,TCSR
1702	001714	000000	0	
1703				
1704	001716	022370	EM56	:'XMIT INTERRUPT AT PRIORITY 7''
1705	001720	024265	DH1	:'TEST# ERR PC TCSR''
1706	001722	025046	DT1	:\$TESTN,\$ERRPC,TCSR
1707	001724	000000	0	
1708				
1709	001726	022426	EM57	:'XMIT INTERRUPTS WITH ENABLE CLEAR''
1710	001730	024265	DH1	:'TEST# ERR PC TCSR''
1711	001732	025046	DT1	:\$TESTN,\$ERRPC,TCSR
1712	001734	000000	0	
1713				
1714	001736	022470	EM60	:'XMIT DID NOT INTERRUPT''
1715	001740	024265	DH1	:'TEST# ERR PC TCSR''
1716	001742	025046	DT1	:\$TESTN,\$ERRPC,TCSR
1717	001744	000000	0	
1718				
1719	001746	022517	EM61	:'XMIT RE-INTERRUPTED''
1720	001750	024265	DH1	:'TEST# ERR PC TCSR''
1721	001752	025046	DT1	:\$TESTN,\$ERRPC,TCSR
1722	001754	000000	0	
1723				
1724	001756	022543	EM62	:'LOADING TBUF DID NOT CLEAR INTERRUPT''
1725	001760	024265	DH1	:'TEST# ERR PC TCSR''
1726	001762	025046	DT1	:\$TESTN,\$ERRPC,TCSR
1727	001764	000000	0	
1728				
1729	001766	022610	EM63	:'RCVR ACTIVE NOT SET''
1730	001770	024337	DH6	:'TEST# ERR PC RCSR''
1731	001772	025066	DT6	:\$TESTN,\$ERRPC,RCSR
1732	001774	000000	0	

1733	001776	022634	EM64	:'RECEIVER DONE NEVER SET''
1734	002000	024337	DH6	:'TEST# ERR PC RCSR''
1735	002002	025066	DT6	:\$TESTN,\$ERRPC,RCSR
1736	002004	000000	0	
1737				
1738	002006	022660	EM65	:'RCVR ACTIVE NOT CLEARED WITH DONE SET2
1739	002010	024337	DH6	:'TEST# ERR PC RCSR''
1740	002012	025066	DT6	:\$TESTN,\$ERRPC,RCSR
1741	002014	000000	0	
1742				
1743	002016	022726	EM66	:'RESET DID NOT CLEAR RCVR DONE''
1744	002020	024337	DH6	:'TEST# ERR PC RCSR''
1745	002022	025066	DT6	:\$TESTN,\$ERRPC,RCSR
1746	002024	000000	0	
1747				
1748	002026	022764	EM67	:'RDR ENABLE SET DID NOT CLEAR RCVR DONE''
1749	002030	024337	DH6	:'TEST# ERR PC RCSR''
1750	002032	025066	DT6	:\$TESTN,\$ERRPC,RCSR
1751	002034	000000	0	
1752				
1753	002036	023027	EM70	:'READING RBUF DID NOT CLEAR RCVR DONE''
1754	002040	024337	DH6	:'TEST# ERR PC RCSR''
1755	002042	025066	DT6	:\$TESTN,\$ERRPC,RCSR
1756	002044	000000	0	
1757				
1758	002046	023074	EM71	:'RCVR INTERRUPTS WITH ENABLE CLEAR''
1759	002050	024337	DH6	:'TEST# ERR PC RCSR''
1760	002052	025066	DT6	:\$TESTN,\$ERRPC,RCSR
1761	002054	000000	0	
1762				
1763	002056	023243	EM72	:'RCVR DID NOT INTERRUPT''
1764	002060	024337	DH6	:'TEST# ERR PC RCSR''
1765	002062	025066	DT6	:\$TESTN,\$ERRPC,RCSR
1766	002064	000000	0	
1767				
1768	002066	023136	EM73	:'RCVR INTERRUPTS AT PRIORITY 7''
1769	002070	024337	DH6	:'TEST# ERR PC RCSR''
1770	002072	025066	DT6	:\$TESTN,\$ERRPC,RCSR
1771	002074	000000	0	
1772				
1773	002076	023174	EM74	:'RCVR INTERRUPT REQUEST PASSED WITH ENABLE CLEAR''
1774	002100	024337	DH6	:'TEST# ERR PC RCSR''
1775	002102	025066	DT6	:\$TESTN,\$ERRPC,RCSR
1776	002104	000000	0	

1777	002106	023243	EM75	:'RCVR DID NOT INTERRUPT''
1778	002110	024337	DH6	:'TEST# ERR PC RCSR''
1779	002112	025066	DT6	:\$TESTN,\$ERRPC,RCSR
1780	002114	000000	0	
1781	002116	023272	EM76	:'RECEIVER RE-INTERRUPTED''
1782	002120	024337	DH6	:'TEST# ERR PC RCSR''
1783	002122	025066	DT6	:\$TESTN,\$ERRPC,RCSR
1784	002124	000000	0	
1785				
1786	002126	023316	EM77	:'READING RBUF DID NOT CLEAR INTERRUPT''
1787	002130	024337	DH6	:'TEST# ERR PC RCSR''
1788	002132	025066	DT6	:\$TESTN,\$ERRPC,RCSR
1789	002134	000000	0	
1790				
1791	002136	023363	EM100	:'RESET DID NOT CLEAR RCVR INTERRUPT''
1792	002140	024337	DH6	:'TEST# ERR PC RCSR''
1793	002142	025066	DT6	:\$TESTN,\$ERRPC,RCSR
1794	002144	000000	0	
1795				
1796				
1797	002146	023426	EM101	:'"OR" FLAG DID NOT SET''
1798	002150	024337	DH6	
1799	002152	025066	DT6	:\$TESTN,\$ERRPC,RCSR
1800	002154	000000	0	
1801				
1802	002156	023454	EM102	:'"ERROR" NOT SET WITH "OR" FLAG''
1803	002160	024337	DH6	:'TEST# ERR PC RCSR''
1804	002162	025066	DT6	:\$TESTN,\$ERRPC,RCSR
1805	002164	000000	0	
1806	002166	023513	EM103	:'BREAK DID NOT TRANSMIT ALL ZEROES''
1807	002170	024536	DH103	:'TEST# ERR PC RCSR DATA''
1808	002172	025144	DT103	:\$TESTN,\$ERRPC,RCSR,\$BDDAT
1809	002174	000000	0	
1810				
1811	002176	023551	EM104	:'BREAK DID NOT SET FRAMING ERROR''
1812	002200	024337	DH6	:'TEST# ERR PC RCSR''
1813	002202	025066	DT6	:\$TESTN,\$ERRPC,RCSR
1814	002204	000000	0	
1815				
1816	002206	023606	EM105	:'DATA COMPARE ERROR''
1817	002210	024573	DH105	:'TEST# ERR PC RCSR GOOD BAD''
1818	002212	025156	DT105	:\$TESTN,\$ERRPC,RCSR,GD,BD
1819	002214	000000	0	

1820	002216	023631	EM106	:'LOOP-BACK DATA COMPARE ERROR''
1821	002220	024573	DH105	:'TEST# ERR PC RCSR GOOD BAD''
1822	002222	025156	DT105	:\$TESTN,\$ERRPC,RCSR,GD,BD
1823	002224	000000	0	
1824				
1825	002226	023666	EM107	:'TIMEOUT IN EXERCISER TEST''
1826	002230	024337	DH6	:'TEST# ERR PC RCSR''
1827	002232	025066	DT6	:\$TESTN,\$ERRPC,RCSR
1828	002234	000000	0	
1829				
1830	002236	023720	EM110	:'INCORRECT RECEIVE COUNT
1831	002240	024637	DH110	:'TEST# ERR PC RCSR TRANS RCV''
1832	002242	025172	DT110	:\$TESTN,\$ERRPC,RCSR,XMTCNT,RCVCNT
1833	002244	000000	0	
1834				
1835	002246	023750	EM111	:'DATA COMPARE ERROR IN EXERCISER''
1836	002250	024573	DH105	:'TEST# ERR PC RCSR GOOD BAD''
1837	002252	025156	DT105	:\$TESTN,\$ERRPC,RCSR,GD,BD
1838	002254	000000	0	
1839				
1840	002256	024010	EM112	:'TRAP CATCHER''
1841	002260	024703	DH112	:'TEST# ERR PC RCSR OLDPC TRAP ADR''
1842	002262	025206	DT112	:\$TESTN,\$ERRPC,RCSR,OLDPC,BDVECT
1843	002264	000000	0	
1844				
1845	002266	024025	EM113	:'NO CLK INTERRUPT IN EXERCISER''
1846	002270	024411	DH10	:'TEST# ERR PC LKS''
1847	002272	025106	DT10	:\$TESTN,\$ERRPC,LKS
1848	002274	000000	0	
1849				
1850	002276	024063	EM114	:'"ERROR" NOT SET WITH "FR" FLAG''
1851	002300	024337	DH6	:'TEST# ERR PC RCSR''
1852	002302	025066	DT6	:\$TESTN,\$ERRPC,RCSR
1853	002304	000000	0	
1854				
1855	002306	024122	EM115	:'RCV ACTIVE NOT CLEAR WITH INIT
1856	002310	024337	DH6	:'TEST# ERR PC RCSR''
1857	002312	025066	DT6	:\$TESTN,\$ERRPC,RCSR
1858	002314	000000	0	
1859				
1860	002316	024161	EM116	:'RCV ACTIVE WITHOUT "START" BIT
1861	002320	024337	DH6	:'TEST# ERR PC RCSR''
1862	002322	025066	DT6	:\$TESTN,\$ERRPC,RCSR
1863	002324	000000	0	
1864				
1865	002326	024220	EM117	:'RDR ENABLE NOT CLEAR WITH RCV ACTIVE
1866	002330	024337	DH6	:'TEST# ERR PC RCSR''
1867	002332	025066	DT6	:\$TESTN,\$ERRPC,RCSR
1868	002334	000000	0	
1869				
1870				
1871	002336	000000		:'STORAGE LOCATIONS
1872	002340	000000		FIRST: .WORD 0
1873	002342	000000		LOC1: .WORD 0
1874	002344	000000		LOC2: .WORD 0
1875	002346			SAVE0: .WORD 0
1876	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)



1877	002432	000000				JIMSTK: .WORD 0	
1878	002434	000000				SAVEPS: .WORD 0	:SAVE PSW AREA
1879	002436	000000				OLDSUM: .WORD 0	:CHECKSUM STORAGE
1880	002440	000000				RCVCNT: .WORD 0	
1881	002442	000000				XMTCNT: .WORD 0	
1882	002444	000000				CLKCNT: .WORD 0	
1883	002446	000000				STPSW: .WORD 0	
1884	002450	000000				SRPSW: .WORD 0	
1885	002452	000000				SCPSW: .WORD 0	
1886	002454					BUF: .BLKW 44	
1887	002564	020	000			CNTLP: .ASCIZ <20>	
1888	002566	136	120	000		PROMPT: .ASCIZ /*P/<0><CR><LF>/CONSOLE/<CR><LF>/>>>/<377>	
1889	002610	124	057	101		TA: .ASCIZ \$T/A\$<CR><LF>	
1890							
1891						.EVEN	
1892	002616	000000				SECND: .WORD 0	
1893	002620	000000				OLDPC: .WORD 0	
1894	002622	000000				BDVECT: .WORD 0	
1895							
1896							
1897							
1898							
1899							
1900	002624	000000				CTSTFL: .WORD 0	:CONSLE UNDER TEST FLAG
1901	002626	000000				TMP1: .WORD 0	:TEMP LOCATION FOR TABLE OFFSETS
1902	002630	000000				TMP2: .WORD 0	:TEMP LOCATION FOR DEVICE COUNT
1903	002632	000000				TMP3: .WORD 0	:LOCATION FOR DEVICE MAP BIT TEST MASK
1904						:REGISTER AND VECTOR ADDRESSES FOR THE DL-11W UNDER TEST	
1905							
1906	002634	000000				RCSR: .WORD 0	
1907	002636	000000				RBUF: .WORD 0	
1908	002640	000000				TCSR: .WORD 0	
1909	002642	000000				TBUF: .WORD 0	
1910	002644	000000				RVECT: .WORD 0	
1911	002646	000000				RPSW: .WORD 0	
1912	002650	000000				TVECT: .WORD 0	
1913	002652	000000				TPSW: .WORD 0	
1914							
1915						:CONSOLE REGISTER AND VECTOR ADDRESSES FOR THE DL-11W	
1916							
1917	002654	177560				CRCSR: 177560	:ADDRESS OF RECEIVER COMMAND/STATUS REGISTER
1918	002656	177562				CRBUF: 177562	:ADDRESS OF RECEIVER BUFFER
1919	002660	177564				CTCSR: 177564	:ADDRESS OF TRANSMITTER COMMAND/STATUS REGISTER
1920	002662	177566				CTBUF: 177566	:ADDRESS OF TRANSMITTER BUFFER
1921	002664	000060				CRVECT: 60	:RECEIVER INTERRUPT VECTOR
1922	002666	000062				CRPSW: 62	
1923	002670	000064				CTVECT: 64	:TRANSMITTER INTERRUPT VECTOR
1924	002672	000066				CTPSW: 66	
1925							
1926						:REAL TIME CLOCK REGISTER AND VECTOR ADDRESSES	
1927	002674	177546				LKS: .WORD 177546	
1928	002676	000100				RTCVT: .WORD 100	
1929	002700	000102				RTCPSTW: .WORD 102	
1930							
1931	002702					ADRTBL: .BLKW 20	
1932	002742					VCTTBL: .BLKW 20	
1933	003002	000000				FLAG44: .WORD 0	

1934 003004 176500  
 1935 003006 176502  
 1936 003010 176504  
 1937 003012 176506  
 1938  
 1939  
 1940  
 1941  
 1942 003014 012702 002702  
 1943 003020 013700 001142  
 1944 003024 010001  
 1945 003026 062701 000170  
 1946 003032 010022  
 1947 003034 062700 000010  
 1948 003040 020001  
 1949 003042 003773  
 1950 003044 000207  
 1951  
 1952  
 1953  
 1954 003046 005037 001070  
 1955 003052 005037 001066  
 1956 003056 005037 001072  
 1957 003062 005037 002624  
 1958 003066 005037 001076  
 1959 003072 005037 001100  
 1960 003076 005037 003002  
 1961 003102 005737 001112  
 1962 003106 001003  
 1963 003110 012737 000400 001112  
 1964 003116 012737 000006 000004  
 1965 003124 012737 000003 000006  
 1966  
 1967

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  
 1\$: MOV R0,(R2)+ ;MOVE DEVICE ADDRESS TO TABLE  
 ADD #10,R0 ;POINT R0 TO NEXT DEVICE ADDRESS  
 CMP R0,R1 ;FINISHED GENERATING TABLE?  
 BLE 1\$ ;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 \$USWR ;IS \$USWR LOADED?  
 BNE 1\$ ;BR IF YES  
 MOV #400,\$USWR ;ELSE, DEFAULT TO \$USWR=400  
 1\$: MOV #6,@#4 ;INITIALIZE TIMEOUT VECTORS TO TRAP  
 MOV #3,@#6 ;CATCHER ROUTINE

.SBTTL INITIALIZE THE COMMON TAGS

::CLEAR THE COMMON TAGS (\$CMTAG) AREA  
 MOV #CMTAG,R6 ;:FIRST LOCATION TO BE CLEARED  
 CLR (R6)+ ;:CLEAR MEMORY LOCATION  
 CMP #SWR,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 #SPWRDN,@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  
 MOV #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

```

INITIALIZE THE COMMON TAGS

003272 012737 003326 000004      MOV    #64$,@#ERRVEC    ;;SET UP ERROR VECTOR
003300 012737 177570 001040      MOV    #DSWR,SWR        ;;SETUP FOR A HARDWARE SWICH REGISTER
003306 012737 177570 001042      MOV    #DDISP,DISPLAY   ;;AND A HARDWARE DISPLAY REGISTER
003314 022777 177777 175516      CMP    #-1,@SWR         ;;TRY TO REFERENCE HARDWARE SWR
003322 001012                    BNE    66$              ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                ;;AND THE HARDWARE SWR IS NOT = -1

003324 000403                    BR     65$              ;;BRANCH IF NO TIMEOUT
003326 012716 003334            64$:  MOV    #65$, (SP)      ;;SET UP FOR TRAP RETURN
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$:

1968
1969
1970
1971 003376 013746 000010      MOV    @#10,-(SP)      ;** SAVE VECTOR
1972 003402 012737 003430 000010      MOV    #10$,@#10      ;** SET UP FOR TRAP
1973 003410 000007                    MFPT                    ;** WHAT CPU??
1974 003412 122700 000001                    CMPB  #1,R0            ;** ARE WE A 11/44
1975 003416 001007                    BNE   11$              ;** NO GO RESET VECTOR
1976 003420 052737 000001 003002      BIS   #1,@#FLAG44    ;** YES SET FLAG
1977 003426 000403                    BR    11$              ;** SKIP RTI
1978 003430 012716 003436            10$:  MOV    #11$, (SP)     ;** SET STACK RETURN
1979 003434 000002                    RTI                      ;** RETURN
1980 003436 012637 000010            11$:  MOV    (SP)+,@#10     ;** RESTORE VECTOR
1981 003442 032777 000020 175370      BIT   #BIT4,@SWR     ;TEST CLOCK ONLY?
1982 003450 001404                    BEQ   INIT            ;BR IF NOT
1983 003452 005237 002624                    INC   CTSTFL          ;ELSE, SET CONSOLE TEST FLAG TO ENABLE CLOCK TESTS
1984 003456 000137 004620                    JMP   ID              ; AND JUMP TO TYPE PROGRAM ID
1985 003462 132737 000001 001106  INIT:  BITB  #BIT0,$ENV     ;CHECK IF ON APT
1986 003470 001404                    BEQ   MANL            ;BR IF NOT APT
1987 003472 132737 000200 001107      BITB  #BIT7,$ENVM    ;DID APT SIZE
1988 003500 001056                    BNE   APTSZD          ;BR, IF APT SIZED
1989 003502 032777 000040 175330  MANL:  BIT   #BIT5,@SWR     ;WAS '$DEVN' MANUALLY SET?
1990 003510 001052                    BNE   APTSZD          ;IF YES, SKIP SELF-SIZING
1991
1992 003512 004737 003014            SIZE:  JSR   PC,DEVADR     ;GENERATE DEVICE ADDRESS TABLE
1993 003516 005037 002630                    CLR   TMP2            ;CLR TEMP LOCATION TO KEEP DEVICE COUNT
1994 003522 005037 001144                    CLR   $DEVN           ;CLEAR DEVICE MAP
1995 003526 013703 000004                    MOV   @#4,R3          ;SAVE TIMEOUT VECTOR
1996 003532 012737 003562 000004      MOV   #4$,@#4        ;SET TIMEOUT POINTER
1997 003540 013700 001142                    MOV   $BASE,R0        ;LOAD BASE ADDRESS
1998 003544 062700 000160                    ADD   #160,R0         ;POINT R0 TO UNIT #15 (UNIT#0=CONSOLE)
1999 003550 005710            3$:   TST   (R0)           ;CHECK FOR DEVICE EXISTANCE
2000 003552 005237 001144                    INC   $DEVN           ;INDICATE DEVICE EXISTANCE IN DEVICE MAP
2001 003556 005237 002630                    INC   TMP2            ;INCREMENT DEVICE COUNT
2002 003562 012706 001000            4$:   MOV   #1000,SP       ;RESET STACK POINTER
2003 003566 006337 001144                    ASL   $DEVN           ;ADJUST DEVICE MAP FOR NEXT UNIT CHECK
2004 003572 162700 000010                    SUB   #10,R0          ;POINT R0 TO NEXT DEVICE NUMBER
2005 003576 023700 001142                    CMP   $BASE,R0        ;FINISHED SIZING?
2006 003602 003762                    BLE   3$              ;BR, IF BASE ADDRESS HAS NOT BEEN CHECKED
2007 003604 013700 002654                    MOV   CRCSR,R0        ;LOAD CONSOLE DEVICE ADDRESS

```

```

2008 003610 012737 003630 000004      MOV      #5$,@#4      ;SET UP TIMEOUT POINTER
2009 003616 005710                TST      (R0)        ;TEST FOR CONSOLE EXISTANCE
2010 003620 005237 001144                INC      $DEV M      ;INDICATE CONSOLE EXISTANCE IN DEVICE MAP
2011 003624 005237 002630                INC      TMP2        ;INCREMENT DEVICE COUNT
2012 003630 010337 000004      5$:     MOV      R3,@#4      ;RESTORE TIMEOUT VECTOR
2013
2014 003634 000415                BR       VCTADR      ;BR TO GENERATE VECTOR ADDRESS TABLE
2015
2016 003636 005037 002630      APTSZD: CLR      TMP2        ;CLEAR TEMP LOCATION TO KEEP DEVICE CNT
2017 003642 013702 001144                MOV      $DEV M,R2   ;MOVE DEVICE MAP TO R2
2018 003646 005702      TSTDVM: TST      R2        ;TEST MSB OF DEVICE MAP
2019 003650 100002                BPL      1$          ;BR, IF MSB IS ZERO
2020 003652 005237 002630                INC      TMP2        ;INCREMENT DEVICE COUNT, IF MSB=1
2021 003656 006302      1$:     ASL      R2        ;SHIFT NEXT BIT INTO MSB POSITION
2022 003660 001401                BEQ      DVADT       ;BR, IF NO OTHER BITS ARE SET IN $DEV M
2023 003662 000771                BR       TSTDVM      ;CONTINUE CHECKING $DEV M, IF MORE BITS SET
2024 003664 004737 003014      DVADT: JSR      PC,DEVADR ;GENERATE DEVICE ADDRESS TABLE
2025
2026      ;GENERATE VECTOR ADDRESS TABLE
2027
2028 003670 012702 002742      VCTADR: MOV      #VCTTBL,R2   ;GET LOCATION OF VECTOR TABLE
2029 003674 013700 001136                MOV      @#$VECT1,R0 ;COPY BASE VECTOR
2030 003700 042700 177000                BIC      #177000,R0   ;CLEAR UPPER BITS
2031 003704 010001                MOV      R0,R1        ;
2032 003706 062701 000170                ADD      #170,R1      ;POINT R1 TO LAST DEVICE VECTOR
2033 003712 010022      1$:     MOV      R0,(R2)+     ;PUT VECTOR ADDRESS IN TABLE
2034 003714 062700 000010                ADD      #10,R0       ;POINT R0 TO NEXT VECTOR ADDRESS
2035 003720 020001                CMP      R0,R1        ;FINISHED GENERATING VECTOR TABLE?
2036 003722 003773                BLE      1$          ;BR, IF LAST VECTOR IS NOT LOADED
2037
2038      ;MOVE DEVICE COUNT INTO DEVICE COUNT MESSAGE
2039
2040 003724 013700 002630                MOV      TMP2,R0     ;COPY DEVICE COUNT INTO R0
2041 003730 005001                CLR      R1          ;CLEAR AUXILIARY REGISTER
2042 003732 000300                SWAB    R0          ;PUT DEVICE COUNT IN UPPER BYTE OF R0
2043 003734 006300                ASL      R0          ;MOVE MSB OF COUNT INTO
2044 003736 006300                ASL      R0          ;MSB OF R0
2045 003740 006300      SHIFT: ASL      R0          ;PUT MSB OF COUNT INTO CARRY
2046 003742 106101                ROLB    R1          ;MOVE MSB OF COUNT INTO R1
2047 003744 006300                ASL      R0          ;MOVE NEXT BIT TO CARRY
2048 003746 106101                ROLB    R1          ;MOVE INTO R1
2049 003750 006300                ASL      R0          ;MOVE LAST BIT OF DIGIT
2050 003752 106101                ROLB    R1          ;INTO R1
2051 003754 062701 000060                ADD      #60,R1      ;CONVERT DIGIT TO ASCII
2052 003760 000301                SWAB    R1          ;MOVE DIGIT TO UPPER BYTE
2053 003762 032701 000020                BIT      #BIT4,R1    ;HAVE BOTH DIGITS BEEN MOVED TO R1?
2054 003766 001764                BEQ     SHIFT       ;BR, IF NOT
2055 003770 010137 025016                MOV      R1,M2A     ;MOVE DEVICE COUNT TO OUTPUT MESSAGE
2056
2057
2058 003774 052737 000002 002632 BEGIN: BIS      #BIT1,TMP3   ;SET UP BIT MASK TO TEST $DEV M FOR DEVICES EXCEPT CONSOLE
2059 004002 005037 002626                CLR      TMP1        ;CLEAR LOCATION TO STORE TABLE OFFSETS
2060 004006 032737 000001 001144                BIT      #BIT0,$DEV M ;IS CONSOLE TO BE TESTED?
2061 004014 001001                BNE     TCONS       ;BR, IF CONSOLE IS TO BE TESTED
2062 004016 000414                BR       TSTDVM      ;BR, TO TEST OTHER DEVICES
2063 004020 005237 002624      TCONS: INC      CTSTFL   ;INDICATE CONSOLE UNDER TEST
2064 004024 012700 002654                MOV      #CRCSR,R0   ;SET UP CONSOLE DEVICE ADDRESSES

```

```

2065 004030 012701 002634          MOV    #RCSR,R1      ;POINT R1 TO UUT ADDRESS TABLE
2066 004034 012021 002634          1$:  MOV    (R0)+,(R1)+ ;TRANSFER CONSOLE ADDRESSES
2067 004036 022701 002652          CMP    #TPSW,R1     ;FINISHED TRANSFER?
2068 004042 002374 002652          BGE    1$           ;BR, IF NOT
2069 004044 000137 004172          JMP    TST1        ;GO TEST CONSOLE INTERFACE
2070
2071                                ;PREPARE ADDRESSES AND VECTORS FOR UUT
2072 004050 033737 002632 001144  TSTDEV: BIT    TMP3,$DEVM ;CHECK TO SEE IF DEVICE IS TO BE TESTED
2073 004056 001010 002632          BNE    SETADR      ;BR, IF YES
2074 004060 006337 002632          ASL    TMP3        ;SHIFT MASK TO CHECK NEXT $DEVM BIT
2075 004064 062737 000002 002626  ADD    #2,TMP1     ;INCREMENT TABLE INDEX
2076 004072 005237 001100          INC    $UNIT       ;INCREMENT UNIT NUMBER
2077 004076 000764 001100          BR     TSTDEV      ;GO TEST NEXT BIT OF DEVICE MAP
2078
2079 004100 005237 001100          SETADR: INC    $UNIT ;UPDATE UNIT NUMBER
2080 004104 006337 002632          ASL    TMP3        ;UPDATE DEVICE MAP TEST MASK
2081 004110 013702 002626          MOV    TMP1,R2     ;MOVE TABLE OFFSET TO R2
2082 004114 062737 000002 002626  ADD    #2,TMP1     ;UPDATE TABLE OFFSET FOR NEXT DEVICE
2083 004122 016200 002702          MOV    ADRTBL(R2),R0 ;PUT UUT ADDRESS INTO R0
2084 004126 012701 002634          MOV    #RCSR,R1   ;POINT R1 TO STORAGE AREA FOR UUT ADDRESSES
2085 004132 010021 002634          ADR:  MOV    R0,(R1)+ ;TRANSFER UUT ADDRESS
2086 004134 062700 000002          ADD    #2,R0      ;POINT TO NEXT UUT REGISTER
2087 004140 030027 000006          BIT    R0,#6      ;FINISHED TRANSFER?
2088 004144 001372 000006          BNE    ADR        ;BR, IF NOT
2089
2090 004146 016200 002742          VECT: MOV    VCTTBL(R2),R0 ;PUT UUT VECTOR INTO R0
2091 004152 010021 002742          MOV    R0,(R1)+  ;TRANSFER UUT VECTORS TO ACTIVE TABLE AREA
2092 004154 062700 000002          ADD    #2,R0      ;POINT TO NEXT VECTOR
2093 004160 030027 000006          BIT    R0,#6      ;FINISHED TRANSFER?
2094 004164 001372 000006          BNE    VECT       ;BR, IF NOT
2095 004166 000137 004172          JMP    TST1        ;GO TEST DEVICE

```

2096

.SBTTL TEST # 1 - TEST ABILITY TO REFERENCE TCSR  
 :\*\*\*\*\*  
 :\*TEST 1 TEST ABILITY TO REFERENCE TCSR  
 :\*\*\*\*\*

2097 004172 000004  
 2098 004174 013703 000004  
 2099 004200 012737 004214 000004  
 2100 004206 005777 176426  
 2101 004212 000412  
 2102 004214 022626  
 2103 004216 005737 002624  
 2104 004222 001002  
 2105 004224 104001  
 2106 004226 000404  
 2107 004230  
 004230 004737 015714  
 004234 000001  
 2108 004236 000000  
 2109 004240 010337 000004  
 2110  
 2111  
 2112

TST1: SCOPE  
 MOV @#4,R3 ;SAVE TIMEOUT VECTOR  
 MOV #1\$,@#4 ;SET UP TIMEOUT VECTOR  
 TST @TCSR ;REFERENCE THE XMIT COMMAND/STATUS REG.  
 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 +1 ;REPORT ERROR TO APT & TTY  
 BR 4\$ ;BR TO END OF TEST  
 2\$: JSR PC,\$ATY4 ;: ONLY REPORT A FATAL ERROR  
 1 ;: THE ERROR NUMBER (FROM APT LIST)  
 3\$: HALT  
 4\$: MOV R3,@#4 ;RESTORE TIMEOUT VECTOR

2113  
 2114 004244 000004  
 2115 004246 013703 000004  
 2116 004252 012737 004266 000004  
 2117 004260 005777 176356  
 2118 004264 000412  
 2119 004266 022626  
 2120 004270 005737 002624  
 2121 004274 001002  
 2122 004276 104002  
 2123 004300 000404  
 2124 004302  
 004302 004737 015714  
 004306 000002  
 2125 004310 000000  
 2126 004312 010337 000004

```

.SBTTL TEST # 2 - TEST ABILITY TO REFERENCE TBUF
:*****
:*TEST 2 TEST ABILITY TO REFERENCE TBUF
:*****
TST2: SCOPE
      MOV @#4,R3 ;SAVE TIMEOUT VECTOR
      MOV #1$,@#4 ;SET UP TIMEOUT VECTOR
      TST @TBUF ;REFERENCE THE XMIT BUFFER
      BR 4$ ;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,@#4 ;RESTORE TIMEOUT VECTOR
    
```

```

2127 .SBTTL TEST # 3 - TEST THAT BIT2(MAINT. BIT) CAN BE SET & RESET
      :*****
      :*TEST 3 TEST THAT BIT2(MAINT. BIT) CAN BE SET & RESET
      :*****
      TST3: SCOPE
2128 004316 000004 RESET ;CLEAR EVERYTHING
2129 004320 000005 176310 BIT #BIT2,@TCSR ;TEST FOR BIT2 OF TCSR CLEAR
2130 004322 032777 000004 BEQ 3$ ;BR IF CLEAR
2131 004330 001411
2132 004332 005737 002624 TST CTSTFL ;CHECK IF DEVICE IS CONSOLE
2133 004336 001002 BNE 1$ ;IF YES, SKIP ERROR TYPEOUT
2134 004340 104015 ERROR +15 ;BIT2 OF TCSR NOT CLEAR AFTER RESET
2135 004342 000404 BR 3$
2136
2137 004344 1$: JSR PC,$ATY4 ;: ONLY REPORT A FATAL ERROR
      004344 004737 015714 15 ;: THE ERROR NUMBER (FROM APT LIST)
      004350 000015
2138 004352 000000 2$: HALT
2139
2140 004354 052777 000004 176256 3$: BIS #BIT2,@TCSR ;SET BIT2 OF TCSR
2141 004362 032777 000004 176250 BIT #BIT2,@TCSR ;TEST FOR BIT2 SET
2142 004370 001001 BNE 4$ ;BR IF SET
2143
2144 004372 104016 ERROR +16 ;BIT2 OF TCSR WILL NOT SET
2145
2146 004374 042777 000004 176236 4$: BIC #BIT2,@TCSR ;CLEAR BIT2 OF TCSR
2147 004402 032777 000004 176230 BIT #BIT2,@TCSR ;TEST BIT2 CLEAR
2148 004410 001411 BEQ 7$ ;BR IF CLEAR
2149
2150 004412 005737 002624 TST CTSTFL ;CHECK IF DEVICE IS CONSOLE
2151 004416 001002 BNE 5$ ;IF YES, SKIP ERROR TYPEOUT
2152 004420 104017 ERROR +17
2153 004422 000404 BR 7$
2154 004424 5$: JSR PC,$ATY4 ;: ONLY REPORT A FATAL ERROR
      004424 004737 015714 17 ;: THE ERROR NUMBER (FROM APT LIST)
      004430 000017
2155 004432 000000 6$: HALT ;BIT0 OF TCSR WILL NOT CLEAR
2156
2157 004434 052777 000004 176176 7$: BIS #BIT2,@TCSR ;SET BIT2 OF TCSR
2158 004442 000005 RESET ;CLEAR BIT2 WITH RESET
2159 004444 032777 000004 176166 BIT #BIT2,@TCSR ;TEST FOR BIT2 CLEAR
2160 004452 001404 BEQ 10$ ;** IF CLEAR, GO TO NEXT TEST
2161
2162 004454 042777 000004 176156 BIC #BIT2,@TCSR ;CLEAR BIT2, TO PRINT ERROR
2163 004462 104020 ERROR +20 ;RESET DID NOT CLEAR BIT2 OF TCSR
2164
2165
2166 004464 000240 10$: NOP ;**
    
```



```

2167 .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
:*****
TST4: SCOPE
2168 004466 000004 000004 176142 BIS #BIT2,@TCSR ;** ENABLE MAINT. WRAP
2169 004470 052777 000004 176142 CLR @TBUF ;LOAD XBUF
2170 004476 005077 176140 CLR @TBUF ;CHECK DONE
2171 004502 105777 176132 TSTB @TCSR ;BR IF CLEAR
2172 004506 100021 BPL 3$ ;FILL SECOND BUFFER, BECAUSE REFRESH COULD CAUSE
2173 ; FIRST TEST TO FAIL
2174 004510 005077 176126 CLR @TBUF ;FILL DOUBLE BUFFER
2175 004514 105777 176120 TSTB @TCSR ;CHECK DONE
2176 004520 100014 BPL 3$ ;BR IF CLEAR
2177
2178 004522 005737 002624 TST CTSTFL ;CHECK IF DEVICE IS CONSOLE
2179 004526 001005 BNE 1$ ;IF YES, SKIP ERROR TYPEOUT
2180 004530 042777 000004 176122 BIC #BIT2,@CTCSR ;** DISABLE MAINTENANCE MODE FOR
; ** CONSOLE TO ALLOW FOR COMMUNICATION
; ** WITH TERMINAL.
; DONE NOT CLEARED WITH TBUF FULL
; BR TO END OF TEST
2181 004536 104003 ERROR +3
2182 004540 000404 BR 3$
2183 004542 1$: JSR PC,$ATY4 ;: ONLY REPORT A FATAL ERROR
;: THE ERROR NUMBER (FROM APT LIST)
004542 004737 015714 3 HALT ;TCSR 'DONE' NOT CLEARED WITH TBUF FULL
2184 004550 000000 2$: CLR R0 ;CLEAR TIMER
2185 004552 005000 3$: TSTB @TCSR ;CHECK FOR XMIT DONE
2186 004554 105777 176060 4$: BMI ID ;IF DONE SETS, BR TO END OF TEST
2187 004560 100417 INC R0 ;INCREMENT TIMER
2188 004562 005200 BNE 4$ ;BR IF TIMER NOT DONE
2189 004564 001373
2190
2191 004566 005737 002624 TST CTSTFL ;CHECK IF DEVICE IS CONSOLE
2192 004572 001005 BNE 5$
2193 004574 042777 000004 176056 BIC #BIT2,@CTCSR ;** DISABLE MAINTENANCE MODE FOR
; ** CONSOLE TO ALLOW FOR COMMUNICATION
; ** WITH TERMINAL.
; TCSR 'DONE' DOES NOT SET
; BR TO END OF TEST
2194 004602 104004 ERROR +4
2195 004604 000405 BR ID
2196 004606 5$: JSR PC,$ATY4 ;: ONLY REPORT A FATAL ERROR
;: THE ERROR NUMBER (FROM APT LIST)
004606 004737 015714 4 HALT
004612 000004 4$: BR ENDB7 ;** BR TO NEXT TEST,
; ** AND SKIP THE TYPEOUT THAT FOLLOWS
; ** BECAUSE OF THIS FAILURE
2197 004614 000000
2198 004616 000427
2199
2200
2201
2202 004620 ID: BIC #BIT2,@CTCSR ;** DISABLE MAINTENANCE MODE FOR
; ** CONSOLE TO ALLOW FOR COMMUNICATION
; ** WITH TERMINAL.
; UNDER ACT11?
2203 004620 042777 000004 176032 CMP @#42,@#46 ;IF YES, SKIP IDENT. TYPEOUT
2204 004626 023737 000042 000046 BEQ 6$ ;IS THIS THE FIRST PASS?
2205 004634 001412 TST $PASS ;IF NOT BR TO NEXT TEST & SKIP THE IDENTIFICATION TYPEOUTS
2206 004636 005737 001074 BNE 6$ ;IS THIS THE FIRST SUBPASS?
2207 004642 001007 TST $DEVCT ;IF NOT, BR TO NEXT TEST
2208 004644 005737 001076 BNE 6$
2209 004650 001004
    
```

```
2210 004652 104401          TYPE          ;TYPE PROGRAM IDENTIFICATION
2211 004654 024754          M1
2212 004656 104401          TYPE          ;TYPE NUMBER OF DEVICES UNDER TEST
2213 004660 025014          M2
2214 004662 032777 000020 174150 6$: BIT #BIT4,@SWR ;CLOCK TEST ONLY?
2215 004670 001402          BEQ ENDB7 ;** BR IF NOT
2216 004672 000137 005536          JMP TCLOCK ;ELSE, JUMP TO TEST CLOCK
2217 004676          ENDB7:
2218 004676 005000          CLR R0 ;** DELAY ENOUGH TIME TO ALLOW ANY CHARACTERS
      004700 012701 000002          MOV #2,R1 ;** THAT MIGHT BE IN THE PROCESS OF BEING
      004704 005300          DEC R0 ;** TRANSMITTED TO FINISH BEFORE MAINTENANCE
      004706 001376          BNE 40$ ;** WRAP FOR THE UART UNDER TEST IS DISABLED.
      004710 005301          DEC R1 ;** THIS WILL INHIBIT ANY COMMUNICATION
      004712 001374          BNE 40$ ;** TO HARDWARE MEDIA SUCH AS TU58 THAT MIGHT
      ;** BE ATTACHED TO UART UNDER TEST.
```

```
2219          .SBTTL TEST # 5 - TEST THAT TCSR 'DONE' SETS WITH RESET
                :*****
                :*TEST 5          TEST THAT TCSR 'DONE' SETS WITH RESET
                :*****
004714 000004  TST5:  SCOPE
2220 004716 052777 000004 175714  BIS      #BIT2,@TCSR      ;** ENABLE MAINT. WRAP
2221 004724 005077 175712      CLR      @TBUF          ;LOAD TRANSMIT BUFFER
2222 004730 105777 175704 1$:  TSTB   @TCSR          ;WAIT FOR DONE
2223 004734 100375      BPL      1$
2224 004736 005077 175700      CLR      @TBUF          ;LOAD SECOND BUFFER
2225 004742 000240      NOP
2226 004744 000005      RESET          ;CLEAR DONE WITH RESET
2227 004746 105777 175666  TSTB   @TCSR          ;CHECK FOR DONE SET
2228 004752 100401      BMI      10$          ;** BR TO NEXT TEST IF DONE SET
2229
2230 004754 104005      ERROR  +5.          ;TCSR 'DONE' DOES NOT SET WITH RESET
2231
2232 004756 000240 10$:  NOP          ;**
2233
2234
```

2235

.SBTTL TEST # 6 - TEST ABILITY TO ACCESS RCSR  
:\*\*\*\*\*  
:\*TEST 6 TEST ABILITY TO ACCESS RCSR  
:\*\*\*\*\*

2236 004760 000004  
2236 004762 013703 000004  
2237 004766 012737 005002 000004  
2238 004774 005777 175634  
2239 005000 000402  
2240  
2241 005002 022626  
2242 005004 104006  
2243 005006 010337 000004

TST6: SCOPE  
MOV @#4,R3 ;SAVE TIMEOUT VECTOR  
MOV #1\$,@#4 ;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,@#4 ;RESTORE TIMEOUT VECTOR

2244

```
.SBTTL TEST # 7 - TEST ABILITY TO ACCESS RBUF  
:*****  
:*TEST 7 TEST ABILITY TO ACCESS RBUF  
:*****  
TST7: SCOPE
```

```
2245 005012 000004  
2246 005014 013703 000004  
2247 005020 012737 005034 000004  
2248 005026 005777 175604  
2249 005032 000402  
2250 005034 022626  
2251 005036 104007  
2252 005040 010337 000004
```

```
MOV @#4,R3 ;SAVE TIMEOUT VECTOR  
MOV #1$,@#4 ;SET UP TIMEOUT VECTOR  
TST @RBUF ;ACCESS RBUF  
BR 2$ ;BR TO END OF TEST  
  
1$: CMP (SP)+,(SP)+ ;RESTORE SP AFTER TIMEOUT  
ERROR +7 ;CAN NOT ACCESS RBUF  
2$: MOV R3,@#4 ;RESTORE TIMEOUT VECTOR
```

B 5

2253

.SBTTL TEST # 10 - TEST THAT BIT0(BREAK BIT) CAN BE SET & CLEARED & RESET  
 \*\*\*\*\*  
 \*TEST 10 TEST THAT BIT0(BREAK BIT) CAN BE SET & CLEARED & RESET  
 \*\*\*\*\*  
 TST10: SCOPE

2254	005044	000004				BIT	#BIT8,@SWR		:IS BREAK FUNCTION ENABLED?
2255	005046	032777	000400	173764		BEQ	10\$		** BR TO NEXT TEST, IF NOT
2256	005054	001475				BIT	#BIT0,FLAG44		** IS THIS A 11/44
2257	005056	032737	000001	003002		BEQ	9\$		** NO
2258	005064	001407				TST	CTSTFL		** YES THIS IS 11/44. IS THIS THE CONSOLE
2259	005066	005737	002624						** SLU
2260	005072	001404				BEQ	9\$		** NO
2261	005074	032777	000010	173736		BIT	#BIT03,@SWR		** THIS IS THE CONSOLE SLU.SHOULD THE BREAK
2262									** TEST BE PERFORMED
2263	005102	001462				BEQ	10\$		** NO
2264									
2265	005104	000005			9\$:	RESET			:CLEAR EVERYTHING
2266	005106	032777	000001	175524		BIT	#BIT0,@TCSR		:CHECK BIT0 OF TCSR CLEAR
2267	005114	001411				BEQ	3\$		:BR IF CLEAR
2268	005116	005737	002624			TST	CTSTFL		
2269	005122	001002				BNE	1\$		
2270	005124	104011				ERROR	+11		:BIT0 WAS NOT CLEAR AFTER RESET
2271	005126	000404				BR	3\$		
2272	005130				1\$:				
	005130	004737	015714			JSR	PC,\$ATY4		::ONLY REPORT A FATAL ERROR
	005134	000011				11			::THE ERROR NUMBER (FROM APT LIST)
2273	005136	000000			2\$:	HALT			
2274									
2275	005140	052777	000001	175472	3\$:	BIS	#BIT0,@TCSR		:SET BIT0 IN TCSR
2276	005146	032777	000001	175464		BIT	#BIT0,@TCSR		:TEST BIT0 OF TCSR
2277	005154	001001				BNE	4\$		:BR IF SET
2278									
2279	005156	104012				ERROR	+12		:BIT0 OF TCSR WILL NOT SET
2280									
2281	005160	042777	000001	175452	4\$:	BIC	#BIT0,@TCSR		:CLEAR BIT0 OF TCSR
2282	005166	032777	000001	175444		BIT	#BIT0,@TCSR		:TEST BIT0 OF TCSR
2283	005174	001411				BEQ	7\$		
2284	005176	005737	002624			TST	CTSTFL		
2285	005202	001002				BNE	5\$		
2286	005204	104013				ERROR	+13		:BIT0 OF TCSR WILL NOT CLEAR
2287	005206	000404				BR	7\$		
2288	005210				5\$:				
	005210	004737	015714			JSR	PC,\$ATY4		::ONLY REPORT A FATAL ERROR
	005214	000013				13			::THE ERROR NUMBER (FROM APT LIST)
2289	005216	000000			6\$:	HALT			
2290									
2291	005220	052777	000001	175412	7\$:	BIS	#BIT0,@TCSR		:SET BIT0 IN TCSR
2292	005226	000005				RESET			:CLEAR BIT0 WITH RESET
2293	005230	032777	000001	175402		BIT	#BIT0,@TCSR		:TEST BIT0 CLEAR
2294	005236	001404				BEQ	10\$		** BR IF CLEAR
2295	005240	042777	000001	175372		BIC	#BIT0,@TCSR		:CLEAR BIT0, TO PRINT ERROR
2296	005246	104014				ERROR	+14		:RESET DID NOT CLEAR BIT0 OF TCSR
2297	005250	000240			10\$:	NOP			**

2298

.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  
 \*\*\*\*\*

```

2299 005252 000004
2299 005254 000005
2300 005256 017703 175366
2301 005262 012777 005312 *175360
2302 005270 004737 014502
2303 005274 000340
2304 005276 032777 000100 175334
2305 005304 001404
2306 005306 104021
2307
2308 005310 000402
2309
2310 005312 022626
2311 005314 104022
2312
2313
2314 005316 052777 000100 175314 2$:
2315 005324 032777 000100 175306
2316 005332 001001
2317
2318 005334 104023
2319
2320
2321 005336 042777 000100 175274 3$:
2322 005344 032777 000100 175266
2323 005352 001401
2324 005354 104024
2325
2326
2327 005356 052777 000100 175254 4$:
2328 005364 000005
2329 005366 032777 000100 175244
2330 005374 001401
2331
2332 005376 104025
2333
2334 005400 010377 175244 5$:
    
```

```

TST11: SCOPE
        RESET                ;CLEAR EVERYTHING
        MOV @TVECT,R3        ;SAVE XMIT VECTOR
        MOV #1$,@TVECT      ;SET UP INTERRUPT VECTOR FOR ERROR REPORT
        JSR PC,WRPSW        ;SET PSW TO PRIORITY=7
                                .WORD 340
        BIT #BIT6,@TCSR     ;TEST BIT6 OF TCSR
        BEQ 2$              ;BR IF ZERO
        ERROR +21
                                ;BIT6 IN TCSR NOT CLEAR AFTER RESET
        BR 2$
1$:    CMP (SP)+,(SP)+      ;RESTORE SP AFTER INTERRUPT
        ERROR +22
                                ;XMIT INTERRUPT OCCURRED PRIO=7
2$:    BIS #BIT6,@TCSR     ;SET BIT6 OF TCSR
        BIT #BIT6,@TCSR     ;TEST BIT6 OF TCSR
        BNE 3$             ;BR, IF SET
        ERROR +23
                                ;CANNOT SET BIT6 OF TCSR
3$:    BIC #BIT6,@TCSR     ;CLEAR BIT6 OF TCSR
        BIT #BIT6,@TCSR     ;TEST BIT6 OF TCSR
        BEQ 4$             ;BR IF CLEAR
        ERROR +24
                                ;CANNOT CLEAR BIT6 OF TCSR
4$:    BIS #BIT6,@TCSR     ;SET BIT6 OF TCSR
        RESET              ;CLEAR BIT6 WITH RESET
        BIT #BIT6,@TCSR     ;TEST BIT6 OF TCSR
        BEQ 5$             ;BR IF CLEAR
        ERROR +25
                                ;CANNOT CLEAR BIT6 OF TCSR WITH RESET
5$:    MOV R3,@TVECT      ;RESTORE XMIT VECTOR
    
```

```

2335          .SBTTL TEST # 12 - TEST THAT BIT6 OF RCSR CAN BE SET & RESET
          :*****
          :*TEST 12      TEST THAT BIT6 OF RCSR CAN BE SET & RESET
          :*****
          TST12:  SCOPE
          RESET          ;CLEAR EVERYTHING
          MOV    @RVECT,R3 ;SAVE RECEIVE VECTOR
          MOV    #1$,@RVECT ;SET UP INTERRUPT VECTOR FOR ERROR REPORT
          JSR    PC,WRPSW   ;SET PSW TO PRIORITY=7
          .WORD  340
          BIT    #BIT6,@RCSR ;TEST BIT6 OF RCSR
          BEQ    2$
          ERROR  +26
          ;BIT6 OF RCSR NOT CLEAR AFTER RESET
          BR     2$
          1$:  CMP    (SP)+,(SP)+ ;RESTORE SP AFTER INTERRUPT
          ERROR  +27
          ;RCVR INTERRUPT WITH PRIORITY=7
          2$:  BIS    #BIT6,@RCSR ;SET BIT6 OF RCSR
          BIT    #BIT6,@RCSR ;TEST BIT6 OF RCSR
          BNE    3$
          ;BR, IF SET
          ERROR  +30
          ;CANNOT SET BIT6 OF RCSR
          3$:  BIC    #BIT6,@RCSR ;CLEAR BIT6 OF RCSR
          BIT    #BIT6,@RCSR ;TEST BIT6 OF RCSR
          BEQ    4$
          ;BR, IF CLEAR
          ERROR  +31
          ;CANNOT CLEAR BIT6 OF RCSR
          4$:  BIS    #BIT6,@RCSR ;SET BIT6 OF RCSR
          RESET  ;CLEAR BIT6 OF RCSR WITH RESET
          BIT    #BIT6,@RCSR ;TEST BIT6 OF RCSR
          BEQ    5$
          ;BR, IF CLEAR
          ERROR  +32
          ;CANNOT CLEAR BIT6 OF RCSR WITH RESET
          5$:  MOV    R3,@RVECT ;RESTORE RECEIVE VECTOR
          TCLOCK: MOV   #12,$TSTNM ;ADJUST TEST NUMBER TO (NEXT TEST - 1)
    
```



2374

.SBTTL TEST # 13 - TEST ABILITY TO ACCESS LKS  
 :\*\*\*\*\*  
 :\*TEST 13 TEST ABILITY TO ACCESS LKS  
 :\*\*\*\*\*

2375	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,@SWR ;ARE LINE CLOCK TESTS INHIBITED?
	005562	001014			BNE	TST14 ;IF YES, SKIP THIS TEST
2376	005564	013703	000004		MOV	@#4,R3 ;SAVE TIMEOUT VECTOR
2377	005570	012737	005604	000004	MOV	#1\$,@#4 ;SET UP TIMEOUT VECTOR
2378	005576	005777	175072		TST	@LKS ;ACCESS LKS
2379	005602	000402			BR	2\$ ;NO TIMEOUT - BR TO END OF TEST
2380						
2381	005604	022626			1\$: CMP	(SP)+,(SP)+ ;RESTORE SP AFTER TIMEOUT
2382	005606	104010			ERROR	+10 ;CAN NOT ACCESS LKS
2383						
2384	005610	010337	000004		2\$: MOV	R3,@#4 ;RESTORE TIMEOUT VECTOR
2385						

2386

.SBTTL TEST # 14 - TEST THAT BIT6 OF LKS CAN BE SET & RESET  
 :\*\*\*\*\*  
 :\*TEST 14 TEST THAT BIT6 OF LKS CAN BE SET & RESET  
 :\*\*\*\*\*

2387	005614	000004				TST14: SCOPE		
	005616	005737	002624			TST	CTSTFL	; IS CONSOLE UNDER TEST?
	005622	001460				BEQ	TST15	; IF NOT, SKIP THIS TEST
	005624	032777	000100	173206		BIT	#BIT6,@SWR	; ARE LINE CLOCK TESTS INHIBITED?
	005632	001054				BNE	TST15	; IF YES, SKIP THIS TEST
2388	005634	000005				RESET		
2389	005636	017703	175034			MOV	@RTCVT,R3	; SAVE LINE CLOCK VECTOR
2390	005642	012777	005672	175026		MOV	#1\$,@RTCVT	; SET UP INTERRUPT VECTOR FOR ERROR REPORT
2391	005650	004737	014502			JSR	PC,WRPSW	; SET PSW TO PRIORITY 7
2392	005654	000340					.WORD 340	
2393	005656	032777	000100	175010		BIT	#BIT6,@LKS	; TEST BIT6 OF LKS
2394	005664	001404				BEQ	2\$	
2395	005666	104033				ERROR	+33	
2396								; BIT6 OF LKS NOT CLEAR AFTER RESET
2397	005670	000402				BR	2\$	
2398								
2399	005672	022626			1\$:	CMR	(SP)+,(SP)+	; RESTORE SP AFTER INTERRUPT
2400	005674	104034				ERROR	+34	
2401								; LKS INTERRUPT WITH PRIORITY=7
2402								
2403	005676	052777	000100	174770	2\$:	BIS	#BIT6,@LKS	; SET BIT6 OF LKS
2404	005704	032777	000100	174762		BIT	#BIT6,@LKS	; TEST BIT6 OF LKS
2405	005712	001001				BNE	3\$	; BR IF SET
2406								
2407	005714	104035				ERROR	+35	
2408								; CANNOT SET BIT6 OF LKS
2409								
2410	005716	042777	000100	174750	3\$:	BIC	#BIT6,@LKS	; CLEAR BIT6 OF LKS
2411	005724	032777	000100	174742		BIT	#BIT6,@LKS	; TEST BIT6 OF LK
2412	005732	001401				BEQ	4\$	
2413	005734	104036				ERROR	+36	
2414								; CANNOT CLEAR BIT6 OF LKS
2415	005736	052777	000100	174730	4\$:	BIS	#BIT6,@LKS	; SET BIT6 OF LKS
2416	005744	000005				RESET		; CLEAR BIT6 OF LKS WITH RESET
2417	005746	032777	000100	174720		BIT	#BIT6,@LKS	; TEST BIT6 OF LKS
2418	005754	001401				BEQ	5\$	; BR IF CLEAR
2419								
2420	005756	104037				ERROR	+37	
2421								; CANNOT CLEAR BIT6 OF LKS WITH RESET
2422	005760	010377	174712		5\$:	MOV	R3,@RTCVT	; RESTORE LINE CLOCK VECTOR

2423

.SBTTL TEST # 15 - TEST FOR DUAL ADDRESSING OF REGISTERS

\*\*\*\*\*  
 :\*TEST 15 TEST FOR DUAL ADDRESSING OF REGISTERS  
 \*\*\*\*\*

2424	005764	000004				TST15: SCOPE		
2424	005766	013703	000004			MOV @#4,R3		;SAVE TIMEOUT VECTOR
2425	005772	013704	000006			MOV @#6,R4		;SAVE TIMEOUT PSW
2426	005776	012737	006130	000004		MOV #4\$,@#4		;SET UP TIMEOUT VECTOR
2427	006004	012737	000340	000006		MOV #340,@#6		;KEEP PRIO=7
2428	006012	000005				RESET		;CLEAR EVERYTHING
2429	006014	012700	000002			MOV #BIT1,R0		;SET UP BIT MASK
2430	006020	032777	000020	173012		BIT #BIT4,@SWR		;CLOCK TEST ONLY?
2431	006026	001404				BEQ 1\$		;BR IF NOT
2432	006030	013737	002674	001020		MOV LKS,\$GDADR		;ELSE, MOVE GOOD LKS ADDRESS INTO \$GDADR
2433	006036	000403				BR 2\$		
2434	006040	013737	002634	001020	1\$:	MOV RCSR,\$GDADR		;MOVE GOOD RCSR ADDRESS INTO \$GDADR
2435	006046	013737	001020	001022	2\$:	MOV \$GDADR,\$BDADR		;MOVE GOOD ADDRESS INTO TEST ADDRESS LOCATION
2436	006054	040037	001022			BIC R0,\$BDADR		;CREATE BAD ADDRESS BY COMPLEMENTING ONE BIT
2437	006060	023737	001020	001022		CMP \$GDADR,\$BDADR		;ARE ADDRESSES IDENTICAL?
2438	006066	001002				BNE 3\$		;IF NOT, TEST THIS ADDRESS
2439	006070	050037	001022			BIS R0,\$BDADR		;ELSE, BIT SET THIS BIT POSITION TO GENERATE BAD ADDRESS
2440	006074	017737	172722	001024	3\$:	MOV @ \$BDADR,\$GDDAT		;SAVE CONTENTS OF BAD ADDRESS IF IT EXISTS
2441	006102	052777	000100	172712		BIS #BIT6,@ \$BDADR		;SET BIT6 USING BAD ADDRESS
2442	006110	032777	000100	172702		BIT #BIT6,@ \$GDADR		;CHECK TO SEE IF GOOD ADDRESS CONTAINS BIT6
2443	006116	001011				BNE 6\$		;BR IF SET ---> ERROR
2444	006120	013777	001024	172674		MOV \$GDDAT,@ \$BDADR		;RESTORE ANY MEMORY LOCATION THAT WAS ALTERED
2445	006126	000401				BR 5\$		;BR TO CONTINUE TEST
2446	006130	022626			4\$:	CMP (SP)+,(SP)+		;RESTORE SP AFTER TIMEOUT
2447	006132	006300			5\$:	ASL R0		;SHIFT BIT MASK TO NEXT POSITION
2448	006134	105700				TSTB R0		;COMPLEMENTED ALL BITS FROM 1 - 7?
2449	006136	100343				BPL 2\$		;BR, IF NOT.
2450	006140	000401				BR 7\$		;BR TO NEXT TEST
2451								
2452	006142	104040			6\$:	ERROR +40		;DUAL ADDRESSING ERROR
2453								; \$BDADR = DUAL ADDRESS
2454								; \$GDADR = GOOD ADDRESS
2455								
2456	006144	010337	000004		7\$:	MOV R3,@#4		;RESTORE TIMEOUT VECTOR
2457	006150	010437	000006			MOV R4,@#6		;RESTORE TIMEOUT PSW

::++  
 ::++

2458

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

2459	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,@SWR		: ARE LINE CLOCK TESTS INHIBITED?
	006172	001033				BNE TST17		: IF YES, SKIP THIS TEST
2460	006174	000005				RESET		: CLEAR EVERYTHING & SET BIT7 OF LKS
2461	006176	105777	174472		1\$:	TSTB @LKS		: TEST FOR BIT7 OF LKS
2462	006202	100401				BMI 2\$		: BR IF SET
2463								
2464	006204	104041				ERROR +41		: BIT7 OF LKS DID NOT SET WITH RESET
2465								
2466	006206	042777	000200	174460	2\$:	BIC #BIT7,@LKS		: CLEAR BIT7 OF LKS
2467	006214	032777	000200	174452		BIT #BIT7,@LKS		: TEST BIT7 OF LKS
2468	006222	001410				BEQ 3\$		
2469	006224	042777	000200	174442		BIC #BIT7,@LKS		: TRY ONE MORE TIME BECAUSE THE CLOCK
2470	006232	032777	000200	174434		BIT #BIT7,@LKS		: MAY HAVE SET IMMEDIATELY AFTER THE FIRST CLEAR
2471	006240	001401				BEQ 3\$		
2472								
2473	006242	104042				ERROR +42		: CAN NOT CLEAR BIT7 OF LKS
2474								
2475	006244	005000			3\$:	CLR R0		: CLEAR TIMER
2476	006246	105777	174422		(CONT:	TSTB @LKS		: TEST FOR BIT7 OF LKS
2477	006252	100403				BMI TST17		: BR, IF SET
2478	006254	005200				INC R0		: INCREMENT TIMER
2479	006256	001373				BNE CONT		: CONTINUE UNTIL TIME EXPIRES
2480								
2481	006260	104043				ERROR +43		: BIT7 OF LKS DOES NOT SET



2530

.SBTTL TEST # 20 - TEST RTC FOR DOUBLE INTERRUPTS

\*\*\*\*\*  
 :\*TEST 20 TEST RTC FOR DOUBLE INTERRUPTS  
 \*\*\*\*\*

```

TST20: SCOPE
2531 006522 000004
006524 005737 002624
006530 001473
006532 032777 000100 172300
006540 001067
2532 006542 000005
2533 006544 017703 174126
2534 006550 017704 174124
2535 006554 012777 006646 174114
2536 006562 012777 000340 174110
2537 006570 004737 014502
2538 006574 000240
2539 006576 005077 174072
2540 006602 012777 000100 174064
2541 006610 105777 174060 1$:
2542 006614 100375
2543 006616 005077 174052
2544 006622 012777 000100 174044
2545 006630 105777 174040 2$:
2546 006634 100375
2547 006636 000240
2548 006640 000240
2549 006642 104047
2550 006644 000401
2551 006646 022626 3$:
2552 006650 012777 006676 174020 4$:
2553 006656 004737 014502
2554 006662 000240
2555 006664 000240
2556 006666 000240
2557 006670 000240
2558 006672 000240
2559 006674 000402
2560
2561 006676 022626 5$:
2562 006700 104050
2563
2564
2565 006702 042777 000100 173764 6$:
2566 006710 010377 173762
2567 006714 010477 173760
    
```

2568

.SBTTL TEST # 21 - TEST THAT RTC INTERRUPT CLEARS WITH RESET  
 :\*\*\*\*\*  
 :\*TEST 21 TEST THAT RTC INTERRUPT CLEARS WITH RESET  
 :\*\*\*\*\*

2569	006720	000004			TST21: SCOPE		
	006722	005737	002624		TST	CTSTFL	:IS CONSOLE UNDER TEST?
	006726	001452			BEL	TST22	:IF NOT, SKIP THIS TEST
	006730	032777	000100	172102	BIT	#BIT6,@SWR	:ARE LINE CLOCK TESTS INHIBITED?
	006736	001046			BNE	TST22	:IF YES, SKIP THIS TEST
2570	006740	004737	014502		JSR	PC,WRPSW	:SET PRIORITY TO 7
2571	006744	000340				.WORD 340	
2572	006746	017703	173724		MOV	@RTCVT,R3	:SAVE LINE CLOCK VECTOR
2573	006752	012777	007044	173716	MOV	#3\$,@RTCVT	:POINT RTC VECTOR TO ERROR REPORT
2574	006760	005077	173710		CLR	@LKS	:CLEAR DONE FLAG AND
2575	006764	012777	000100	173702	MOV	#BIT6,@LKS	:SET CLOCK INTERRUPT ENABLE
2576	006772	105777	173676		1\$: TSTB	@LKS	:WAIT FOR DONE (INTERRUPT REQUEST)
2577	006776	100375			BPL	1\$	
2578	007000	005077	173670		CLR	@LKS	:CLEAR DONE FLAG AND
2579	007004	012777	000100	173662	MOV	#BIT6,@LKS	:SET CLOCK INTERRUPT ENABLE
2580	007012	105777	173656		2\$: TSTB	@LKS	:WAIT FOR DONE (INTERRUPT REQUEST)
2581	007016	100375			BPL	2\$	
2582	007020	000240			NOP		:GIVE TIME FOR ANY INTERRUPT
2583	007022	000005			RESET		:CLEAR PENDING INTERRUPT WITH RESET
2584	007024	004737	014502		JSR	PC,WRPSW	:SET PRIORITY TO 5
2585	007030	000240				.WORD 240	
2586	007032	000240			NOP		:GIVE TIME FOR ANY INTERRUPT
2587	007034	042777	000100	173632	BIC	#BIT6,@LKS	:DISALLOW INTERRUPTS
2588	007042	000402			BR	4\$	:BR TO END OF TEST
2589							
2590	007044	022626			3\$: CMP	(SP)+,(SP)+	:RESTORE SP AFTER INTERRUPT
2591	007046	104051			ERROR	+51	:RESET DID NOT CLEAR INTERRUPT
2592							
2593	007050	010377	173622		4\$: MOV	R3,@RTCVT	:RESTORE LINE CLOCK VECTOR

```

2594          .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
              :*****
2595 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,@SWR      :ARE LINE CLOCK TESTS INHIBITED?
007072 001053          BNE          TST23          :IF YES, SKIP THIS TEST
2596 007074 004737 014502 JSR          PC,WRPSW      :SET PRIORITY TO 7
2597 007100 000340          .WORD          340
2598 007102 017703 173570 MOV          @RTCVT,R3     :SAVE LINE CLOCK VECTOR
2599 007106 012777 007204 173562 MOV          #3,@RTCVT    :POINT RTC VECTOR TO ERROR REPORT
2600 007114 005077 173554 CLR          @LKS         :CLEAR DONE FLAG AND
2601 007120 012777 000100 173546 MOV          #BIT6,@LKS   :SET CLOCK INTERRUPT ENABLE
2602 007126 105777 173542 1$: TSTB         @LKS         :WAIT FOR DONE (INTERRUPT REQUEST)
2603 007132 100375          BPL          1$
2604 007134 005077 173534 CLR          @LKS         :CLEAR DONE FLAG AND
2605 007140 012777 000100 173526 MOV          #BIT6,@LKS   :SET CLOCK INTERRUPT ENABLE
2606 007146 105777 173522 2$: TSTB         @LKS         :WAIT FOR DONE (INTERRUPT REQUEST)
2607 007152 100375          BPL          2$
2608 007154 000240          NOP
2609 007156 042777 000200 173510 BIC          #BIT7,@LKS   :GIVE TIME FOR ANY INTERRUPT
2610 007164 004737 014502 JSR          PC,WRPSW      :CLEAR DONE & INTERRUPT
2611 007170 000240          .WORD          240      :ALLOW INTERRUPTS
2612 007172 000240          NOP
2613 007174 042777 000100 173472 BIC          #BIT6,@LKS   :GIVE TIME FOR ANY INTERRUPT
2614 007202 000402          BR           4$         :DISALLOW INTERRUPTS
2615          :BR TO END OF TEST
2616
2617 007204 022626          3$: CMP          (SP)+,(SP)+ :RESTORE SP AFTER INTERRUPT
2618 007206 104052          ERROR        +52      :CLEARING BIT7 OF LKS DID NOT CLEAR INTERRUPT
2619
2620 007210 010377 173462          4$: MOV          R3,@RTCVT   :RESTORE LINE CLOCK VECTOR
2621 007214 004737 014502 JSR          PC,WRPSW      :RESTORE PRIORITY TO 7
2622 007220 000340          .WORD          340
    
```



2623

.SBTTL TEST # 23 - TEST CLOCK REPEATABILITY  
:\*\*\*\*\*  
:\*TEST 23 TEST CLOCK REPEATABILITY  
:\*\*\*\*\*

2624	007222	000004				TST23: SCOPE		
	007224	005737	002624			TST	CTSTFL	: IS CONSOLE UNDER TEST?
	007230	001467				BEQ	TST24	: IF NOT, SKIP THIS TEST
	007232	032777	000100	171600		BIT	#BIT6,@SWR	: ARE LINE CLOCK TESTS INHIBITED?
	007240	001063				BNE	TST24	: IF YES, SKIP THIS TEST
2625	007242	042777	000100	173424		BIC	#BIT6,@LKS	: DISALLOW INTERRUPTS
2626								
2627	007250	005000				CLR	R0	: CLEAR A TIMER
2628	007252	012701	177777			MOV	#-1,R1	: SET A FLAG INDICATING FIRST PASS THRU THIS LOOP
2629	007256	005002			1\$:	CLR	R2	: CLEAR CLOCK COUNTER
2630	007260	005077	173410			CLR	@LKS	: CLEAR DONE
2631	007264	105777	173404		2\$:	TSTB	@LKS	: SYNC ON DONE
2632	007270	100375				BPL	2\$	
2633	007272	005077	173376			CLR	@LKS	: CLEAR DONE
2634	007276	105777	173372		3\$:	TSTB	@LKS	: IS CLOCK DONE?
2635	007302	100003				BPL	4\$	: BR IF NOT, TO INCREMENT TIMER
2636	007304	005202				INC	R2	: IF DONE, INCREMENT CLOCK COUNT
2637	007306	005077	173362			CLR	@LKS	: CLEAR DONE
2638	007312	005200			4\$:	INC	R0	: INCREMENT TIMER
2639	007314	001370				BNE	3\$	: BR IF TIME REMAINS
2640	007316	005201				INC	R1	: INCREMENT LOOP PASS FLAG
2641	007320	001003				BNE	CMPARE	: BR IF TWO PASSES HAVE BEEN MADE
2642	007322	010237	002336			MOV	R2,FIRST	: IF NOT, STORE FIRST CLOCK COUNT
2643	007326	000753				BR	1\$	: DO LOOP AGAIN
2644	007330	013701	002336		CMPARE:	MOV	FIRST,R1	: RECALL FIRST CLOCK COUNT
2645	007334	160201				SUB	R2,R1	: CALCULATE DIFFERENCE OF TWO COUNTS
2646	007336	100001				BPL	TOLER	: IF POSITIVE, SKIP NEGATION OF DIFFERENCE
2647	007340	005401				NEG	R1	: MAKE DIFFERENCE A POSITIVE NUMBER
2648	007342	032737	000001	003002	TOLER:	BIT	#BIT0,FLAG44	: ** IS THIS A 11/44
2649	007350	001403				BEQ	6\$	: ** NO
2650	007352	020127	000002			CMP	R1,#2	: ** YES; COMPARE DIFFERENCE WITH DESIRED
2651								: ** TOLERANCE OF 2
2652	007356	000402				BR	7\$	: **
2653	007360	020127	000001		6\$:	CMP	R1,#1	: COMPARE DIFFERENCE WITH DESIRED TOLERANCE
2654	007364	003403			7\$:	BLE	5\$	: BR, IF LOWER/EQUAL TO TOLERANCE
2655								
2656	007366	010237	002616			MOV	R2,SECND	: STORE SECOND COUNT
2657	007372	104053				ERROR	+53	: CLOCK REPEATABILITY ERROR
2658								
2659	007374	032777	000020	171436	5\$:	BIT	#BIT4,@SWR	: CLOCK TESTS ONLY?
2660	007402	001402				BEQ	TST24	: BR IF NOT
2661	007404	000137	014326			JMP	\$EOP	: ELSE, JUMP TO END OF PASS ROUTINE

2662

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

2663	007410	000004							
2664	007412	042777	000100	173220					
2665	007420	017703	173224						
2666	007424	012777	007450	173216					
2667	007432	105777	173202		1\$:				
2668	007436	100375							
2669	007440	004737	014502						
2670	007444	000140							
2671	007446	000402							
2672	007450	022626			2\$:				
2673	007452	104054							
2674									
2675	007454	012777	007504	173166	3\$:				
2676	007462	052777	000100	173150					
2677	007470	000240							
2678	007472	000240							
2679	007474	000240							
2680	007476	000240							
2681	007500	104055							
2682	007502	000401							
2683	007504	022626			4\$:				
2684	007506	042777	000100	173124	5\$:				
2685	007514	010377	173130						

```

TST24: SCOPE
        BIC    #BIT6,@TCSR    ;CLEAR TRANSMIT INTERRUPT ENABLE
        MOV    @TVECT,R3     ;SAVE XMIT VECTOR
        MOV    #2$,@TVECT    ;POINT XMIT VECTOR TO ERROR REPORT
1$:     TSTB   @TCSR         ;WAIT FOR DONE
        BPL    1$
        JSR    PC,WRPSW      ;SET PSW TO PRIORITY 3
        .WORD  140
        BR    3$

2$:     CMP    (SP)+,(SP)+    ;RESTORE SP AFTER INTERRUPT
        ERROR  +54

3$:     MOV    #4$,@TVECT    ;XMIT INTERRUPTS WITH INTERRUPT ENABLE CLEAR
        BIS    #BIT6,@TCSR   ;SET XMIT VECTOR TO END OF TEST
        NOP                                ;ENABLE INTERRUPTS
        NOP                                ;ALLOW
        NOP                                ;TIME
        NOP                                ;FOR
        NOP                                ;INTERRUPT
        ERROR  +55            ;XMIT DID NOT INTERRUPT
        BR    5$             ;BRANCH OVER STACK CORRECTION
4$:     CMP    (SP)+,(SP)+    ;RESTORE SP AFTER INTERRUPT
5$:     BIC    #BIT6,@TCSR   ;DISABLE INTERRUPTS
        MOV    R3,@TVECT    ;RESTORE XMIT VECTOR
    
```

2686

.SBTTL TEST # 25 - TEST THAT XMIT INTERRUPTS DO NOT OCCUR WHEN DISABLED  
 :\*\*\*\*\*  
 :\*TEST 25 TEST THAT XMIT INTERRUPTS DO NOT OCCUR WHEN DISABLED  
 :\*\*\*\*\*  
 TST25: SCOPE

```

2687 007520 000004
2687 007522 042777 000100 173110      BIC    #BIT6,@TCSR    ;DISABLE INTERRUPTS
2688 007530 004737 014502              JSR    PC,WRPSW      ;SET PSW TO PRIORITY 7
2689 007534 000340                      .WORD  340
2690 007536 017703 173106              MOV    @TVECT,R3    ;SAVE XMIT VECTOR
2691 007542 012777 007576 173100      MOV    #2$,@TVECT   ;POINT XMIT VECTOR TO ERROR REPORT
2692 007550 105777 173064      1$:   TSTB   @TCSR      ;WAIT FOR DONE
2693 007554 100375                      BPL    1$
2694 007556 052777 000100 173054      BIS    #BIT6,@TCSR  ;ENABLE INTERRUPT
2695 007564 000240                      NOP
2696 007566 000240                      NOP    ;ALLOW
2697 007570 000240                      NOP    ;TIME
2698 007572 000240                      NOP    ;FOR
2699 007574 000402                      NOP    ;INTERRUPT
2700                                BR     3$           ;CONTINUE TEST
2701 007576 022626              2$:   CMP    (SP)+,(SP)+ ;RESTORE SP AFTER INTERRUPT
2702 007600 104056                      ERROR  +56
2703                                ;XMIT INTERRUPTS AT PRIORITY=7
2704 007602 042777 000100 173030      3$:   BIC    #BIT6,@TCSR ;CLEAR INTERRUPT ENABLE
2705 007610 012777 007636 173032      MOV    #4$,@TVECT   ;POINT XMIT VECTOR TO ERROR REPORT
2706 007616 004737 014502              JSR    PC,WRPSW      ;SET PSW TO PRIORITY 3
2707 007622 000140                      .WORD  140
2708 007624 000240                      NOP
2709 007626 000240                      NOP    ;ALLOW
2710 007630 000240                      NOP    ;TIME
2711 007632 000240                      NOP    ;FOR
2712 007634 000402                      NOP    ;INTERRUPT
2713                                BR     5$           ;BR TO END OF TEST-NO INTERRUPT
2714 007636 022626              4$:   CMP    (SP)+,(SP)+ ;RESTORE SP AFTER INTERRUPT
2715 007640 104057                      ERROR  +57
2716                                ;XMIT INTERRUPT OCCURES WITH BIT6 CLEAR
2717 007642 010377 173002      5$:   MOV    R3,@TVECT  ;RESTORE XMIT VECTOR
    
```

2718

.SBTTL TEST # 26 - TEST TRANSMITTER FOR DOUBLE INTERRUPTS  
 :\*\*\*\*\*  
 :\*TEST 26 TEST TRANSMITTER FOR DOUBLE INTERRUPTS  
 :\*\*\*\*\*  
 TST26: SCOPE

```

2719 007646 000004          BIC      #BIT6,@TCSR      ;CLEAR INTERRUPT ENABLE
2720 007650 042777 000100 172762  MOV      @TVECT,R3        ;SAVE XMIT VECTOR
2721 007656 017703 172766          MOV      @TPSW,R4        ;SAVE XMIT PSW VECTOR
2722 007662 017704 172764          MOV      #2$,@TVECT      ;SET UP XMIT VECTOR
2723 007666 012777 007740 172754  MOV      #340,@TPSW      ;SET PIO 7 AFTER INTERRUPT
2724 007674 012777 000340 172750  MOV      PC,WRPSW        ;SET PSW TO PRIORITY 3
2725 007702 004737 014502          JSR      PC,WRPSW        ;SET PSW TO PRIORITY 3
2726 007706 000140          .WORD   140
2727 007710 105777 172724          1$:    TSTB   @TCSR        ;WAIT FOR DONE
2728 007714 100375          BPL     1$
2729 007716 052777 000100 172714  BIS     #BIT6,@TCSR      ;ENABLE INTERRUPTS
2730 007724 000240          NOP
2731 007726 000240          NOP      ;ALLOW
2732 007730 000240          NOP      ;TIME
2733 007732 000240          NOP      ;FOR
2734 007734 104060          ERROR  +60              ;INTERRUPT
2735 007736 000401          BR     25$              ;XMIT INTERRUPT DID NOT OCCUR
2736 007740 022626          2$:    CMP     (SP)+,(SP)+  ;BRANCH OVER STACK CORRECTION INTERRUPT
2737 007742 012777 007776 172700  25$:   MOV     #4$,@TVECT    ;RESTORE SP AFTER INTERRUPT
2738 007750 004737 014502          JSR     PC,WRPSW        ;POINT XMIT VECTOR TO ERROR
2739 007754 000140          .WORD  140              ;SET PSW TO PRIORITY 3
2740 007756 000240          NOP      ;ALLOW
2741 007760 000240          NOP      ;TIME
2742 007762 000240          NOP      ;FOR
2743 007764 000240          NOP      ;INTERRUPT
2744 007766 042777 000100 172644  BIC     #BIT6,@TCSR      ;DISABLE INTERRUPTS
2745 007774 000402          BR     5$                ;BR TO END OF TEST
2746
2747 007776 022626          4$:    CMP     (SP)+,(SP)+  ;RESTORE SP AFTER INTERRUPT
2748 010000 104061          ERROR  +61
2749
2750 010002 010377 172642          5$:    MOV     R3,@TVECT    ;XMIT RE-INTERRUPTED
2751 010006 010477 172640          MOV     R4,@TPSW        ;RESTORE XMIT VECTOR
2752

```

2753

.SBTTL TEST # 27 - TEST THAT XMIT INTERRUPT CLEARS WITH LOADING TBUF  
 :\*\*\*\*\*  
 :\*TEST 27 TEST THAT XMIT INTERRUPT CLEARS WITH LOADING TBUF  
 :\*\*\*\*\*  
 TST27: SCOPE

```

2754 010012 000004          BIC    #BIT6,@TCSR    ;DISABLE INTERRUPTS
2755 010014 042777 000100 172616 JSR    PC,WRPSW      ;SET PSW TO PRIORITY 7
2756 010022 004737 014502          .WORD  340
2757 010026 000340          MOV    @TVECT,R3     ;SAVE XMIT VECTOR
2758 010030 017703 172614          MOV    #2$,@TVECT   ;POINT XMIT VECTOR TO ERROR
2759 010034 012777 010122 172606 BIS    #BIT2,@TCSR   ;** ENABLE MAINT. WRAP
2760 010042 052777 000004 172570 BIS    #BIT6,@TCSR   ;ENABLE INTERRUPTS
2761 010050 052777 000100 172562 CLR    @TBUF         ;LOAD TBUF
2762 010056 005077 172560          CLR    @TCSR        ;WAIT FOR DONE (INTERRUPT)
2763 010062 105777 172552 1$:  TSTB  @TCSR
2764 010066 100375          BPL    1$
2765 010070 005077 172546          CLR    @TBUF        ;FILL SECOND BUFFER TO RESET INT.
2766 010074 004737 014502          JSR    PC,WRPSW     ;ALLOW INTERRUPTS
2767 010100 000140          .WORD  140
2768 010102 000240          NOP
2769 010104 000240          NOP
2770 010106 000240          NOP
2771 010110 000240          NOP
2772 010112 042777 000100 172520 BIC    #BIT6,@TCSR   ;DISABLE INTERRUPTS
2773 010120 000405          BR     3$           ;BR TO END OF TEST
2774 010122 022626          CMP    (SP)+,(SP)+  ;RESTORE SP AFTER INTERRUPT
2775 010124 042777 000004 172526 BIC    #BIT2,@CTCSR  ;** DISABLE MAINTENANCE MODE FOR
; ** CONSOLE TO ALLOW FOR COMMUNICATION
; ** WITH TERMINAL.
2776 010132 104062          ERROR +62
2777
2778 010134 010377 172510 3$:  MOV    R3,@TVECT    ;LOADING TBUF DID NOT CLEAR INTERRUPT.
2779
2780 010140 005000          CLR    R0           ;RESTORE XMIT VECTOR
010142 012701 000002          MOV    #2,R1
010146 005300 40$:  DEC    R0
010150 001376          BNE   40$
010152 005301          DEC    R1
010154 001374          BNE   40$
; ** 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.
    
```

```

2781          .SBTTL TEST # 30 - TEST THAT RCVR ACTIVE & DONE SET & CLEAR
                :*****
                :*TEST 30          TEST THAT RCVR ACTIVE & DONE SET & CLEAR
                :*****
                TST30: SCOPE
2782 010156 000004          BIT      #BIT0,FLAG44      ;**
2783 010160 032737 000001 003002          BNE      RCVDON          ;**
2784 010170 000005          RESET          ;CLEAR EVERYTHING
2785 010172 052777 000004 172440          BIS      #BIT2,@TCSR     ;SET MAINTENANCE WRAP
2786 010200 005000          CLR      R0              ;** DELAY ENOUGH TIME TO ALLOW ANY CHARACTERS
                010202 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
                ;**
                010206 105777 172424          42$: TSTB   @RBUF          ;**
                010212 005300          DEC      R0              ;**
                010214 001374          BNE      42$            ;**
                010216 005301          DEC      R1              ;**
                010220 001372          BNE      42$            ;**
2787 010222 005000          CLR      R0              ;CLEAR A TIMER
2788 010224 005077 172412          CLR      @TBUF          ;LOAD TRANSMIT BUFFER
2789 010230 032777 004000 172376 WACTV: BIT      #BIT11,@RCSR     ;TEST RCVR ACTIVE BIT
2790 010236 001006          BNE      2$              ;BR IF SET
2791 010240 005200          INC      R0              ;INCREMENT TIMER IF NOT SET
2792 010242 001372          BNE      WACTV          ;CONTINUE WAIT IF TIME REMAINS
2793 010244 042777 000004 172406          BIC      #BIT2,@CTCSR   ;** DISABLE MAINTENANCE MODE FOR
                ;** CONSOLE TO ALLOW FOR COMMUNICATION
                ;** WITH TERMINAL.
                ;**
2794          ERROR      +63          ;RCVR ACTIVE DID NOT SET WHILE RECEIVING
2795 010252 104063
2796
2797 010254          2$: CLR      R0              ;** DELAY ENOUGH TIME TO ALLOW ANY CHARACTERS
                010254 005000          MOV      #2,R1          ;** THAT MIGHT BE IN THE PROCESS OF BEING
                010256 012701 000002          ;** TRANSMITTED TO FINISH BEFORE MAINTENANCE
                010262 005300          40$: DEC      R0              ;** WRAP FOR THE UART UNDER TEST IS DISABLED.
                010264 001376          BNE      40$            ;** THIS WILL INHIBIT ANY COMMUNICATION
                010266 005301          DEC      R1              ;** TO HARDWARE MEDIA SUCH AS TU58 THAT MIGHT
                010270 001374          BNE      40$            ;** BE ATTACHED TO UART UNDER TEST.
                ;**
2798 010272 000005          RESET          ;VERIFY "INIT" CLEARS RCV ACTIVE
2799 010274 032777 004000 172332          BIT      #BIT11,@RCSR
2800 010302 001401          BEQ      3$              ;INIT DID NOT CLEAR RCV ACTIVE
2801
2802 010304 104115          ERROR      +115
2803
2804 010306 005000          3$: CLR      R0              ;CLEAR A TIMER
2805 010310 052777 000004 172322          BIS      #BIT2,@TCSR     ;SET MAINTENANCE WRAP
2806 010316 062700 000000          WT: ADD      #0,R0          ;WAIT AT LEAST ONE BIT TIME
2807 010322 005200          INC      R0
2808 010324 001374          BNE      WT
2809 010326 033777 004000 172300          BIT      BIT11,@RCSR     ;VERIFY RCV ACTIVE STILL CLEAR
2810 010334 001404          BEQ      RCVDON         ;BR IF CLEAR
2811
2812 010336 042777 000004 172314          BIC      #BIT2,@CTCSR   ;** DISABLE MAINTENANCE MODE FOR
                ;** CONSOLE TO ALLOW FOR COMMUNICATION
                ;** WITH TERMINAL.
                ;**
2813 010344 104116          ERROR      +116          ;RCV ACTIVE WITHOUT "START" BIT
2814
    
```

```

2815 010346 052777 000004 172264 RCVDON: BIS #BIT2,@TCSR ;SET MAINTENCE WRAP
2816 010354 005000 CLR R0 ;CLEAR TIMER
2817 010356 005077 172260 CLR @TBUF ;LOAD TRANSMIT BUFFER
2818 010362 105777 172246 WDONE: TSTB @RCSR ;CHECK FOR RECEIVER DONE
2819 010366 100406 BMI 5$ ;BR, IF DONE
2820 010370 005200 INC R0 ;INCREMENT TIMER, IF NOT DONE
2821 010372 001373 BNE WDONE ;CONTINUE WAIT IF TIME REMAINS
2822 010374 042777 000004 172256 BIC #BIT2,@CTCSR ;** DISABLE MAINTENANCE MODE FOR
; ** CONSOLE TO ALLOW FOR COMMUNICATION
; ** WITH TERMINAL.

2823 010402 104064 ERROR +64 ;RECEIVER DONE NEVER SET
2824
2825
2826 010404 032737 000001 003002 5$: BIT #BIT0,FLAG44 ;** 11/44??
2827 010412 001010 BNE 6$ ;** DO NOT EXECUTE THIS SECTION
2828 010414 032777 004000 172212 BIT #BIT11,@RCSR ;CHECK FOR RCVR ACTIVE CLEAR
2829 010422 001404 BEQ 6$ ;BR, IF CLEAR
2830 010424 042777 000004 172226 BIC #BIT2,@CTCSR ;** DISABLE MAINTENANCE MODE FOR
; ** CONSOLE TO ALLOW FOR COMMUNICATION
; ** WITH TERMINAL.

2831 010432 104065 ERROR +65 ;RCVR ACTIVE DID NOT CLEAR WITH RCVR DONE
2832
2833
2834 010434 000005 6$: RESET ;CLEAR DONE WITH RESET
2835 010436 105777 172172 TSTB @RCSR ;CHECK FOR DONE CLEAR
2836 010442 001404 BEQ 7$
2837
2838 010444 042777 000004 172206 BIC #BIT2,@CTCSR ;** DISABLE MAINTENANCE MODE FOR
; ** CONSOLE TO ALLOW FOR COMMUNICATION
; ** WITH TERMINAL.

2839 010452 104066 ERROR +66 ;RESET DID NOT CLEAR RCVR DONE
2840
2841
2842 010454 7$:
2843 010454 042777 000004 172176 BIC #BIT2,@CTCSR ;** DISABLE MAINTENANCE MODE FOR
; ** CONSOLE TO ALLOW FOR COMMUNICATION
; ** WITH TERMINAL.

```

2844

```
.SBTTL TEST # 31 - TEST THAT READING RBUF CLEARS RECEIVER DONE
:*****
:*TEST 31 TEST THAT READING RBUF CLEARS RECEIVER DONE
:*****
```

```
TST31: SCOPE
        RESET
        BIS #BIT2,@TCSR ;CLEAR EVERYTHING
        CLR R0 ;SET MAINTENANCE WRAP
        MOV #2,R1 ;** 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

        42$: TSTB @RBUF
        DEC R0
        BNE 42$
        DEC R1
        BNE 42$
        CLR @TBUF ;LOAD TRANSMITTER
        1$: TSTB @RCSR ;WAIT FOR RECEIVER DONE
        BPL 1$
        MOV @RBUF,R0 ;READ RECEIVE BUFFER
        BIC #BIT2,@TCSR ;** DISABLE MAINTENANCE MODE FOR
        ;** CONSOLE TO ALLOW FOR COMMUNICATION
        ;** WITH TERMINAL.
        TSTB @RCSR ;CHECK FOR RECEIVE DONE CLEAR
        BEQ 10$ ;** BR, IF CLEAR TO NEXT TEST
        ERROR +70

        10$: NOP ;READING RBUF DID NOT CLEAR RCVR DONE
```

```
2845 010464 000005
2846 010466 052777 000004 172144
2847 010474 005000
        010476 012701 000002
        010502 105777 172130
        010506 005300
        010510 001374
        010512 005301
        010514 001372
2848 010516 005077 172120
2849 010522 105777 172106
2850 010526 100375
2851 010530 017700 172102
2852 010534 042777 000004 172116
        2853 010542 105777 172066
2854 010546 001401
2855 010550 104070
2856
2857 010552 000240
2858
```



2859

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

2860	010554	000004			TST32: SCOPE		
2861	010556	032737	000001	003002	BIT	#BIT0,FLAG44	:** 11/44 ??
2862	010564	001044			BNE	10\$	:** YES DO NOT EXECUTE THIS TEST
2863	010566	000005			RESET		:CLEAR EVERYTHING
2864	010570	052777	000001	172036	BIS	#BIT0,@RCSR	:SET RDR ENABLE
2865	010576	052777	000004	172034	BIS	#BIT2,@TCSR	:SET MAINTENANCE WRAP
	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	@RBUF	
	010616	005300			DEC	R0	:**
	010620	001374			BNE	42\$	:**
	010622	005301			DEC	R1	:**
	010624	001372			BNE	42\$	:**
2866	010626	005077	172010		CLR	@TBUF	:LOAD TRANSMITTER
2867	010632	105777	171776		1\$: TSTB	@RCSR	:WAIT FOR RECEIVER DONE
2868	010636	100375			BPL	1\$	
2869	010640	032777	000001	171766	BIT	#BIT0,@RCSR	:VERIFY RCV ACTIVE CLEARED RDR ENABLE
2870	010646	001401			BEQ	2\$	:BR IF CLEAR
2871							
2872	010650	104117			ERROR	+117	:RDR ENABLE NOT CLEARED WITH RCV ACTIVE
2873							
2874	010652	052777	000001	171754	2\$: BIS	#BIT0,@RCSR	:CLEAR DONE BY SETTING RDR ENABLE
2875	010660	105777	171750		TSTB	@RCSR	:CHECK FOR DONE CLEAR
2876	010664	001404			BEQ	10\$	:** BR, IF CLEAR TO NEXT TEST
2877	010666	042777	000004	171764	BIC	#BIT2,@TCSR	:** DISABLE MAINTENANCE MODE FOR
							:** CONSOLE TO ALLOW FOR COMMUNICATION
							:** WITH TERMINAL.
2878	010674	104067			ERROR	+67	
2879							:SETTING RDR ENABLE DID NOT CLEAR RCVR DONE
2880	010676	000240			10\$: NOP		:**

2881

.SBTTL TEST # 33 - TEST THAT RCVR INTERRUPTS ONLY WHEN ENABLED  
 :\*\*\*\*\*  
 :\*TEST 33 TEST THAT RCVR INTERRUPTS ONLY WHEN ENABLED  
 :\*\*\*\*\*

2882	010700	000004				TST33: SCOPE		
	010702	042777	000100	171730		BIC	#BIT6,@TCSR	:DISABLE TRANSMIT INTERRUPTS
2883	010710	042777	000100	171716		BIC	#BIT6,@RCSR	:DISABLE RECEIVER INTERRUPTS
2884	010716	052777	000004	171714		BIS	#BIT2,@TCSR	:SET MAINTENANCE WRAP
2885	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	@RBUF	
	010736	005300				DEC	R0	:**
	010740	001374				BNE	42\$	:**
	010742	005301				DEC	R1	:**
	010744	001372				BNE	42\$	:**
2886	010746	017703	171672			MOV	@RVECT,R3	:SAVE RECEIVE VECTOR
2887	010752	012777	011010	171664		MOV	#2\$,@RVECT	:POINT RCV VECTOR TO ERROR REPORT
2888	010760	004737	014502			JSR	PC,WRPSW	:SET PSW TO PRIORITY 3
2889	010764	000140					.WORD 140	
2890	010766	005077	171650			CLR	@TBUF	:SEND A CHARACTER
2891	010772	105777	171636		1\$:	TSTB	@RCSR	:WAIT FOR RECEIVER DONE
2892	010776	100375				BPL	1\$	
2893	011000	042777	000004	171652		BIC	#BIT2,@CTCSR	:** DISABLE MAINTENANCE MODE FOR
								:** CONSOLE TO ALLOW FOR COMMUNICATION
								:** WITH TERMINAL.
2894	011006	000405				BR	3\$	:CONTINUE TEST
2895								
2896	011010				2\$:			
2897	011010	042777	000004	171642		BIC	#BIT2,@CTCSR	:** DISABLE MAINTENANCE MODE FOR
								:** CONSOLE TO ALLOW FOR COMMUNICATION
								:** WITH TERMINAL.
2898	011016	022626				CMP	(SP)+,(SP)+	:RESTORE SP AFTER INTERRUPT
2899	011020	104071				ERROR	+71	:RECEIVER INTERRUPTS WITH INT. ENABLE CLEAR
2900								
2901	011022	012777	011060	171614	3\$:	MOV	#4\$,@RVECT	:POINT RCV VECTOR TO END OF TEST
2902	011030	052777	000100	171576		BIS	#BIT6,@RCSR	:ENABLE RCV INTERRUPTS
2903	011036	000240				NOP		:** GIVE ANY INTERRUPTS TIME
2904	011040	000240				NOP		:** GIVE ANY INTERRUPTS TIME
2905	011042	000240				NOP		:** GIVE ANY INTERRUPTS TIME
2906	011044	000240				NOP		:** GIVE ANY INTERRUPTS TIME
2907	011046	042777	000004	171604		BIC	#BIT2,@CTCSR	:** DISABLE MAINTENANCE MODE FOR
								:** CONSOLE TO ALLOW FOR COMMUNICATION
								:** WITH TERMINAL.
2908	011054	104072				ERROR	+72	:RCVR DID NOT INTERRUPT
2909	011056	000401				BR	5\$	:BRANCH OVER STACK CORRECTION
2910	011060	022626			4\$:	CMP	(SP)+,(SP)+	:RESTORE SP AFTER INTERRUPT
2911	011062	042777	000100	171544	5\$:	BIC	#BIT6,@RCSR	:DISABLE INTERRUPTS
2912	011070	042777	000004	171562		BIC	#BIT2,@CTCSR	:** DISABLE MAINTENANCE MODE FOR
								:** CONSOLE TO ALLOW FOR COMMUNICATION
								:** WITH TERMINAL.
2913	011076	010377	171542			MOV	R3,@RVECT	:RESTORE RECEIVE VECTOR

2914

.SBTTL TEST # 34 - TEST THAT RCVR INTERRUPTS DO NOT OCCUR WHEN DISABLED  
\*\*\*\*\*  
:TEST 34 TEST THAT RCVR INTERRUPTS DO NOT OCCUR WHEN DISABLED  
\*\*\*\*\*  
TST34:

```
2915 011102 000004
2915 011104 000005
2916 011106 004737 014502
2917 011112 000340
2918 011114 017703 171524
2919 011120 012777 011216 171516
2920 011126 052777 000004 171504
2921 011134 005000
    011136 012701 000002

    011142 105777 171470
    011146 005300
    011150 001374
    011152 005301
    011154 001372
2922 011156 105777 167666
2923 011162 100375
2924 011164 005077 171452
2925 011170 105777 171440
2926 011174 100375
2927 011176 052777 000100 171430
2928 011204 000240
2929 011206 000240
2930 011210 000240
2931 011212 000240
2932 011214 000405
2933 011216
2934 011216 042777 000004 171434

2935 011224 022626
2936 011226 104073
2937
2938 011230 042777 000100 171376
2939 011236 012777 011266 171400
2940 011244 004737 014502
2941 011250 000140
2942 011252 000240
2943 011254 000240
2944 011256 042777 000004 171374

2945 011264 000405
2946
2947 011266
2948 011266 042777 000004 171364

2949 011274 022626
2950 011276 104074
2951
2952 011300 010377 171340
```

```
SCOPE
RESET
JSR PC,WRPSW
    .WORD 340
MOV @RVECT,R3
MOV #2,@RVECT
BIS #BIT2,@TCSR
CLR R0
MOV #2,R1

42$: TSTB @RBUF
    DEC R0
    BNE 42$
    DEC R1
    BNE 42$

6$: TSTB @STPS
    BPL 6$
    CLR @TBUF
1$: TSTB @RCSR
    BPL 1$
    BIS #BIT6,@RCSR
    NOP
    NOP
    NOP
    NOP
    BR 3$

2$: BIC #BIT2,@TCSR

    CMP (SP)+,(SP)+
    ERROR +73

3$: BIC #BIT6,@RCSR
    MOV #4,@RVECT
    JSR PC,WRPSW
    .WORD 140
    NOP
    NOP
    BIC #BIT2,@TCSR

5$: BR 5$

4$: BIC #BIT2,@TCSR

    CMP (SP)+,(SP)+
    ERROR +74

5$: MOV R3,@RVECT
```

```
:CLEAR EVERYTHING
:SET PSW TO PRIORITY 7

:SAVE RECEIVE VECTOR
:POINT RCVR VECTOR TO ERROR REPORT
: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
:**
:**
:TEST FOR XMIT READY
:LOOP IF NOT
:SEND A CHARACTER
:WAIT FOR RECEIVER DONE

:ENABLE INTERRUPTS
:** GIVE TIME FOR INTERRUPT
:** GIVE TIME FOR INTERRUPT
:** GIVE TIME FOR INTERRUPT
:** GIVE TIME FOR INTERRUPT
:CONTINUE TEST

:** DISABLE MAINTENANCE MODE FOR
:** CONSOLE TO ALLOW FOR COMMUNICATION
:** WITH TERMINAL.
:RESTORE SP AFTER INTERRUPT
:RCVR INTERRUPTS AT PRIORITY 7

:CLEAR INTERRUPT ENABLE
:POINT RCVR VECTOR TO ERROR REPORT
:SET PSW TO PRIORITY 3

:GIVE TIME FOR ANY INTERRUPT
:GIVE TIME FOR ANY INTERRUPT
:** DISABLE MAINTENANCE MODE FOR
:** CONSOLE TO ALLOW FOR COMMUNICATION
:** WITH TERMINAL.
:BR TO END OF TEST, IF NO INTERRUPT

:** 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
```

2953

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

```

2954 011304 000004
2954 011306 000005
2955 011310 017703 171330
2956 011314 017704 171326
2957 011320 012777 011434 171316
2958 011326 012777 000340 171312
2959 011334 004737 014502
2960 011340 000140
2961 011342 052777 000004 171270
2962 011350 005000
    011352 012701 000002

    011356 105777 171254
    011362 005300
    011364 001374
    011366 005301
    011370 001372
2963 011372 005077 171244
2964 011376 105777 171232
2965 011402 100375
2966 011404 042777 000004 171246

2967 011412 052777 000100 171214
2968 011420 000240
2969 011422 000240
2970 011424 000240
2971 011426 000240
2972
2973 011430 104075
2974 011432 000401
2975 011434 022626
2976 011436 012777 011502 171200
2977 011444 004737 014502
2978 011450 000140
2979 011452 000240
2980 011454 000240
2981 011456 000240
2982 011460 000240
2983 011462 042777 000100 171144
2984 011470 010377 171150
2985 011474 010477 171146
2986 011500 000402
2987
2988 011502 022626
2989 011504 104076
2990
2991 011506 010377 171132
    
```

```

TST35: SCOPE
        RESET ;CLEAR EVERYTHING
        MOV @RVECT,R3 ;SAVE RECEIVE VECTOR
        MOV @RPSW,R4 ;SAVE RECEIVE PSW VECTOR
        MOV #2$,@RVECT ;POINT RCV VECTOR TO CONTINUE TEST
        MOV #340,@RPSW ;SET PRIORITY TO 7 AFTER INTERRUPT
        JSR PC,WRPSW ;SET PSW TO PRIORITY 3
        .WORD 140
        BIS #BIT2,@TCSR ;SET MAINTENANCE WRAP
        CLR R0 ;** DELAY ENOUGH TIME TO ALLOW ANY CHARACTERS
        MOV #2,R1 ;** THAT MIGHT BE IN THE PROCESS OF BEING
        ;** RECEIVED TO FINISH AFTER MAINTENANCE
        ;** WRAP FOR THE UART UNDER TEST IS ENABLED.
        ;** READ TO CLEAR DONE
        DEC R0
        BNE 42$
        DEC R1
        BNE 42$
        CLR @TBUF ;SEND A CHARACTER
        TSTB @RCSR ;WAIT FOR RCVR DONE
        BPL 1$
        BIC #BIT2,@CTCSR ;** DISABLE MAINTENANCE MODE FOR
        ;** CONSOLE TO ALLOW FOR COMMUNICATION
        ;** WITH TERMINAL.
        BIS #BIT6,@RCSR ;ENABLE RCV INTERRUPTS
        NOP ;** GIVE SOME TIME
        NOP ;** GIVE SOME TIME
        NOP ;** GIVE SOME TIME
        NOP ;** GIVE SOME TIME
        ERROR +75 ;RCVR INTERRUPT DID NOT OCCUR
        BR 25$ ;BRANCH OVER STACK CORRECTION
        CMP (SP)+,(SP)+ ;RESTORE SP AFTER INTERRUPT
        MOV #3$,@RVECT ;POINT RCV VECTOR TO ERROR REPORT
        JSR PC,WRPSW ;RESET PSW TO PRIORITY 3
        .WORD 140
        NOP ;** GIVE SOME TIME
        NOP ;** GIVE SOME TIME
        NOP ;** GIVE SOME TIME
        NOP ;** GIVE SOME TIME
        BIC #BIT6,@RCSR ;CLEAR INTERRUPT ENABLE
        MOV R3,@RVECT ;RESTORE RECEIVE VECTOR
        MOV R4,@RPSW ;RESTORE RECEIVE PSW VECTOR
        BR 4$ ;BR TO END OF TEST
        CMP (SP)+,(SP)+ ;RESTORE SP AFTER INTERRUPT
        ERROR +76
        MOV R3,@RVECT ;RECEIVER RE-INTERRUPTED
        ;RESTORE RECEIVE VECTOR
    
```

2992

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

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

```

2993 011512 000004
2993 011514 000005
2994 011516 004737 014502
2995 011522 000340
2996 011524 017703 171114
2997 011530 012777 011646 171106
2998 011536 052777 000100 171070
2999 011544 052777 000004 171066
3000 011552 005000
      011554 012701 000002

      011560 105777 171052
      011564 005300
      011566 001374
      011570 005301
      011572 001372
3001 011574 005077 171042
3002 011600 105777 171030
3003 011604 100375
3004 011606 042777 000004 171044

3005 011614 005777 171016
3006 011620 004737 014502
3007 011624 000140
3008 011626 000240
3009 011630 000240
3010 011632 000240
3011 011634 000240
3012 011636 042777 000100 170770
3013 011644 000402
3014
3015 011646 022626
3016 011650 104077
3017
3018 011652 010377 170766
    
```

```

TST36: SCOPE
        RESET                ;CLEAR EVERYTHING
        JSR      PC,WRPSW    ;SET PSW PRIORITY TO 7
                .WORD      340
        MOV      @RVECT,R3   ;SAVE RECEIVE VECTOR
        MOV      #2,@RVECT  ;POINT RCV VECTOR TO ERROR REPORT
        BIS      #BIT6,@RCSR ;SET RCVR INTERRUPT ENABLE
        BIS      #BIT2,@TCSR ;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
        TSTB     @RBUF
        DEC      R0
        BNE     42$
        DEC      R1
        BNE     42$
        CLR      @TBUF      ;SEND A CHARACTER
        TSTB     @RCSR      ;WAIT FOR DONE (INTERRUPT)
        BPL     1$
        BIC      #BIT2,@TCSR ;** DISABLE MAINTENANCE MODE FOR
                ;** CONSOLE TO ALLOW FOR COMMUNICATION
                ;** WITH TERMINAL.
        TST      @RBUF      ;READ RBUF TO CLEAR PENDING INTERRUPT ;DPM002
        JSR      PC,WRPSW    ;SET PSW TO PRIORITY 3
                .WORD      140
        NOP
        NOP                ;** ALLOW TIME FOR ANY ERRONEOUS INTERRUPT
        NOP                ;** ALLOW TIME FOR ANY ERRONEOUS INTERRUPT
        NOP                ;** ALLOW TIME FOR ANY ERRONEOUS INTERRUPT
        NOP                ;** ALLOW TIME FOR ANY ERRONEOUS INTERRUPT
        BIC      #BIT6,@RCSR ;NO INTERRUPT-CLEAR INT. ENABLE
        BR       3$
        CMP      (SP)+,(SP)+ ;RESTORE SP AFTER INTERRUPT
        ERROR    +77        ;READING RBUF DID NOT CLEAR INTERRUPT
        MOV      R3,@RVECT  ;RESTORE RECEIVE VECTOR
    
```

3019

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

3020	011656	000004			SCOPE				
	011660	000005			RESET				;CLEAR EVERYTHING
3021	011662	004737	014502		JSR	PC,WRPSW			;SET PGW TO PRIORITY 7
3022	011666	000340				.WORD	340		
3023	011670	017703	170750		MOV	@RVECT,R3			;SAVE RECEIVE VECTOR
3024	011674	012777	012004	170742	MOV	#2\$,@RVECT			;POINT RCV VECTOR TO ERROR REPORT
3025	011702	052777	000100	170724	BIS	#BIT6,@RCSR			;SET RCV INTERRUPT ENABLE
3026	011710	052777	000004	170722	BIS	#BIT2,@TCSR			;SET MAINTENANCE WRAP
3027	011716	005000			CLR	R0			** DELAY ENOUGH TIME TO ALLOW ANY CHARACTERS
	011720	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
	011724	105777	170706		42\$: TSTB	@RBUF			
	011730	005300			DEC	R0			**
	011732	001374			BNE	42\$			**
	011734	005301			DEC	R1			**
	011736	001372			BNE	42\$			**
3028	011740	012777	000377	170674	MOV	#377,@TBUF			;SEND AN ALL 1'S CHARACTER
3029	011746	105777	170662		1\$: TSTB	@RCSR			;WAIT FOR RCV DONE
3030	011752	100375			BPL	1\$			
3031	011754	000005			RESET				;CLEAR RCV INTERRUPT & RBUF
3032	011756	004737	014502		JSR	PC,WRPSW			;SET PSW TO PRIORITY 3
3033	011762	000140				.WORD	140		
3034	011764	000240			NOP				** ALLOW TIME FOR AN ERRONEOUS INTERRUPT
3035	011766	000240			NOP				** ALLOW TIME FOR AN ERRONEOUS INTERRUPT
3036	011770	000240			NOP				** ALLOW TIME FOR AN ERRONEOUS INTERRUPT
3037	011772	000240			NOP				** ALLOW TIME FOR AN ERRONEOUS INTERRUPT
3038	011774	042777	000100	170632	BIC	#BIT6,@RCSR			;NO INTERRUPT-CLEAR INT. ENABLE
3039	012002	000402			BR	3\$			;CONTINUE TEST
3040									
3041									
3042	012004	022626			2\$: CMP	(SP)+,(SP)+			;RESTORE SP AFTER INTERRUPT
3043	012006	104100			ERROR	+100			;RESET DID NOT CLEAR RCVR INTERRUPT
3044									
3045	012010	010377	170630		3\$: MOV	R3,@RVECT			;RESTORE RECEIVE VECTOR

3046

.SBTTL TEST # 40 - TEST THAT THE 'OR' ERROR & 'ERROR' CAN BE SET  
 :\*\*\*\*\*  
 :\*TEST 40 TEST THAT THE 'OR' ERROR & 'ERROR' CAN BE SET  
 :\*\*\*\*\*  
 TST40: SCOPE

3047	012014	000004				BIT	#BIT10,@SWR	:IS THIS TEST ENABLED
3048	012016	032777	002000	167014		BEQ	TST41	:IF NOT ENABLED, BR TO NEXT TEST
3049	012024	001465				BIT	#BIT0,FLAG44	:** IS THIS A 11/44
3050	012026	032737	000001	003002		BEQ	9\$	:** NO
3051	012034	001407				TST	CTSTFL	:** YES THIS IS 11/44. IS THIS THE CONSOLE
3052	012036	005737	002624					:** SLU
3053	012042	001404				BEQ	9\$	:** NO
3054	012044	032777	000010	166766		BIT	#BIT03,@SWR	:** THIS IS THE CONSOLE SLU.SHOULD THE OVERRUN
3055								:** ERROR TEST BE PERFORMED
3056	012052	001452				BEQ	TST41	:** NO
3057	012054	000005			9\$:	RESET		:CLEAR EVERYTHING
3058	012056	052777	000004	170554		BIS	#BIT2,@TCSR	:SET MAINTENANCE WRAP
3059	012064	005000				CLR	R0	:** DELAY ENOUGH TIME TO ALLOW ANY CHARACTERS
	012066	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.
	012072	105777	170540		42\$:	TSTB	@RBUF	:** READ TO CLEAR DONE
	012076	005300				DEC	R0	:**
	012100	001374				BNE	42\$	:**
	012102	005301				DEC	R1	:**
	012104	001372				BNE	42\$	:**
3060	012106	012700	000003			MOV	#3,R0	:SET CHARACTER COUNT TO SEND 3 CHAR.
3061	012112	005077	170524		1\$:	CLR	@TBUF	:LOAD TRANSMIT BUFFER
3062	012116	105777	170516		2\$:	TSTB	@TCSR	:WAIT FOR TRANSMIT DONE
3063	012122	100375				BPL	2\$	
3064	012124	005300				DEC	R0	:DECREMENT CHARACTER COUNT
3065	012126	001371				BNE	1\$	:BR IF ALL CHARACTERS NOT TRANSMITTED
3066	012130	042777	000004	170522		BIC	#BIT2,@TCSR	:** DISABLE MAINTENANCE MODE FOR
								:** CONSOLE TO ALLOW FOR COMMUNICATION
								:** WITH TERMINAL.
3067	012136	032777	040000	170472		BIT	#BIT14,@RBUF	:TEST FOR 'OR' ERROR FLAG
3068	012144	001001				BNE	3\$	:BR, IF SET
3069	012146	104101				ERROR	+101	
3070								: 'OR' ERROR FLAG DID NOT SET
3071								
3072	012150	032777	100000	170460	3\$:	BIT	#BIT15,@RBUF	:TEST 'ERROR' FLAG
3073	012156	001001				BNE	4\$	:BR, IF SET
3074	012160	104102				ERROR	+102	
3075								: 'ERROR' FLAG DID NOT SET WITH 'OR' FLAG
3076	012162				4\$:			
3077	012162	005000				CLR	R0	:** DELAY ENOUGH TIME TO ALLOW ANY CHARACTERS
	012164	012701	000002			MOV	#2,R1	:** THAT MIGHT BE IN THE PROCESS OF BEING
	012170	005300			40\$:	DEC	R0	:** TRANSMITTED TO FINISH BEFORE MAINTENANCE
	012172	001376				BNE	40\$	:** WRAP FOR THE UART UNDER TEST IS DISABLED.
	012174	005301				DEC	R1	:** THIS WILL INHIBIT ANY COMMUNICATION
	012176	001374				BNE	40\$	:** TO HARDWARE MEDIA SUCH AS TU58 THAT MIGHT
								:** BE ATTACHED TO UART UNDER TEST.

3078

.SBTTL TEST # 41 - TEST THAT BREAK TRANSMITS ALL ZEROES  
 :\*\*\*\*\*  
 :\*TEST 41 TEST THAT BREAK TRANSMITS ALL ZEROES  
 :\*\*\*\*\*

```

TST41: SCOPE
3079 012200 000004          BIT      #BIT8,@SWR      ;IS BREAK FUNCTION TEST ENABLED?
3080 012202 032777 000400 166630 BEQ      TST42          ;BR TO NEXT TEST, IF NOT ENABLED
3081 012210 001501          BIT      #BIT0,FLAG44  ;** IS THIS A 11/44
3082 012212 032737 000001 003002 BEQ      9$            ;** NO
3083 012220 001407          TST      CTSTFL        ;** YES THIS IS 11/44. IS THIS THE CONSOLE
3084 012222 005737 002624          ;** SLU
3085 012226 001404          BEQ      9$            ;** NO
3086 012230 032777 000010 166602 BIT      #BIT03,@SWR   ;** THIS IS THE CONSOLE SLU.SHOULD THE BREAK
3087                                ;** TEST BE PERFORMED
3088 012236 001466          BEQ      TST42        ;** NO
3089 012240 000005          9$: RESET            ;CLEAR EVERYTHING
3090 012242 052777 000004 170370 BIS      #BIT2,@TCSR   ;SET MAINTENANCE WRAP
3091 012250 005000          CLR      R0           ;** DELAY ENOUGH TIME TO ALLOW ANY CHARACTERS
    012252 012701 000002          MOV      #2,R1        ;** THAT MIGHT BE IN THE PROCESS OF BEING
    ;** RECEIVED TO FINISH AFTER MAINTENANCE
    ;** WRAP FOR THE UART UNDER TEST IS ENABLED.
    ;** READ TO CLEAR DONE
    ;**
    ;**
    42$: TSTB      @RBUF          ;TRANSMIT ALL ONES TO RCVR
    DEC      R0              ;WAIT FOR RCVR DONE
    BNE     42$
    DEC     R1
    BNE     42$
3092 012272 012777 177777 170342 MOV      #-1,@TBUF
3093 012300 105777 170330 1$: TSTB      @RCSR
3094 012304 100375          BPL      1$
3095 012306 005077 170324          CLR      @RBUF        ;CLEAR DONE (LEAVING ALL ONES IN RBUF)
3096 012312 052777 000001 170320 BIS      #BIT0,@TCSR   ;TRANSMIT BREAK
3097 012320 005000          CLR      R0           ;CLEAR A TIMER
3098 012322 117737 170306 001026 2$: MOVB     @RCSR,$BDDAT ;WAIT FOR RCVR DONE
3099 012330 100411          BMI     CONT41        ;BR IF DONE
3100 012332 005200          INC     R0            ;IF NOT, INCREMENT TIMER
3101 012334 001372          BNE     2$            ;BR IF TIME REMAINS
3102
3103 012336 042777 000001 170274 BIC      #BIT0,@TCSR   ;CLEAR BREAK BIT
3104 012344 042777 000004 170306 BIC      #BIT2,@CTCSR ;** DISABLE MAINTENANCE MODE FOR
    ;** CONSOLE TO ALLOW FOR COMMUNICATION
    ;** WITH TERMINAL.
    ;BREAK DID NOT TRANSMIT ANYTHING
3105 012352 104103          ERROR   +103
3106
3107 012354 105777 170256          CONT41: TSTB     @RBUF ;CHECK RECEIVE BUFFER FOR ZERO
3108 012360 001407          BEQ     3$            ;BR, IF ZERO
3109 012362 042777 000001 170250 BIC      #BIT0,@TCSR   ;CLEAR BREAK BIT
3110 012370 042777 000004 170262 BIC      #BIT2,@CTCSR ;** DISABLE MAINTENANCE MODE FOR
    ;** CONSOLE TO ALLOW FOR COMMUNICATION
    ;** WITH TERMINAL.
3111
3112 012376 104103          ERROR   +103          ;BREAK DID NOT TRANSMIT ALL ZEROES
3113
3114 012400 042777 000001 170232 3$: BIC      #BIT0,@TCSR   ;CLEAR BREAK BIT
3115 012406 042777 000004 170244 BIC      #BIT2,@CTCSR ;** DISABLE MAINTENANCE MODE FOR
    ;** CONSOLE TO ALLOW FOR COMMUNICATION
    ;** WITH TERMINAL.
    
```



3116

.SBTTL TEST # 42 - TEST THAT 'FR' ERROR CAN BE SET DURING BREAK  
 :\*\*\*\*\*  
 :\*TEST 42 TEST THAT 'FR' ERROR CAN BE SET DURING BREAK  
 :\*\*\*\*\*

3117	012414	000004				TST42: SCOPE	
3118	012416	032777	002000	166414		BIT #BIT10,@SWR	:IS THE 'TEST ERROR FLAGS' BIT SET
3119	012424	001464				BEQ TST43	:BR TO NEXT TEST, IF NOT SET
3120	012426	032777	000400	166404		BIT #BIT8,@SWR	:IS BREAK FUNCTION ENABLED
3121	012434	001460				BEQ TST43	:BR TO NEXT TEST, IF NOT SET
3122	012436	032737	000001	003002		BIT #BIT0,FLAG44	** IS THIS A 11/44
3123	012444	001407				BEQ 9\$	** NO
3124	012446	005737	002624			TST CTSTFL	** YES THIS IS 11/44. IS THIS THE CONSOLE
3125	012452	001404				BEQ 9\$	** SLU
3126	012454	032777	000010	166356		BIT #BIT03,@SWR	** NO
3127							** THIS IS THE CONSOLE SLU.SHOULD THE FRAME
3128	012462	001445				BEQ TST43	** ERROR TEST BE PERFORMED
3129	012464	000005			9\$:	RESET	** NO
3130	012466	052777	000004	170144		BIS #BIT2,@TCSR	:CLEAR EVERYTHING
3131	012474	005000				CLR R0	:SET MAINTENANCE WRAP
	012476	012701	000002			MOV #2,R1	** 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.
	012502	105777	170130		42\$:	TSTB @RBUF	** READ TO CLEAR DONE
	012506	005300				DEC R0	**
	012510	001374				BNE 42\$	**
	012512	005301				DEC R1	**
	012514	001372				BNE 42\$	**
3132	012516	052777	000001	170114		BIS #BIT0,@TCSR	:SEND BREAK
3133	012524	005077	170112			CLR @TBUF	:TRANSMIT A CHARACTER TO TIME BREAK
3134	012530	105777	170100		1\$:	TSTB @RCSR	:WAIT FOR RCVR DONE
3135	012534	100375				BPL 1\$	
3136	012536	042777	000001	170074		BIC #BIT0,@TCSR	:CLEAR BREAK BIT
3137	012544	042777	000004	170106		BIC #BIT2,@TCSR	** DISABLE MAINTENANCE MODE FOR
							** CONSOLE TO ALLOW FOR COMMUNICATION
							** WITH TERMINAL.
3138	012552	032777	020000	170056		BIT #BIT13,@RBUF	:CHECK FOR FRAMING ERROR FLAG
3139	012560	001001				BNE 2\$	:BR, IF SET
3140							
3141	012562	104104				ERROR +104	
3142							:BREAK DID NOT SET FRAMING ERROR
3143	012564	032777	100000	170044	2\$:	BIT #BIT15,@RBUF	:TEST 'ERROR' FLAG
3144	012572	001001				BNE 3\$	:BR, IF SET
3145							
3146	012574	104114				ERROR +114	
3147							: 'ERROR' FLAG DID NOT SET WITH 'OR' FLAG
3148	012576				3\$:		

```

3149          .SBTTL TEST # 43 - TEST DATA PATH FROM XMIT TO REC USING MAINT WRAP
              :*****
              :*TEST 43          TEST DATA PATH FROM XMIT TO REC USING MAINT WRAP
              :*****
              TST43: SCOPE

012576 000004
3150
3151 012600 000005          RESET          :CLEAR EVERYTHING
3152 012602 052777 000004 170030          BIS          #BIT2,@TCSR          :SET MAINTENANCE WRAP
3153
3154
3155
3156 012610 005000          CLR          R0          :TRANSMIT A BINARY COUNT PATTERN - UP
012612 012701 000002          MOV          #2,R1          :TO THE BIT POSITION INDICATED BY THE
                                :CONTENTS OF LOCATION '$USWR'
                                **: ** DELAY ENOUGH TIME TO ALLOW ANY CHARACTERS
                                **: ** THAT MIGHT BE IN THE PROCESS OF BEING
                                **: ** RECEIVED TO FINISH AFTER MAINTENANCE
                                **: ** WRAP FOR THE UART UNDER TEST IS ENABLED.
                                **: ** READ TO CLEAR DONE
                                **: **
012616 105777 170014          42$: TSTB          @RBUF          :CLEAR REGISTER FOR TEST DATA
012622 005300          DEC          R0          :INCREMENT THE TEST DATA
012624 001374          BNE          42$          **: ** 11/44 CPU
012626 005301          DEC          R1          **: ** TEST ALL DATA
012630 001372          BNE          42$          **: ** CHECK FOR CONT-P IN TEST DATA
3157 012632 005001          CLR          R1          **: ** DO NOT XMIT ^P
3158 012634 105201          1$: INCB          R1          **: **
3159 012636 032737 000001 003002          BIT          #BIT0,FLAG44          **: ** DO NOT XMIT 220
3160 012644 001406          BEQ          5$          :XMIT A CHARACTER
3161 012646 122701 000020          CMPB         #20,R1          :WAIT FOR RECEIVER DONE
3162 012652 001770          BEQ          1$          :GET RECEIVED CHARACTER
3163 012654 122701 000220          CMPB         #220,R1          :CLEAR LOWEST UNUSED DATA BIT POSITITON IN TEST DATA
3164 012660 001765          BEQ          1$          :COMPARE DATA
3165 012662 010177 167754          5$: MOV          R1,@TBUF          :BR, IF NON-COMPARE
3166 012666 105777 167742          2$: TSTB          @RCSR          :TEST XMIT DATA FOR ZERO
3167 012672 100375          BPL          2$          :BR, IF FINISHED
3168 012674 017702 167736          MOV          @RBUF,R2          :CONTINUE IF NOT
3169 012700 043701 001112          BIC          @#BUSWR,R1          :STORE THE EXPECTED DATA
3170 012704 020102          CMP          R1,R2          :STORE RECEIVED DATA
3171 012706 001003          BNE          3$          **: ** DISABLE MAINTENANCE MODE FOR
3172 012710 105701          TSTB          R1          **: ** CONSOLE TO ALLOW FOR COMMUNICATION
3173 012712 001411          BEQ          4$          **: ** WITH TERMINAL.
3174 012714 000747          BR          1$          :DATA COMPARE DATA
3175 012716 010137 001024          3$: MOV          R1,$GDDAT          **: **
3176 012722 010237 001026          MOV          R2,$BDDAT          **: ** DISABLE MAINTENANCE MODE FOR
3177 012726 042777 000004 167724          BIC          #BIT2,@CTCSR          **: ** CONSOLE TO ALLOW FOR COMMUNICATION
                                **: ** WITH TERMINAL.

3178 012734 104105          ERROR          +105
3179
3180 012736
3181 012736 042777 000004 167714          4$: BIC          #BIT2,@CTCSR          **: **
                                **: ** DISABLE MAINTENANCE MODE FOR
                                **: ** CONSOLE TO ALLOW FOR COMMUNICATION
                                **: ** WITH TERMINAL.

3182
3183
3184
    
```

3185

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

\*\*\*\*\*  
 \*TEST 44 TEST DATA PATHS USING LOOP-BACK CONNECTOR  
 \*\*\*\*\*

3186	012744	000004				TST44: SCOPE			
3186	012746	032777	000200	166064		BIT	#BIT7,@SWR		:IS THIS TEST ENABLED
3187	012754	001442				BEQ	TST45		:BR, IF NOT
3188	012756	000005				RESET			:CLEAR EVERYTHING
3189	012760	005001				CLR	R1		:CLEAR REGISTER FOR TEST DATA
3190									:TRANSMIT A BINARY COUNT PATTERN - UP
3191									:TO THE BIT POSITION INDICATED BY THE
3192									:CONTENTS OF LOCATION '\$USWR'
3193	012762	105201			1\$:	INCB	R1		:INCREMENT THE TEST DATA
3194	012764	032737	000001	003002		BIT	#BIT0,FLAG44		:11/44 CPU
3195	012772	001404				BEQ	5\$		:TEST ALL DATA
3196	012774	023727	002634	177560		CMP	RCSR,#177560		:IS THIS 11/44 CONSOLE TERMINAL SLJ?
3197	013002	001427				BEQ	TST45		:YES, SKIP THIS TEST
3198	013004	010177	167632		5\$:	MOV	R1,@TBUF		:XMIT A CHARACTER
3199	013010	005000				CLR	R0		:CLEAR A TIMER
3200	013012	105777	167616		2\$:	TSTB	@RCSR		:WAIT FOR RECEIVER DONE
3201	013016	100403				BMI	3\$		:BR IF DONE
3202	013020	005200				INC	R0		:INCREMENT TIMER IF NOT
3203	013022	001373				BNE	2\$		:BR IF TIME REMAINS
3204									
3205	013024	104064				ERROR	+64		:RECEIVER DONE NOT SET
3206									
3207	013026	017702	167604		3\$:	MOV	@RBUF,R2		:GET RECEIVED CHARACTER
3208	013032	043701	001112			BIC	@#\$USWR,R1		:CLEAR LOWEST UNUSED DATA BIT POSITITON IN TEST DATA
3209	013036	020102				CMP	R1,R2		:COMPARE DATA
3210	013040	001003				BNE	4\$		:BR, IF NON-COMPARE
3211	013042	105701				TSTB	R1		:TEST XMIT DATA FOR ZERO
3212	013044	001406				BEQ	TST45		:BR, IF FINISHED
3213	013046	000745				BR	1\$		:CONTINUE IF NOT
3214	013050	010137	001024		4\$:	MOV	R1,\$GDDAT		:STORE EXPECTED DATA
3215	013054	010237	001026			MOV	R2,\$BDDAT		:STORE RECEIVED DATA
3216									
3217	013060	104106				ERROR	+106		:DATA COMPARE ERROR USING LOOP-BACK CONNECTOR

3218

.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  
 \*\*\*\*\*

3219	013062	000004			TST45:	SCOPE			
	013064	000005				RESET		:	CLEAR EVERYTHING
3220	013066	005037	002444			CLR	CLKCNT	:	INITIALIZE CLOCK COUNT TO ZERO
3221	013072	004737	014502			JSR	PC,WRPSW	:	SET PRIORITY TO 7
3222	013076	000340				.WORD	340		
3223	013100	017703	167544			MOV	@TVECT,R3	:	SAVE XMIT VECTOR
3224	013104	017704	167534			MOV	@RVECT,R4	:	SAVE RECEIVE VECTOR
3225	013110	017705	167562			MOV	@RTCVT,R5	:	SAVE CLOCK VECTOR
3226	013114	017737	167532	002446		MOV	@TPSW,STPSW	:	SAVE XMIT PSW VECTOR
3227	013122	017737	167520	002450		MOV	@RPSW,SRPSW	:	SAVE RECEIVE PSW VECTOR
3228	013130	017737	167544	002452		MOV	@RTCPSW,SCPSW	:	SAVE CLOCK PSW VECTOR
3229	013136	012777	013542	167504		MOV	#XMIT,@TVECT	:	POINT TRANSMIT VECTOR TO TRANSMIT ROUTINE
3230	013144	012777	000200	167500		MOV	#200,@TPSW	:	NO MULTIPLE INTERRUPTS ALLOWED
3231	013152	012777	013614	167464		MOV	#RCV,@RVECT	:	POINT RECEIVE VECTOR TO RECEIVE ROUTINE
3232	013160	012777	000200	167460		MOV	#200,@RPSW	:	NO MULT INTERRUPTS
3233	013166	005737	002624			TST	CTSTFL	:	IS CONSOLE UNDER TEST?
3234	013172	001415				BEQ	1\$	:	IF NOT SKIP CLOCK SET UP
3235	013174	032777	000100	165636		BIT	#BIT6,@SWR	:	IF YES, ARE CLOCK TEST DISABLED?
3236	013202	001011				BNE	1\$	:	IF YES, SKIP CLOCK SET UP
3237	013204	012777	013630	167464		MOV	#CLK,@RTCVT	:	POINT VECTOR TO CLOCK INTERRUPT ROUTINE
3238	013212	012777	000300	167460		MOV	#300,@RTCPSW	:	NO MULT INTERRUPTS
3239	013220	052777	000100	167446		BIS	#BIT6,@LKS	:	ENABLE CLOCK INTERRUPTS
3240	013226	052777	000004	167404	1\$:	BIS	#BIT2,@TCSR	:	SET MAINTENANCE WRAP
3241	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		42\$:	TSTB	@RBUF	:	
	013246	005300				DEC	R0	:	
	013250	001374				BNE	42\$	:	
	013252	005301				DEC	R1	:	
	013254	001372				BNE	42\$	:	
3242	013256	052777	000100	167354		BIS	#BIT6,@TCSR	:	ENABLE TRANSMIT INTERRUPTS
3243	013264	052777	000100	167342		BIS	#BIT6,@RCSR	:	ENABLE RECEIVE INTERRUPTS
3244	013272	005037	002442			CLR	XMTCNT	:	CLEAR XMIT INTERRUPT COUNTER
3245	013276	005037	002440			CLR	RCVCNT	:	CLEAR RCV INTERRUPT COUNTER
3246	013302	005001				CLR	R1	:	CLEAR A REGISTER FOR TEST DATA USE
3247	013304	005000				CLR	R0	:	CLEAR TIMER
3248	013306	012702	002454			MOV	#BUF,R2	:	POINT R2 TO RECEIVE DATA STORAGE
3249	013312	005077	167324			CLR	@TBUF	:	SEND FIRST CHARACTER
3250	013316	004737	014502			JSR	PC,WRPSW	:	SET PSW TO PRIORITY 3
3251	013322	000140				.WORD	140		
3252								:	WAIT FOR INTERRUPTS
3253	013324	000240			2\$:	NOP		:	STALL
3254	013326	000240				NOP		:	STALL
3255	013330	062700	000000			ADD	#0,R0	:	ADD INSTRUCTIONS ARE USED TO LENGTHEN LOOP TIME
3256	013334	062700	000001			ADD	#1,R0	:	TO COVER THE SLOWEST BAUD RATE ON THE FASTEST CPU
3257	013340	001371				BNE	2\$	:	
3258	013342	032777	000100	167270		BIT	#BIT6,@TCSR	:	FINISHED ENTIRE TRANSMISSION
3259	013350	001402				BEQ	3\$	:	BR, IF INTERRUPTS ARE DISABLED (FINISHED)
3260	013352	000005				RESET		:	CLEAR EVERYTHING
3261									
3262	013354	104107				ERROR	+107	:	TRANSMIT INTERRUPT TIMEOUT IN MAIN. DATA TEST

```

3263
3264 013356 023737 002442 002440 3$: CMP XMTCNT,RCVCNT ;COMPARE THE NUMBER OF INTERRUPTS
3265 013364 001402 BEQ 4$ ;BR, IF EQUAL
3266 013366 000005 RESET ;CLEAR EVERYTHING
3267
3268 013370 104110 ERROR +110 ;RECEIVER DID NOT GET FULL TRANSMISSION
3269 ; IF RCVCNT=0, NO DATA RECEIVED
3270 ; IF RCVCNT=?, THEN (XMTCNT-RCVCNT)
3271 ; EQUALS THE NO, OF INTERRUPTS LOST.
3272 013372 005737 002624 4$: TST CTSTFL ;IS CONSOLE UNDER TEST?
3273 013376 001411 BEQ 5$ ;IF NOT, SKIP CLOCK COUNT CHECK
3274 013400 032777 000100 165432 BIT #BIT6,@SWR ;IF YES, ARE CLOCK TESTS DISABLED?
3275 013406 001005 BNE 5$ ;IF YES, SKIP CLOCK COUNT CHECK
3276 013410 005737 002444 TST CLKCNT ;CHECK FOR AT LESST ONE CLOCK INTERRUPT
3277 013414 001002 BNE 5$ ;BR IF INTERRUPTS OCCURRED
3278 013416 000005 RESET ;CLEAR MAINTENANCE WRAPAROUND
3279 013420 104113 ERROR +113 ;NO CLOCK INTERRUPTS IN EXERCISER
3280
3281 013422 000005 5$: RESET ;CLEAR EVERYTHING
3282 013424 012700 002454 MOV #BUF,R0 ;LOAD RECEIVED DATA POINTER TO R0
3283 013430 005001 CLR R1 ;SET UP REGISTER FOR COMPARISON
3284 013432 022001 COMP: CMP (R0)+,R1 ;COMPARE XMIT & RCV DATA
3285 013434 001014 BNE 6$ ;BR, IF NOT EQUAL
3286 013436 105201 9$: INCB R1 ;INCREMENT COMPARE DATA
3287 013440 032737 000001 003002 BIT #BIT0,FLAG44 ;11/44 CPU?
3288 013446 001403 BEQ 8$ ;YES, SKIP
3289 013450 122701 000020 CMPB #20,R1 ;SKIP 20 DATA IF 11/44
3290 013454 001770 BEQ 9$ ;SKIP 20
3291 013456 032701 000040 8$: BIT #BIT5,R1 ;FINISHED CHECKING RECEIVED DATA?
3292 013462 001763 BEQ COMP ;BR, IF NOT FINISHED
3293 013464 000405 BR 7$ ;BR TO END OF TEST
3294
3295 013466 014037 001026 6$: MOV -(R0), $BDDAT ;STORE BAD DATA FOR ERROR REPORT
3296 013472 010137 001024 MOV R1, $GDDAT ;STORE GOOD DATA FOR ERROR REPORT
3297 013476 104111 ERROR +111 ;DATA COMPARE ERROR IN EXERCISER
3298
3299 013500 010377 167144 7$: MOV R3,@TVECT ;RESTORE XMIT VECTOR
3300 013504 010477 167134 MOV R4,@RVECT ;RESTORE RECEIVE VECTOR
3301 013510 010577 167162 MOV R5,@RTCVT ;RESTORE CLOCK VECTOR
3302 013514 013777 002446 167130 MOV STPSW,@TPSW ;RESTORE XMIT PSW VECTOR
3303 013522 013777 002450 167116 MOV SRPSW,@RPSW ;RESTORE RECEIVE PSW VECTOR
3304 013530 013777 002452 167142 MOV SCPSW,@RTCPSW ;RESTORE CLOCK PSW VECTOR
3305 013536 000137 014272 JMP ENDEV ;GOTO NEXT TEST
3306
3307 013542 005237 002442 XMIT: INC XMTCNT ;INCREMENT XMIT INTERRUPT COUNTER
3308 013546 105201 1$: INCB R1 ;INCREMENT TEST DATA
3309 013550 032737 000001 003002 BIT #BIT0,FLAG44 ;11/44 CPU
3310 013556 001403 BEQ 2$ ;TEST ALL DATA
3311 013560 122701 000020 CMPB #20,R1 ;CHECK DATA FOR ^P
3312 013564 001770 BEQ 1$ ;DO NOT XMIT ^P
3313 013566 032701 000040 2$: BIT #BIT5,R1 ;SEND DATA PATTERN FROM 00 --> 37
3314 013572 001404 BEQ XCONT ;BR, IF MORE DATA TO BE SENT
3315 013574 042777 000100 167036 BIC #BIT6,@TCSR ;CLEAR XMIT INTERRUPT ENABLE
3316 013602 000402 BR XRET ;RETURN, WITHOUT SENDING ANY MORE DATA
3317 013604 110177 167032 XCONT: MOVB R1,@TBUF ;SEND NEW CHARACTER
3318 013610 005000 XRET: CLR R0 ;CLEAR TIMER
3319 013612 000002 RTI ;RETURN
    
```

```
3320
3321 013614 017722 167016      RCV:  MOV   @RBUF, (R2)+  ;STORE RECEIVED DATA
3322 013620 005237 002440      RCV:  INC   RVCNT        ;INCREMENT RCV INTERRUPT COUNTER
3323 013624 005000              RCV:  CLR   R0           ;CLEAR TIMER
3324 013626 000002              RCV:  RTI                    ;RETURN
3325
3326 013630 005237 002444      CLK:  INC   CLKCNT       ;INCREMENT CLOCK INTERRUPT COUNT
3327 013634 000002              CLK:  RTI                    ;RETURN
```

3328

.SBTTL TEST # 46 - TEST CONSOLE WITH WRAP AROUND

\*\*\*\*\*  
\*TEST 46 TEST CONSOLE WITH WRAP AROUND  
\*\*\*\*\*

013636 000004

TST46: SCOPE

\*\*\*\*\*  
: WRAPAROUND TESTING- AUTO INITIATION OF T/A CONSOLE TEST

3329

3330

3331

3332

3333

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

176500  
176502  
176504  
176506  
003014  
025222

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  
\*\*\*\*\*

\*\*\*\*\*  
: RUN THIS TEST ONLY IF  
: SW BIT 2 IS ON  
: SAVE OLD PSW  
: PUT IN NOW PSW  
: USE MY STACK (SO NO CLOBER)  
: TIMING LOOP COUNTERS  
: SAVE LOCATIONS T/A WRITES  
: CALCULATE CHECKSUM  
: SAVE OLD CHECKSUM  
: SEND OUT CONTROL P  
: GET CRLF CONSOLE>>>  
: SEND OUT T/A<CRLF>

013640 032777 000004 165172 WRAP:  
013646 001451  
013650 013737 177776 002434  
013656 012737 000340 177776  
013664 012706 002432  
013670 005037 002340  
013674 012737 000100 002342  
013702 004537 014202  
013706 004537 014010  
013712 010437 002436  
013716 012700 002564  
013722 004537 014030  
013726 012700 002566  
013732 004537 014062  
013736 012700 002610  
013742 004537 014030

BIT #BIT2,@SWR  
BEQ 10\$  
MOV PSW,SAVEPS  
MOV #340,PSW  
MOV #JIMSTK,SP  
CLR LOC1  
MOV #100,LOC2  
JSR R5,SAVETE  
JSR R5,CHKSUM  
MOV R4,OLDSUM  
MOV #CNTLP,R0  
JSR R5,PUTLIN  
MOV #PROMPT,R0  
JSR R5,GETLIN  
MOV #TA,R0  
JSR R5,PUTLIN

```

3381 013746 004537 014126      JSR    R5,GETB      ;GET THE A FROM '-TESTB'
3382                               ;
3383 013752 004537 014236      JSR    R5,RESTTE   ;RESTORE LOCATIONS T/A WRITES
3384                               ;
3385 013756 004537 014010      JSR    R5,CHKSUM   ;RECALCULATE CHECKSUM
3386 013762 020437 002436      CMP    R4,OLDSUM
3387 013766 001401             BEQ    10$
3388 013770 000000             HALT
3389 013772 012706 001000      MOV    #1000,SP    ;RETURN THEIR SP
3390 013776 013737 002434 177776 10$:      MOV    SAVEPS,PSW ;RETURN PSW
3391 014004 000137 014326      JMP    $EOP        ;BR TO END OF PASS ROUTINE
3392
3393
3394
3395
3396
3397
3398
3399
3400
3401
3402
3403
3404
3405
3406
3407
3408
3409
3410 014010 012700 003014      CHKSUM:      MOV    #BGNADD,R0    ;GET STARTING ADDRESS
3411 014014 005004             CLR    R4           ;RESET SUM
3412 014016 062004 025222 1$:      ADD    (R0)+,R4    ;ADD WORD TO SUM
3413 014020 022700             CMP    #ENDADD,R0  ;CHECK FOR END
3414 014024 001374             BNE    1$          ;IF NOT DONE LOOP
3415 014026 000205             RTS    R5          ;DONE RETURN
3416
3417
3418
3419
3420
3421
3422
3423
3424
3425
3426 014030 112001             PUTLIN:      MOV    (R0)+,R1    ;GET DATA
3427 014032 001412             BEQ    10$        ;END OF DATA
3428 014034 004537 014162 176504 1$:      JSR    R5,TIMER   ;PROVIDE FOR TIMOUT
3429 014040 032737 000200             BIT    #BIT7,TCSR58 ;TEST FOR XMIT READY
3430 014046 001772             BEQ    1$        ;WAIT FOR XMIT READY
3431 014050 110137 176506             MOV    R1,TBUF58 ;OUTPUT CHAR
3432 014054 000137 014030             JMP    PUTLIN    ;REPETE
3433 014060 000205             RTS    R5        ;RETURN
3434
3435
3436
3437
    
```

```

:*****
:ROUTINE TO CALCULATE CHECKSUM ON PROGRAM
:INPUT CONDITIONS
:
:   BGNADD = ADDRESS TO START CHECK SUM (INCLUSIVE)
:   ENDADD = ADDRESS TO END CHECK SUM (EXCLUSIVE)
:OUTPUT CONDITIONS
:
:   REG4 = CHECK SUM
:*****
    
```

```

:*****
:THIS ROUTINE OUTPUTS TO THE SERIAL LINE CHARS STARTING AT
:THE ADDRESS IN REG0 UNTIL IT HITS A <377>
:*****
    
```

```

:*****
:THIS ROUTINE INPUTS A CHARS FROM THE SERIAL LINE AND COMPARES
:*****
    
```



```

3438 ; IT WITH THE EXPECTED VALUES POINTED TO BY REGO UNTIL NULL<00>
3439 :
3440 :*****
3441 :
3442 :
3443 014062 112001 GETLIN:      MOVB      (R0)+,R1      ;GET EXPECTED CHAR
3444 014064 105201      INCB      R1
3445 014066 001416      BEQ      10$          ;IF NULL EXIT
3446 014070 105301      DECB      R1
3447 014072 004537 014162 176500 1$:      JSR      R5,TIMER     ;PROVIDE FOR TIMOUT
3448 014076 032737 000200 176500      BIT      #BIT7,RCSR58 ;TEST FOR REC READY
3449 014104 001772      BEQ      1$          ;WAIT FOR REC READY
3450 014106 113702 176502      MOVB      RBUF58,R2
3451 014112 042702 000200      BIC      #BIT7,R2     ;STRIP PARITY
3452 014116 120102      CMPB      R1,R2      ;ARE THEY THE SAME
3453 014120 001760      BEQ      GETLIN     ;SAME GET MORE
3454 014122 000000      HALT
3455 014124 000205 10$:      RTS      R5          ;NOT SAME HALT
3456 ;
3457 ;
3458 ;*****
3459 ;
3460 ; THIS ROUTINE INPUTS CHARS UNTIL IT GETS THE CHAR 'B'
3461 ; AND THEN RETURNS
3462 ;
3463 ;*****
3464 ;
3465 ;
3466 014126 004537 014162 176500 GETB:      JSR      R5,TIMER     ;OUT TIMING LOOP OUT
3467 014132 032737 000200 176500      BIT      #BIT7,RCSR58 ;TEST OFR REC READY
3468 014140 001772      BEQ      GETB       ;NOT DONE YET
3469 014142 113702 176502      MOVB      RBUF58,R2
3470 014146 042702 000200      BIC      #BIT7,R2     ;STRIP PARITY
3471 014152 122702 000102      CMPB      #102,R2    ;CHECK FOR 'B'
3472 014156 001363      BNE      GETB       ;NOT 'B' REPETE
3473 014160 000205      RTS      R5          ;WAS 'B' RETURN
3474 ;
3475 ;
3476 ;*****
3477 ;
3478 ; THIS ROUTINE IS USED AS A TIME OUT FEATURE
3479 ; WHEN THE TIMING LOOPS BOTH REACH 0 THIS ROUTINE WILL
3480 ; CAUSE A HALT
3481 ;
3482 ;*****
3483 ;
3484 014162 005337 002340 TIMER:      DEC      LOC1
3485 014166 001004      BNE      10$        ;DECREPNT TIMING LOOPS
3486 014170 005337 002342      DEC      LOC2
3487 014174 001001      BNE      10$
3488 014176 000000      HALT
3489 014200 000205 10$:      RTS      R5          ;IF ZERO THIS ROUTINE
3490 ;
3491 ;
3492 ;*****
3493 ;
3494 ; THIS ROUTINE SAVES THE LOCATIONS WRITTEN BY THE T/A CONSOLE TEST
    
```

```
3495 ; SO THEY MAY BE RESTORED LATER  
3496 ;  
3497 ;*****  
3498 ;  
3499 014202 013737 000000 002344 SAVETE: MOV 0,SAVE0 ;SAVE LOCATION 0  
3500 014210 012702 000002 MOV #2,R2 ;SET UP INDIRECT PNTER  
3501 014214 012701 002346 MOV #SAVLOC,R1 ;SET UP STORAGE LOC. PNTER  
3502 014220 011221 1$: MOV (R2),(R1)+ ;GET WORD AND SAVE  
3503 014222 000241 CLC ;DONT ROT IN ANY BITS  
3504 014224 006102 ROL R2 ;SET NEXT ADDRESS  
3505 014226 020227 025222 CMP R2,#ENDADD ;SEE IF AT END  
3506 014232 100772 BMI 1$ ;NO REPETE  
3507 014234 000205 RTS R5
```

```
3508 ;  
3509 ;  
3510 ;*****  
3511 ; THIS ROUTINE RESTORES THE SAVED LOCATIONS THAT T/A WRITES INTO  
3512 ;  
3513 ;*****  
3514 ;
```

```
3515 ;  
3516 014236 013737 002344 000000 RESTTE: MOV SAVE0,0 ;SET UP INDIRECT PNTER  
3517 014244 012702 000002 MOV #2,R2 ;SET UP STORAGE LOC. PNTER  
3518 014250 012701 002346 MOV #SAVLOC,R1 ;GET WORD AND RESTORE  
3519 014254 012112 1$: MOV (R1)+,(R2) ;DONT ROT IN ANY BITS  
3520 014256 000241 CLC ;SET NEXT ADDRESS  
3521 014260 006102 ROL R2 ;SEE IF AT END  
3522 014262 020227 025222 CMP R2,#ENDADD ;NO REPETE  
3523 014266 100772 BMI 1$  
3524 014270 000205 RTS R5
```

```
3525                                     ;END OF DEVICE PASS ROUTINE
3526 014272 005037 001002             ENDEV: CLR      $TSTNM      ;CLEAR TEST NO. COUNT FOR SCOPE ROUTINE
3527 014276 005237 001076             INC      $DEVCT      ;INCREMENT DEVICE COUNTER
3528 014302 023737 002630 001076     CMP      TMP2,$DEVCT ;ALL DEVICES TESTED
3529 014310 001002                    BNE      NOEOP       ;BR, IF NO
3530 014312 000137 013640             JMP      WRAP        ;EXECUTE WRAP AROUND AFTER ALL DEVICES
3531 014316 005037 002624             NOEOP: CLR      CTSTFL ;CLEAR CONSOLE UNDER TEST FLAG
3532 014322 000137 004050             JMP      TSTDEV      ;GO TEST NEXT DEVICE
```

3534

```
.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:  
SCOPE  
CLR $STNM ;;ZERO THE TEST NUMBER  
INC $PASS ;;INCREMENT THE PASS NUMBER  
BIC #100000,$PASS ;;DON'T ALLOW A NEG. NUMBER  
DEC (PC)+ ;;LOOP?  
$EOPCT: .WORD 1  
BGT $DOAGN ;;YES  
MOV (PC)+,@(PC)+ ;;RESTORE COUNTER  
$ENDCT: .WORD 1  
TYPE ,ENDMG ;;TYPE 'END PASS'  
$GET42: MOV @#42,R0 ;;GET MONITOR ADDRESS  
BEQ $DOAGN ;;BRANCH IF NO MONITOR  
RESET ;;CLEAR THE WORLD  
$ENDAD: JSR PC,(R0) ;;GO TO MONITOR  
NOP ;;SAVE ROOM  
NOP ;;FOR  
NOP ;;ACT11  
$DOAGN:  
JMP @(PC)+ ;;RETURN  
$RTNAD: .WORD GOAGIN  
$ENULL: .BYTE -1,-1,0 ;;NULL CHARACTER STRING  
 .EVEN  
ENDMG: .ASCIZ <CR><LF>/END PASS /
```

```
014326  
014326 000004  
014330 005037 001002  
014334 005237 001074  
014340 042737 100000 001074  
014346 005327  
014350 000001  
014352 003015  
014354 012737  
014356 000001  
014360 014350  
014362 104401 014416  
014366 013700 000042  
014372 001405  
014374 000005  
014376 004710  
014400 000240  
014402 000240  
014404 000240  
014406  
014406 000137  
014410 014432  
014412 377 377 000  
3535 014416 015 012 105
```

```

3537 014432 005046          GOAGIN: CLR    -(SP)          ;CLEAR ANOTHER LOCATION ON STACK
3538 014434 005216          1$:    INC    (SP)          ;INCREMENT STACK LOCATION
3539 014436 000240          NOP                    ;TAKE UP SOME MORE TIME
3540 014440 001375          BNE    1$             ;BRANCH BACK UNTIL ZERO AGAIN
3541 014442 062706 000002    ADD    #2,SP         ;CLEAN THE LOCATION OFF THE STACK
3542 014446 005037 001076    CLR    $DEVCT       ;CLEAR DEVICE COUNT
3543 014452 022737 000001 002630  CMP    #1,TMP2      ;IS THERE ONLY ONE DEVICE UNDER TEST?
3544 014460 001004          BNE    RSTRT        ;BR, IF NOT
3545 014462 012706 001000    MOV    #1000,SP     ;RESET STACK POINTER
3546 014466 000137 004172    JMP    TST1         ;GO DO ANOTHER PASS
3547
3548 014472 005037 001100    RSTRT: CLR   $UNIT   ;CLEAR UNIT NUMBER
3549 014476 000137 003774    JMP    BEGIN
3550
3551 014502 011646          WRPSW: MOV   (SP),-(SP) ;COPY RETURN PC
3552 014504 017666 000002 000002  MOV   @2(SP),2(SP) ;MOVE NEW PSW TO STACK
3553 014512 062716 000002    ADD   #2,(SP)      ;ADJUST JSR RETURN OVER PRIORITY REQUESTED
3554 014516 000002          RTI                 ;POP RETURN PC & NEW PSW
3555
3556          ;SUBROUTINE TO REPORT UNEXPECTED OR ERRONEOUS TRAPS OR INTERRUPTS
3557
3558 014520 012600          CATCH: MOV   (SP)+,R0 ;GET ADDRESS OF TRAP VECTOR + 4
3559 014522 162700 000004    SUB   #4,R0        ;ADJUST TO POINT TO TRAP ADDRESS
3560 014526 010037 002622    MOV   R0,BDVCT     ;STORE TRAP OR INTERRUPT ADDRESS
3561 014532 016637 000002 002620  MOV   2(SP),OLDPC  ;GET PC WHERE TRAP OR INTERRUPT OCCURRED
3562 014540 104112          ERROR  +112       ;REPORT ERROR
3563
3564 014542 000000          HALT              ;PROGRAM MUST BE RESTARTED AT THIS POINT
  
```

3566  
3567  
3568  
3569  
3570  
3571  
3572  
3573  
3574  
3575  
3576  
3577  
3578  
3579 014544  
3580 014544 105237 001003  
3581 014550 001775  
3582 014552 013777 001002 164262  
3583 014560 005237 001012  
3584 014564 011637 001016  
3585 014570 162737 000002 001016  
3586 014576 117737 164214 001014  
3587 014604 032777 020000 164226  
3588 014612 001004  
3589 014614 004737 014726  
3590 014620 104401 001063  
3591 014624  
3592 014624 122737 000001 001106  
3593 014632 001007  
3594 014634 113737 001014 014646  
3595 014642 004737 015714  
3596 014646 000  
3597 014647 000  
3598 014650 000777  
3599 014652 005777 164162  
3600 014656 100001  
3601 014660 000000  
3602 014662 104406  
3603 014664 032777 001000 164146  
3604 014672 001402  
3605 014674 013716 001010  
3606 014700 005737 001060  
3607 014704 001402  
3608 014706 013716 001060  
3609 014712  
3610 014712 022737 014376 000042  
3611 014720 001001  
3612 014722 000000  
3613 014724  
3614 014724 000002  
3615

```
*****  
: 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  
: *SW09=1      LOOP IN ERROR  
: *CALL  
: *      ERROR  +N      ;;ERROR=EMT AND N=ERROR ITEM NUMBER  
: *****  
$ERROR:  
7$:      INCB      $ERFLG      ;SET THE ERROR FLAG  
      BEQ      7$      ;DON'T LET FLAG GO TO ZERO  
      MOV      $STNM,@DISPLAY ;DISPLAY TEST NUMBER AND ERROR FLAG  
      INC      $ERTTL      ;INCREMENT ERROR COUNT  
      MOV      (SP),$ERRPC      ;GET ADDRESS OF ERROR INSTRUCTION  
      SUB      #2,$ERRPC  
      MOVVB   @$ERRPC,$ITEMB ;STRIP AND SAVE THE ERROR ITEM CODE  
      BIT      #BIT13,@SWR      ;SKIP TYPEOUT IF SET  
      BNE      20$      ;SKIP TYPEOUTS  
      JSR      PC,$ERRTYP      ;GO TO USER ERROR ROUTINE  
      TYPE     .$CRLF  
20$:     CMPB     #APTENV,$ENV      ;RUNNING IN APT MODE  
      BNE     2$      ;NO, SKIP APT ERROR REPORT  
      MOVVB  $ITEMB,21$      ;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$:      TST     @SWR      ;HALT ON ERROR  
      BPL     3$      ;SKIP IF CONTINUE  
      HALT    ;HALT ON ERROR!  
3$:      CKSWR  ;TEST FOR CHANGE IN SOFT-SWR  
      BIT     #BIT09,@SWR      ;LOOP ON ERROR SWITCH SET?  
      BEQ     4$      ;BR IF NO  
      MOV     $LPERR,(SP)      ;FUDGE RETURN FOR LOOPING  
4$:      TST     $ESCAPE      ;CHECK FOR AN ESCAPE ADDRESS  
      BEQ     5$      ;BR IF NONE  
      MOV     $ESCAPE,(SP)      ;FUDGE RETURN ADDRESS FOR ESCAPE  
5$:      CMP     #SENDAD,@#42      ;ACT-11 AUTO-ACCEPT?  
      BNE     6$      ;BR IF NO  
      HALT    ;YES  
6$:      RTI      ;RETURN
```

3617  
 3618

.SBTTL ERROR MESSAGE TYPEOUT ROUTINE

::\*\*\*\*\*  
 ::\*THIS ROUTINE USES THE "ITEM CONTROL BYTE" (\$ITEMB) 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 104401 001063
014726 104401 001063
014732 010046
014734 005000
014736 153700 001014
014742 001004

014744 013746 001016

014750 104402
014752 000434
014754 122700 000177
014760 001003
014762 012700 015076
014766 000406
014770 005300
014772 006300
014774 006300
014776 006300
015000 062700 001146
015004 012037 015014
015010 001404
015012 104401
015014 000000
015016 104401 001063
015022 012037 015032
015026 001404
015030 104401
015032 000000
015034 104401 001063
015040 011000
015042 001004
015044 012600
015046 104401 001063
015052 000207
015054
015054 013046
015056 104402
015060 005710
015062 001770
015064 104401 015072
015070 000771
015072 040 040 000

015076 015106 015146 015176 PFECWS: .WORD PFECM,PFECDH,PFECDT,PFECDF
015106 120 117 127 PFECM: .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
  
```

```

$ERRTYP:
      TYPE      , $CRLF      :: 'CARRIAGE RETURN' & 'LINE FEED'
      MOV      RO, -(SP)    :: SAVE RO
      CLR      RO          :: PICKUP THE ITEM INDEX
      BISB     @#$ITEMB,RO
      BNE      1$          :: IF ITEM NUMBER IS ZERO, JUST
                                :: TYPE THE PC OF THE ERROR
                                :: SAVE $ERRPC FOR TYPEOUT
                                :: ERROR ADDRESS
                                :: GO TYPE--OCTAL ASCII(ALL DIGITS)
                                :: GET OUT
                                :: IS THIS THE POWER MONITOR BIT CALL      :DPM001
                                :: BRANCH IF NOT                          :DPM001
                                :: MOVE ADDR OF PWR MONITOR BIT ERR TO RO :DPM001
                                :: BRANCH TO CALL THE ERROR              :DPM001
                                :: ADJUST THE INDEX SO THAT IT WILL
                                :: WORK FOR THE ERROR TABLE

      MOV      $ERRPC, -(SP)

      TYPOC
      BR      6$
      CMPB     #177,RO
      BNE      100$
      MOV      #PFECWS,RO
      BR      110$
      DEC      RO
      ASL      RO
      ASL      RO
      ASL      RO
      ADD      #$ERRTB,RO
      MOV      (RO)+,2$
      BEQ      3$
      TYPE
      .WORD    0
      TYPE     , $CRLF
      MOV      (RO)+,4$
      BEQ      5$
      TYPE
      .WORD    0
      TYPE     , $CRLF
      MOV      (RO),RO
      BNE      7$
      MOV      (SP)+,RO
      TYPE     , $CRLF
      RTS      PC
      MOV      @ (RO)+, -(SP) :: SAVE @ (RO)+ FOR TYPEOUT
      TYPOC
      TST      (RO)
      BEQ      6$
      TYPE     , 8$
      BR      7$
      .ASCIZ   / /
      .EVEN
      .WORD    PFECM,PFECDH,PFECDT,PFECDF
      .ASCIZ   ?POWER MONITOR BIT WAS FOUND SET?
      .ASCIZ   ?TESTNO ERR PC CPUERR?
      .EVEN
      .WORD    $TESTN,$ERRPC,CPSAVE,0
  
```

3619 015206 000 000 000 PFECDF: .BYTE 0,0,0,0



```
3621
3622      .SBTTL POWER DOWN AND UP ROUTINES
3623      ;:*****
3624      ;*POWER DOWN ROUTINE
3625      ;:*****
3626 015212 012737 015356 000024 $PWRDN: MOV    #ILLUP,@PWRVEC ;SET FOR FAST UP
3627 015220 012737 000340 000026      MOV    #340,@PWRVEC+2 ;PRIO:7
3628 015226 010046      MOV    R0,-(SP)      ;PUSH R0 ON STACK
3629 015230 010146      MOV    R1,-(SP)      ;PUSH R1 ON STACK
3630 015232 010246      MOV    R2,-(SP)      ;PUSH R2 ON STACK
3631 015234 010346      MOV    R3,-(SP)      ;PUSH R3 ON STACK
3632 015236 010446      MOV    R4,-(SP)      ;PUSH R4 ON STACK
3633 015240 010546      MOV    R5,-(SP)      ;PUSH R5 ON STACK
3634 015242 017746 163572      MOV    @SWR,-(SP)    ;PUSH @SWR ON STACK
3635 015246 010637 015362      MOV    SP,$SAVR6    ;SAVE SP
3636 015252 012737 015264 000024      MOV    #PWRUP,@PWRVEC ;SET UP VECTOR
3637 015260 000000      HALT
3638 015262 000776      BR     .-2          ;HANG UP
3639
3640
3641      ;:*****
3642      ;*POWER UP ROUTINE
3643      ;:*****
3644 015264 012737 015356 000024 $PWRUP: MOV    #ILLUP,@PWRVEC ;SET FOR FAST DOWN
3645 015272 013706 015362      MOV    $SAVR6,SP    ;GET SP
3646 015276 012677 163536      MOV    (SP)+,@SWR   ;POP STACK INTO @SWR
3647 015302 012605      MOV    (SP)+,R5
3648 015304 012604      MOV    (SP)+,R4     ;POP STACK INTO R4
3649 015306 012603      MOV    (SP)+,R3     ;POP STACK INTO R3
3650 015310 012602      MOV    (SP)+,R2     ;POP STACK INTO R2
3651 015312 012601      MOV    (SP)+,R1     ;POP STACK INTO R1
3652 015314 012600      MOV    (SP)+,R0     ;POP STACK INTO R0
3653 015316 012737 015212 000024      MOV    #PWRDN,@PWRVEC ;SET UP THE POWER DOWN VECTOR
3654 015324 012737 000340 000026      MOV    #340,@PWRVEC+2 ;PRIO:7
3655 015332 005037 015362      CLR    $SAVR6      ;WAIT LOOP FOR THE TTY
3656 015336 005237 015362      1$:  INC    $SAVR6   ;WAIT FOR THE INC
3657 015342 001375      BNE    1$          ;OF WORD
3658 015344 104401      TYPE                    ;REPORT THE POWER FAILURE
3659 015346 015364      $PWRMG: .WORD $POWER ;POWER FAIL MESSAGE POINTER
3660 015350 013716 001006      MOV    $LPADR,(SP)  ;CHOCOLATE FUDGE RETURN TO BEGINNING OF TEST
3661 015354 000002      RTI                    ;RETURN TO THERE
3662 015356 000000      $ILLUP: HALT        ;THE POWER UP SEQUENCE WAS STARTED
3663 015360 000776      BR     .-2          ; BEFORE THE POWER DOWN WAS COMPLETE
3664 015362 000000      $SAVR6: 0
3665 015364 015 012 120 $POWER: .ASCIZ <15><12>'POWER'
```

3668  
3669

.SBTTL SCOPE HANDLER ROUTINE

```

*****
*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
*AND LOAD THE TEST NUMBER($STNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
*AND LOAD THE ERROR FLAG ($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          $SCOPE:
015374 104406          CKSWR          ;;TEST FOR CHANGE IN SOFT-SWR
015376 032777 040000 163434 1$: BIT #BIT14,@SWR          ;;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 005737 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 :DPM001
015452 001447          BEQ 2001$          ;;BRANCH AROUND ROUTINE IF SO :DPM001
015454 005227 177777          INC #-1          ;;TEST FOR 1ST TIME THROUGH :DPM001
015460 001005          BNE 900$          ;;BRANCH IF NOT :DPM001
015462 013746 000004          MOV 4,-(SP)          ;;SAVE TIMEOUT VECTOR AT 4 :DPM001
015466 012737 015542 000004          MOV #1999$,4          ;;SET VECTOR TO LOCATION BELOW :DPM001
015474 013737 177766 015674 900$: MOV 177766,CPSAVE          ;;MOVE CPU ERR REG VALUE TO LOC FOR TST :DPM001
015502 032737 000001 015674          BIT #BIT00,CPSAVE          ;;SEE IF THE POWER MONITOR BIT IS ON :DPM001
015510 001423          BEQ 2000$          ;;BRANCH TO CONTINUE ROUTINE IF CLEAR :DPM001
015512 042737 000001 177766          BIC #BIT00,177766          ;;CLEAR THE BIT FOUND TO BE SET :DPM001
015520 017746 163314          MOV @SWR,-(SP)          ;;SAVE SWR VALUE :DPM001
015524 042777 001000 163306          BIC #BIT09,@SWR          ;;DON'T ALLOW LOOP ON ERROR ON THIS ERROR :DPM001
015532 104177          EMT +177          ;;CALL SPECIAL POWER FAIL BIT ERROR CALL :DPM001
015534 012677 163300          MOV (SP)+,@SWR          ;;RESTORE SWR TO ORIGINAL VALUE :DPM001
015540 000407          BR 2000$          ;;BRANCH OVER TIMEOUT SECTION :DPM001
015542 022626          1999$: CMP (SP)+,(SP)+          ;;CLEAN UP THE STACK AFTER TIMEOUT :DPM001
015544 012737 177777 015674          MOV #-1,CPSAVE          ;;MOVE -1 TO CPSAVE - NO CPU ERR REG :DPM001
015552 005237 015562          INC 2000$+2          ;;INCREMENT 1ST TIME THROUGH LOCATION :DPM001
015556 000403          BR 910$          ;;BRANCH OVER 1ST TIME THROUGH TEST :DPM001
015560 005227 177777          2000$: INC #-1          ;;TEST FOR 1ST TIME THROUGH :DPM001
015564 001002          BNE 2001$          ;;BRANCH IF NOT :DPM001
015566 012637 000004          910$: MOV (SP)+,4          ;;RESTORE LOCATION 4 :DPM001
015572 105737 001003          2001$: TSTB $ERFLG          ;;HAS AN ERROR OCCURRED?
015576 001412          BEQ $SVLAD          ;;BR IF NO
015600 032777 001000 163232          BIT #BIT09,@SWR          ;;LOOP ON ERROR?
015606 001404          BEQ 4$          ;;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  MOV  $TSTNM,$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  $ESCAPE       ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
015652 112737 000001      001015  MCVB #1,$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 ;DPM001
```

3670

```
3672
3673
3674
3675
3676
3677 015676 112737 000001 016142 $ATY1:  MOVB  #1,$FFLG      ;TO REPORT FATAL ERROR
3678 015704 112737 000001 016140 $ATY3:  MOVB  #1,$MFLG      ;TO TYPE A MESSAGE
3679 015712 000403
3680 015714 112737 000001 016142 $ATY4:  MOVB  #1,$FFLG      ;TO ONLY REPORT FATAL ERROR
3681 015722
3682 015722 010046
3683 015724 010146
3684 015726 105737 016140
3685 015732 001450
3686 015734 122737 000001 001106
3687 015742 001031
3688 015744 132737 000100 001107
3689 015752 001425
3690 015754 017600 000004
3691 015760 062766 000002 000004
3692 015766 005737 001066 1$:
3693 015772 001375
3694 015774 010037 001102
3695 016000 105720 2$:
3696 016002 001376
3697 016004 163700 001102
3698 016010 006200
3699 016012 010037 001104
3700 016016 012737 000004 001066
3701 016024 000413
3702 016026 017637 000004 016052 3$:
3703 016034 062766 000002 000004
3704 016042 013746 177776
3705 016046 004737 016144
3706 016052 000000 4$:
3707 016054 5$:
3708 016054 105737 016142 10$:
3709 016060 001413
3710 016062 005737 001106
3711 016066 001410
3712 016070 005737 001066 11$:
3713 016074 001375
3714 016076 017637 000004 001070
3715 016104 005237 001066
3716 016110 062766 000002 000004 12$:
3717 016116 105037 016142
3718 016122 105037 016141
3719 016126 105037 016140
3720 016132 012601
3721 016134 012600
3722 016136 000207
3723 016140 000
3724 016141 000
3725 016142 000
3726
3727
3728 000200
```

```
*****
:SBTTL  APT COMMUNICATIONS ROUTINE
*****
MOV      R0,-(SP)      ;PUSH R0 ON STACK
MOV      R1,-(SP)      ;PUSH R1 ON STACK
TSTB    $MFLG          ;SHOULD TYPE A MESSAGE?
BEQ      5$            ;IF NOT: BR
CMPB    #APTENV,$ENV   ;OPERATING UNDER APT?
BNE      3$            ;IF NOT: BR
BITB    #APTPOOL,$ENVM ;SHOULD SPOOL MESSAGE?
BEQ      3$            ;IF NOT: BR
MOV      @4(SP),R0     ;GET MESSAGE ADDRESS
ADD      #2,4(SP)      ;BUMP RETURN ADDRESS
TST      $MSGTYPE      ;SEE IF DONE W/ LAST XMISSION?
BNE      1$            ;IF NOT: WAIT
MOV      R0,$MSGAD     ;PUT ADDRESS IN MAILBOX
TSTB    (R0)+          ;FIND END OF MESSAGE
BNE      2$            ;SUB START OF MESSAGE
SUB      $MSGAD,R0     ;GET MESSAGE LENGTH IN WORDS
ASR      R0            ;PUT LENGTH IN MAILBOX
MOV      R0,$MSGGLT    ;TELL APT TO TAKE MESSAGE
BR       5$
MOV      @4(SP),4$     ;PUT MSG ADDR IN JSR LINKAGE
ADD      #2,4(SP)      ;BUMP RETURN ADDRESS
MOV      177776,-(SP)  ;PUSH 177776 ON STACK
JSR     PC,$TYPE       ;CALL TYPE MACRO
WORD    0
TSTB    $FFLG          ;SHOULD REPORT FATAL ERROR?
BEQ      12$           ;IF NOT: BR
TST     $ENV           ;RUNNING UNDER APT?
BEQ      12$           ;IF NOT: BR
TST     $MSGTYPE       ;FINISHED LAST MESSAGE?
BNE      11$           ;IF NOT: WAIT
MOV      @4(SP),$FATAL ;GET ERROR #
INC      $MSGTYPE      ;TELL APT TO TAKE ERROR
ADD      #2,4(SP)      ;BUMP RETURN ADDRESS
CLRB    $FFLG          ;CLEAR FATAL FLAG
CLRB    $LFLG          ;CLEAR LOG FLAG
CLRB    $MFLG          ;CLEAR MESSAGE FLAG
MOV      (SP)+,R1      ;POP STACK INTO R1
MOV      (SP)+,R0      ;POP STACK INTO R1
RTS     PC             ;RETURN
$MFLG:  .BYTE 0
$LFLG:  .BYTE 0
$FFLG:  .BYTE 0
        .EVEN
APTSIZE=200
```

3729 000001  
3730 000100  
3731 000040  
3732

APTENV=001  
APTSPool=100  
APTC SUP=040

3734  
3735

```

.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$: MOV R0,-(SP) ;;SAVE R0
016160 017600 000002 MOV @2(SP),R0 ;;GET ADDRESS OF ASCIZ STRING
016164 122737 000001 001106 CMPB #APTENV,$ENV ;;RUNNING IN APT MODE
016172 001011 BNE 62$ ;;NO,GO CHECK FOR APT CONSOLE
016174 132737 000100 001107 BITB #APTSPool,$ENVM ;;SPOOL MESSAGE TO APT
016202 001405 BEQ 62$ ;;NO,GO CHECK FOR CONSOLE
016204 010037 016214 MOV R0,61$ ;;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$: BITB #APTCSUP,$ENVM ;;APT CONSOLE SUPPRESSED
016224 001003 BNE 60$ ;;YES,SKIP TYPE OUT
016226 112046 2$: MOVB (R0)+,-(SP) ;;PUSH CHARACTER TO BE TYPED ONTO STACK
016230 001005 BNE 4$ ;;BR IF IT ISN'T THE TERMINATOR
016232 005726 TST (SP)+ ;;IF TERMINATOR POP IT OFF THE STACK
016234 012600 60$: MOV (SP)+,R0 ;;RESTORE R0
016236 062716 000002 3$: ADD #2,(SP) ;;ADJUST RETURN PC
016242 000002 RTI ;;RETURN
016244 122716 000011 4$: CMPB #HT,(SP) ;;BRANCH IF <HT>
016250 001430 BEQ 8$
016252 122716 000200 CMPB #CRLF,(SP) ;;BRANCH IF NOT <CRLF>
016256 001006 BNE 5$
016260 005726 TST (SP)+ ;;POP <CR><LF> EQUIV
016262 104401 TYPE ;;TYPE A CR AND LF
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$:   MOVB   #' , (SP)      ;;REPLACE TAB WITH SPACE
016336 004737 016356      9$:   JSR    PC,$TYPEC      ;;TYPE A SPACE
016342 132737 000007 016504  BITB   #7,$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   @ $TKS          ;;CHAR IN KYBD BUFFER?
016362 100022          BPL    10$            ;;BR IF NOT
016364 017746 162456      MOV    @ $TKB,-(SP)     ;;GET CHAR
016370 042716 177600      BIC    #177600,(SP)    ;;STRIP EXTRANEIOUS BITS
016374 122716 000023      CMPB   # $XOFF,(SP)    ;;WAS CHAR XOFF
016400 001012          BNE    102$           ;;BR IF NOT
016402          101$:
016402 105777 162436      TSTB   @ $TKS          ;;WAIT FOR CHAR
016406 100375          BPL    101$           ;;MJD001
016410 117716 162432      MOVB   @ $TKB,(SP)     ;;GET CHAR
016414 042716 177600      BIC    #177600,(SP)    ;;STRIP IT
016420 122716 000021      CMPB   # $XON,(SP)    ;;WAS IT XON?
016424 001366          BNE    101$           ;;BR IF NOT
016426          102$:
016426 005726          TST    (SP)+          ;;FIX STACK
016430          10$:
016430 105777 162414      TSTB   @ $TPS          ;;WAIT UNTIL PRINTER IS READY
016434 100375          BPL    10$            ;;MJD001
016436 126627 000002 000021  CMPB   2(SP),# $XON     ;;IS CHARACTER A RANDOM XON?
016444 001420          BEQ    $TYPEX         ;;BRANCH IF YES
016446 116677 000002 162376  MOVB   2(SP),@ $TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
016454 122766 000015 000002  CMPB   #CR,2(SP)        ;;IS CHARACTER A CARRIAGE RETURN?
016462 001003          BNE    1$            ;;BRANCH IF NO
016464 105037 016504      CLRB   $CHARCNT       ;;YES--CLEAR CHARACTER COUNT
016470 000406          BR     $TYPEX         ;;EXIT
016472 122766 000012 000002  1$:   CMPB   #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          $TYPEX: RTS PC
3736          .SBTTL BINARY TO OCTAL (ASCII) AND TYPE
          *****
          *THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
          *OCTAL (ASCII) NUMBER AND TYPE IT.
          *$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
          *CALL:
          *   MOV    NUM,-(SP)      ;;NUMBER TO BE TYPED
          *   TYPOS          ;;CALL FOR TYPEOUT
          *   .BYTE  N              ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
          *   .BYTE  M              ;;M=1 OR 0
          *                                   ;;1=TYPE LEADING ZEROS
          *                                   ;;0=SUPPRESS LEADING ZEROS
          *$TYPON---ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
          *$TYPOS OR $TYPOC
          *CALL:
          *   MOV    NUM,-(SP)      ;;NUMBER TO BE TYPED
          *   TYPON          ;;CALL FOR TYPEOUT
          *$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
```

```

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



3739  
3740

```

.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 $CKSWR: CMP #SWREG,SWR ;;IS THE SOFT-SWR SELECTED?
016744 001074 BNE 15$ ;;BRANCH IF NO
016746 105777 162072 TSTB @STKS ;;CHAR THERE?
016752 100071 BPL 15$ ;;IF NO, DON'T WAIT AROUND
016754 117746 162066 MOVB @STKB,-(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 ,SCNTLG ;;ECHO THE CONTROL-G (^G)
017006 104401 017500 $GTSWR: TYPE ,SMSWR ;;TYPE CURRENT CONTENTS
017012 013746 000176 MOV SWREG,-(SP) ;;SAVE SWREG FOR TYPEOUT
017016 104402 TYPOC ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
017020 104401 017511 TYPE ,SMNEW ;;PROMPT FOR NEW SWR
017024 005046 19$: CLR -(SP) ;;CLEAR COUNTER
017026 005046 CLR -(SP) ;;THE NEW SWR
017030 105777 162010 7$: TSTB @STKS ;;CHAR THERE?
017034 100375 BPL 7$ ;;IF NOT TRY AGAIN
017036 117746 162004 MOVB @STKB,-(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 ,SCNTLU ;;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),@SWR ;;SAVE NEW SWR
017110 062706 000006 11$: ADD #6,SP ;;CLEAR UP STACK
017114 104401 001063 14$: TYPE ,SCRLF ;;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,@STKS ;;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.

```

```

017200 005266 000002      17$:  INC      2(SP)      ;;KEEP COUNT OF CHAR
017204 056616 177776      BIS      -2(SP), (SP)  ;;SET IN NEW CHAR
017210 000707              BR       7$           ;;GET THE NEXT ONE
017212 104401 001062      18$:  TYPE     $QUES     ;;TYPE ?<CR><LF>
017216 000720              BR       20$         ;;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
$RDCHR: MOV      (SP), -(SP)  ;;PUSH DOWN THE PC
017220 011646 000004 000002  MOV      4(SP), 2(SP)    ;;SAVE THE PS
017222 016666 000004 000002  1$:  TSTB    @STKS        ;;WAIT FOR
017230 105777 161610              BPL      1$           ;;A CHARACTER
017234 100375              MOVB    @STKB, 4(SP)    ;;READ THE TTY
017236 117766 161604 000004  BIC      #^C<177>, 4(SP) ;;GET RID OF JUNK IF ANY
017244 042766 177600 000004  CMP      4(SP), #23     ;;IS IT A CONTROL-S?
017252 026627 000004 000023  BNE      3$           ;;BRANCH IF NO
017260 001013              2$:  TSTB    @STKS        ;;WAIT FOR A CHARACTER
017262 105777 161556              BPL      2$           ;;LOOP UNTIL ITS THERE
017266 100375              MOVB    @STKB, -(SP)   ;;GET CHARACTER
017270 117746 161552              BIC      #^C177, (SP)  ;;MAKE IT 7-BIT ASCII
017274 042716 177600              CMP      (SP)+, #21    ;;IS IT A CONTROL-Q?
017300 022627 000021              BNE      2$           ;;IF NOT DISCARD IT
017304 001366              BR       1$           ;;YES, RESUME
017306 000750              3$:  CMP      4(SP), #XON  ;;IS IT A RANDOM XON?
017310 026627 000004 000021  BEQ      1$           ;;BRANCH IF YES
017316 001744              CMP      4(SP), #140  ;;IS IT UPPER CASE?
017320 026627 000004 000140  BLT      4$           ;;BRANCH IF YES
017326 002407              CMP      4(SP), #175  ;;IS IT A SPECIAL CHAR?
017330 026627 000004 000175  BGT      4$           ;;BRANCH IF YES
017336 003003              BIC      #40, 4(SP)   ;;MAKE IT UPPER CASE
017340 042766 000040 000004  4$:  RTI              ;;GO BACK TO USER
017346 000002
;*****
;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
$RDLIN: MOV      R3, -(SP)  ;;SAVE R3
017350 010346 000004 000002  1$:  MOV      #$TTYIN, R3  ;;GET ADDRESS
017352 012703 017456 000004  2$:  CMP      #$TTYIN+8., R3 ;;BUFFER FULL?
017356 022703 017466              BLOS    4$           ;;BR IF YES
017362 101405              RDCHR   ;;GO READ ONE CHARACTER FROM THE TTY
017364 104407              MOVB    (SP)+, (R3)   ;;GET CHARACTER
017366 112613              CMPB   #177, (R3)    ;;IS IT A RUBOUT
017370 122713 000177 10$:  BNE      3$           ;;SKIP IF NOT
017374 001003              4$:  TYPE     $QUES     ;;TYPE A '?'
017376 104401 001062              BR       1$           ;;CLEAR THE BUFFER AND LOOP
017402 000763              3$:  MOVB    (R3), 9$    ;;ECHO THE CHARACTER
017404 111337 017454              TYPE     9$
017410 104401 017454              CMPB   #15, (R3)+    ;;CHECK FOR RETURN
017414 122723 000015              BNE      2$           ;;LOOP IF NOT RETURN
017420 001356              CLRB   -1(R3)        ;;CLEAR RETURN (THE 15)
017422 105063 177777

```

:RAN001  
:RAN001

TTY INPUT ROUTINE

017426	104401	001064			TYPE	,\$LF	::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			9\$:	.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	/^U/<15><12>	::CONTROL 'U'
017473	136	107	015	\$CNTLG:	.ASCIZ	/^G/<15><12>	::CONTROL 'G'
017500	015	012	123	\$MSWR:	.ASCIZ	<15><12>/SWR = /	
017511	040	040	116	\$MNEW:	.ASCIZ	/ NEW = /	

3741

3743  
3744  
3745

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

017544 011646  
017546 016666 000004 000002  
017554 000002

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

3746

.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.

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

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

```
$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

-----

```
$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
        $CKSWR ;;CALL=CKSWR    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
```

```

3748 .SBTTL ECHO TEST
3749 ;:*****
3750 ;*THIS ROUTINE WILL ECHO ANY CHARACTER TYPED
3751 ;*AT THE CONSOLE
3752 ;*THE TEST IS HALTED BY TYPING A CONTROL-C
3753 ;*TEST CAN BE RESTARTED AFTER HALTING BY JUST CONTINUING
3754 ;:*****
3755 ECHO:
3756 017600 000005 RESET ;CLEAR EVERYTHING
3757 017602 112777 000052 163052 MOVB #'*,@CTBUF ;TRANSMIT PROMPT '*'
3758 017610 105777 163040 2$: TSTB @CTCSR ;WAIT FOR INPUT
3759 017614 100375 BPL 2$
3760 017616 117777 163034 163036 MOVB @CRBUF,@CTBUF ;ECHO INPUT
3761 017624 017700 163026 MOV @CRBUF,R0 ;STORE INPUT
3762 017630 100023 BPL 5$ ;BR IF 'ERROR' NOT SET
3763 017632 052701 010000 BIS #BIT12,R1 ;SET PARITY ERROR TEST MASK
3764 017636 030100 BIT R1,R0 ;CHECK FOR PARITY ERROR FLAG
3765 017640 001403 BEQ 3$ ;BR IF NOT SET
3766 017642 004737 017724 JSR PC,MSG ;REPORT PARITY ERROR
3767 017646 017752 MPAR
3768 017650 006301 3$: ASL R1 ;SHIFT MASK TO TEST 'FR' FLAG
3769 017652 030100 BIT R1,R0 ;TEST FOR FRAMING ERROR FLAG
3770 017654 001403 BEQ 4$ ;BR IF NOT SET
3771 017656 004737 017724 JSR PC,MSG ;REPSORT FRAMING ERROR
3772 017662 017763 MFR
3773 017664 006301 4$: ASL R1 ;SHIFT MASK TO TEST 'OR' FLAG
3774 017666 030100 BIT R1,R0 ;TEST FOR OVERFLOW ERROR
3775 017670 001403 BEQ 5$ ;BR IF NOT SET
3776 017672 004737 017724 JSR PC,MSG ;REPORT OVERFLOW ERROR
3777 017676 017775 MOR
3778 017700 042700 000200 5$: BIC #BIT7,R0 ;CLEAR ANY PARITY BIT
3779 017704 022700 000003 CMP #3,R0 ;WAS INPUT CONTROL-C
3780 017710 001337 BNE 2$ ;BR IF NOT
3781 017712 004737 017724 JSR PC,MSG ;REPORT PROGRAM STOP
3782 017716 020010 MSTOP
3783 017720 000000 HALT ;END OF TEST HALT
3784 017722 000726 BR ECHO ;AFTER END OF TEST HALT
3785 ; PRESS CONTINUE TO RESTART ECHO TEST
3786
3787 017724 013600 MSG: MOV @(SP)+,R0 ;PICK UP MESSAGE POINTER
3788 017726 062746 000002 ADD #2,-(SP) ;ADJUST RETURN PC
3789 017732 105777 162722 WAIT: TSTB @CTCSR ;WAIT FOR XMIT DONE
3790 017736 100375 BPL WAIT
3791 017740 112077 162716 MOVB (R0)+,@CTBUF ;SEND CHARACTER
3792 017744 105710 TSTB (R0) ;IS THIS END OF MESSAGE?
3793 017746 001371 BNE WAIT ;BR IF NOT
3794 017750 000207 RTS PC ;RETURN
3795
3796 017752 015 012 120 MPAR: .ASCIZ <CR><LF>/PARITY/
3797 017763 015 012 106 MFR: .ASCIZ <CR><LF>/FRAMING/
3798 017775 015 012 117 MOR: .ASCIZ <CR><LF>/OVERFLOW/
3799 020010 015 012 123 MSTOP: .ASCIZ <CR><LF>/STOP/

```

3801  
3802  
3803  
3804  
3805  
3806  
3807  
3808  
3809  
3810  
3811  
3812  
3813  
3814  
3815 020020 012701 000040  
3816 020024 012700 000040  
3817 020030 105777 162624  
3818 020034 100375  
3819 020036 010177 162620  
3820 020042 105201  
3821 020044 005300  
3822 020046 001370  
3823 020050 004737 017724  
3824  
3825 020054 001063  
3826 020056 105777 162572  
3827 020062 100404  
3828 020064 032701 000200  
3829 020070 001353  
3830 020072 000754  
3831  
3832 020074 005077 162556  
3833 020100 000000  
3834 020102 000746

```
.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    @CTCSR           ;WAIT FOR DONE
        BPL     2$
        MOV     R1,@CTBUF        ;TRANSMIT A CHARACTER
        INCB   R1               ;INCREMENT CHARACTER CODE
        DEC    R0               ;DECREMENT CHAR COUNT
        BNE    2$              ;BR IF LINE NOT COMPLETE
        JSR    PC,MSG           ;SSUE CR,LINE FEED
        $CRLF
        TSTB   @CRCSR           ;ANY CHARACTER RECEIVED?
        BMI    3$              ;BR IF YES
        BIT    #BIT7,R1         ;FINISHED ONE PASS OF WRITABLE CHARACTERS?
        BNE    OUTTST          ;BR IF YES
        BR     1$              ;IF NOT WRITE NEXT LINE
3$:     CLR     @CRBUF          ;CLEAR RECEIVER
        HALT
        BR     OUTTST          ;RESTART TEST IF CONTINUED
```

```

3836
3837
3838 020104      103      101      116  EM1:  .ASCIZ  /CAN NOT ACCESS TCSR/
3839 020130      103      101      116  EM2:  .ASCIZ  /CAN NOT ACCESS TBUF/
3840 020154      124      103      123  EM3:  .ASCIZ  /TCSR DONE NOT CLEARED WITH TBUF FULL/
3841 020221      124      103      123  EM4:  .ASCIZ  /TCSR DONE NOT SET/
3842 020243      124      103      123  EM5:  .ASCIZ  /TCSR DONE NOT SET WITH RESET/
3843 020300      103      101      116  EM6:  .ASCIZ  /CAN NOT ACCESS RCSR/
3844 020324      103      101      116  EM7:  .ASCIZ  /CAN NOT ACCESS RBUF/
3845 020350      103      101      116  EM10: .ASCIZ  /CAN NOT ACCESS LKS/
3846 020373      102      111      124  EM11: .ASCIZ  /BIT0 OF TCSR NOT CLEAR AFTER RESET/
3847 020436      103      101      116  EM12: .ASCIZ  /CAN NOT SET BIT0 OF TCSR/
3848 020467      103      101      116  EM13: .ASCIZ  /CAN NOT CLEAR BIT0 OF TCSR/
3849 020522      122      105      123  EM14: .ASCIZ  /RESET DID NOT CLEAR BIT0 OF TCSR/
3850 020563      102      111      124  EM15: .ASCIZ  /BIT2 OF TCSR NOT CLEAR AFTER RESET/
3851 020626      103      101      116  EM16: .ASCIZ  /CAN NOT SET BIT2 OF TCSR/
3852 020657      103      101      116  EM17: .ASCIZ  /CAN NOT CLEAR BIT2 OF TCSR/
3853 020712      122      105      123  EM20: .ASCIZ  /RESET DID NOT CLEAR BIT2 OF TCSR/
3854 020753      102      111      124  EM21: .ASCIZ  /BIT6 OF TCSR NOT CLEAR AFTER RESET/
3855 021016      130      115      111  EM22: .ASCIZ  /XMIT INTERRUPT AT PRIORITY 7/
3856 021053      103      101      116  EM23: .ASCIZ  /CAN NOT SET BIT6 OF TCSR/
3857 021104      103      101      116  EM24: .ASCIZ  /CAN NOT CLEAR BIT6 OF TCSR/
3858 021137      122      105      123  EM25: .ASCIZ  /RESET DID NOT CLEAR BIT6 OF TCSR/
3859 021200      102      111      124  EM26: .ASCIZ  /BIT6 OF RCSR NOT CLEAR AFTER RESET/
3860 021243      122      103      126  EM27: .ASCIZ  /RCVR INTERRUPT WITH PRIORITY 7/
3861 021302      103      101      116  EM30: .ASCIZ  /CAN NOT SET BIT6 OF RCSR/
3862 021333      103      101      116  EM31: .ASCIZ  /CAN NOT CLEAR BIT6 OF RCSR/
3863 021366      103      101      116  EM32: .ASCIZ  /CAN NOT CLEAR BIT6 OF RCSR WITH RESET/
3864 021434      102      111      124  EM33: .ASCIZ  /BIT6 OF LKS NOT CLEAR AFTER RESET/
3865 021476      114      113      123  EM34: .ASCIZ  /LKS INTERRUPT WITH PRIORITY 7/
3866 021534      103      101      116  EM35: .ASCIZ  /CAN NOT SET BIT6 OF LKS/
3867 021564      103      101      116  EM36: .ASCIZ  /CAN NOT CLEAR BIT6 OF LKS/
3868 021616      122      105      123  EM37: .ASCIZ  /RESET DID NOT CLEAR BIT6 OF LKS/
3869 021656      104      125      101  EM40: .ASCIZ  /DUAL ADDRESSING ERROR/
3870 021704      102      111      124  EM41: .ASCIZ  /BIT6 OF LKS NOT SET AFTER RESET/
3871 021744      103      101      116  EM42: .ASCIZ  /CAN NOT CLEAR BIT7 OF LKS/
3872 021776      102      111      124  EM43: .ASCIZ  /BIT7 OF LKS DOES NOT SET/
3873 022027      122      124      103  EM44: .ASCIZ  /RTC INTERRUPT AT PRIORITY 7/
3874 022063      122      124      103  EM45: .ASCIZ  /RTC INTERRUPTS WHEN DISABLED/
3875 022120      122      124      103  EM47:
3876 022120      122      124      103  EM46: .ASCIZ  /RTC INTERRUPT DID NOT OCCUR/
3877 022154      122      124      103  EM50: .ASCIZ  /RTC DOUBLE INTERRUPT/
3878 022201      122      105      123  EM51: .ASCIZ  /RESET DID NOT INTERRUPT/
3879 022231      122      124      103  EM52: .ASCIZ  /RTC INTERRUPT DID NOT CLEAR WITH BIT7 OF LKS/
3880 022306      103      114      117  EM53: .ASCIZ  /CLOCK REPEATABILITY/
3881 022332      130      115      111  EM54: .ASCIZ  /XMIT INTERRUPTS WHEN DISABLED/
3882 022370      130      115      111  EM56: .ASCIZ  /XMIT INTERRUPTS AT PRIORITY 7/
3883 022426      130      115      111  EM57: .ASCIZ  /XMIT INTERRUPTS WITH ENABLE CLEAR/
3884 022470      130      115      111  EM55:
3885 022470      130      115      111  EM60: .ASCIZ  /XMIT DID NOT INTERRUPT/
3886 022517      130      115      111  EM61: .ASCIZ  /XMIT RE-INTERRUPTED/
3887 022543      114      117      101  EM62: .ASCIZ  /LOADING TBUF DID NOT CLEAR INTERRUPT/
3888 022610      122      103      126  EM63: .ASCIZ  /RCVR ACTIVE NOT SET/
3889 022634      122      103      126  EM64: .ASCIZ  /RCVR DONE NEVER SET/
3890 022660      122      103      126  EM65: .ASCIZ  /RCVR ACTIVE NOT CLEARED WITH DONE SET/
3891 022726      122      105      123  EM66: .ASCIZ  /RESET DID NOT CLEAR RCVR DONE/
3892 022764      122      104      122  EM67: .ASCIZ  /RDR ENABLE DID NOT CLEAR RCVR DONE/
  
```

```

3893 023027      122      105      101  EM70:  .ASCIZ  /READING RBUF DID NOT CLEAR RCVR DONE/
3894 023074      122      103      126  EM71:  .ASCIZ  /RCVR INTERRUPTS WITH ENABLE CLEAR/
3895 023136      122      103      126  EM73:  .ASCIZ  /RCVR INTERRUPTS AT PRIORITY 7/
3896 023174      122      103      126  EM74:  .ASCIZ  /RCVR INT RQST PASSED WITH ENABLE CLEAR/
3897 023243
3898 023243      122      103      126  EM75:  .ASCIZ  /RCVR DID NOT INTERRUPT/
3899 023272      122      103      126  EM76:  .ASCIZ  /RCVR RE-INTERRUPTED/
3900 023316      122      105      101  EM77:  .ASCIZ  /READING RBUF DID NOT CLEAR INTERRUPT/
3901 023363      122      105      123  EM100: .ASCIZ  /RESET DID NOT CLEAR RCVR INTERRUPT/
3902 023426      047      117      122  EM101: .ASCIZ  /'OR' FLAG DID NOT SET/
3903 023454      047      105      122  EM102: .ASCIZ  /'ERROR' NOT SET WITH 'OR' FLAG/
3904 023513      102      122      105  EM103: .ASCIZ  /BREAK DID NOT XMIT ALL ZEROES/
3905 023551      102      122      105  EM104: .ASCIZ  /BREAK DID NOT SET 'FR' ERROR/
3906 023606      104      101      124  EM105: .ASCIZ  /DATA COMPARE ERROR/
3907 023631      114      117      117  EM106: .ASCIZ  /LOOP-BACK DATA COMPARE ERROR/
3908 023666      124      111      115  EM107: .ASCIZ  /TIMEOUT IN EXERCISER TEST/
3909 023720      111      116      103  EM110: .ASCIZ  /INCORRECT RECEIVE COUNT/
3910 023750      104      101      124  EM111: .ASCIZ  /DATA COMPARE ERROR IN EXERCISER/
3911 024010      124      122      101  EM112: .ASCIZ  /TRAP CATCHER/
3912 024025      116      117      040  EM113: .ASCIZ  /NO CLK INTERRUPT IN EXERCISER/
3913 024063      042      105      122  EM114: .ASCIZ  /'ERROR' NOT SET WITH 'FR' FLAG/
3914 024122      122      103      126  EM115: .ASCIZ  /RCV ACTIVE NOT CLEAR WITH INIT/
3915 024161      122      103      126  EM116: .ASCIZ  /RCV ACTIVE WITHOUT 'START' BIT/
3916 024220      122      104      122  EM117: .ASCIZ  /RDR ENABLE NOT CLEAR WITH RCV ACTIVE/
3917
3918 024265      124      105      123  DH1:   .ASCIZ  /TEST#  ERROR#  TCSR/
3919 024312      124      105      123  DH2:   .ASCIZ  /TEST#  ERR PC  TBUF/
3920 024337      124      105      123  DH6:   .ASCIZ  /TEST#  ERR PC  RCSR/
3921 024364      124      105      123  DH7:   .ASCIZ  /TEST#  ERR PC  RBUF/
3922 024411      124      105      123  DH10:  .ASCIZ  /TEST#  ERR PC  LKS/
3923 024435      124      105      123  DH40:  .ASCIZ  /TEST#  ERR PC  GOOD  BAD/
3924 024471      124      105      123  DH53:  .ASCIZ  /TEST#  ERR PC  LKS  CNT1  CNT2/
3925 024536      124      105      123  DH103: .ASCIZ  /TEST#  ERR PC  RCSR  DATA/
3926 024573      124      105      123  DH105: .ASCIZ  /TEST#  ERR PC  RCSR  GOOD  BAD/
3927 024637      124      105      123  DH110: .ASCIZ  /TEST#  ERR PC  RCSR  TRANS  RCV/
3928 024703      124      105      123  DH112: .ASCIZ  /TEST#  ERR PC  RCSR  OLDPC  TRAP  ADR/
3929
3930 024754      015      012      103  M1:    .ASCIZ  <CR><LF>?CZDLDH0 DL11-W 11/44 MFM SLU?
3931
3932 025014      015      012
3933 025016      040      040      040  M2:    .ASCII  <CR><LF>
3934
3935
3936 025046      001072  001016  002640  DT1:   .WORD   $TESTN,$ERRPC,$TCSR,0
3937 025056      001072  001016  002642  DT2:   .WORD   $TESTN,$ERRPC,$TBUF,0
3938 025066      001072  001016  002634  DT6:   .WORD   $TESTN,$ERRPC,$RCSR,0
3939 025076      001072  001016  002636  DT7:   .WORD   $TESTN,$ERRPC,$RBUF,0
3940 025106      001072  001016  002674  DT10:  .WORD   $TESTN,$ERRPC,$LKS,0
3941 025116      001072  001016  001020  DT40:  .WORD   $TESTN,$ERRPC,$GDADR,$BDADR,0
3942 025130      001072  001016  002674  DT53:  .WORD   $TESTN,$ERRPC,$LKS,$FIRST,$SECND,0
3943 025144      001072  001016  002634  DT103: .WORD   $TESTN,$ERRPC,$RCSR,$BDDAT,0
3944 025156      001072  001016  002634  DT105: .WORD   $TESTN,$ERRPC,$RCSR,$GDDAT,$BDDAT,0
3945 025172      001072  001016  002634  DT110: .WORD   $TESTN,$ERRPC,$RCSR,$XMITCNT,$RCVCNT,0
3946 025206      001072  001016  002634  DT112: .WORD   $TESTN,$ERRPC,$RCSR,$OLDPC,$BDVECT,0
3947 025222      000000
3948
3949
    ENDADR: .WORD   0 ;END ADDRESS FOR CHECKSUM AND SAVE AREA
                ;OF WRAP AROUND CONSOLE TEST
    
```



3950

000001

.END

ABASE = 176500	BGNADD= 003014	DH105 024573	EM26 021200	GETB 014126
ACDW1 = 000000	BIT0 = 000001	DH110 024637	EM27 021243	GETLIN 014062
ACDW2 = 000000	BIT00 = 000001	DH112 024703	EM3 020154	GOAGIN 014432
ACPUOP= 000000	BIT01 = 000002	DH2 024312	EM30 021302	GTSWR = 104405
ADDW0 = 000000	BIT02 = 000004	DH40 024435	EM31 021333	HT = 000011
ADDW1 = 000000	BIT03 = 000010	DH53 024471	EM32 021366	ID 004620
ADDW10= 000000	BIT04 = 000020	DH6 024337	EM33 021434	INIT 003462
ADDW11= 000000	BIT05 = 000040	DH7 024364	EM34 021476	IOTVEC= 000020
ADDW12= 000000	BIT06 = 000100	DISPLA 001042	EM35 021534	JIMSTK 002432
ADDW13= 000000	BIT07 = 000200	DISPRE 000174	EM36 021564	LF = 000012
ADDW14= 000000	BIT08 = 000400	DSWR = 177570	EM37 021616	LKS 002674
ADDW15= 000000	BIT09 = 001000	DT1 025046	EM4 020221	LOC1 002340
ADDW2 = 000000	BIT1 = 000002	DT10 025106	EM40 021656	LOC2 002342
ADDW3 = 000000	BIT10 = 002000	DT103 025144	EM41 021704	MANL 003502
ADDW4 = 000000	BIT11 = 004000	DT105 025156	EM42 021744	MFPT = 000007
ADDW5 = 000000	BIT12 = 010000	DT110 025172	EM43 021776	MFR 017763
ADDW6 = 000000	BIT13 = 020000	DT112 025206	EM44 022027	MOR 017775
ADDW7 = 000000	BIT14 = 040000	DT2 025056	EM45 022063	MPAR 017752
ADDW8 = 000000	BIT15 = 100000	DT40 025116	EM46 022120	MSG 017724
ADDW9 = 000000	BIT2 = 000004	DT53 025130	EM47 022120	MSTOP 020010
ADEVCT= 000000	BIT3 = 000010	DT6 025066	EM5 020243	M1 024754
ADEVVM = 000000	BIT4 = 000020	DT7 025076	EM50 022154	M2 025014
ADR 004132	BIT5 = 000040	DVADT 003664	EM51 022201	M2A 025016
ADRTBL 002702	BIT6 = 000100	ECHO 017600	EM52 022231	NOEOP 014316
AENV = 000000	BIT7 = 000200	EMTVEC= 000030	EM53 022306	OLDPC 002620
AENVM = 000000	BIT8 = 000400	EM1 020104	EM54 022332	OLDSUM 002436
AFATAL= 000000	BIT9 = 001000	EM10 020350	EM55 022470	OUTTST 020020
AMADR1= 000000	BPT = 000003	EM100 023363	EM56 022370	PFECDF 015206
AMADR2= 000000	BPTVEC= 000014	EM101 023426	EM57 022426	PFECDH 015146
AMADR3= 000000	BUF 002454	EM102 023454	EM6 020300	PFECTD 015176
AMADR4= 000000	CATCH 014520	EM103 023513	EM60 022470	PFECM 015106
AMAMS1= 000000	CHKSUM 014010	EM104 023551	EM61 022517	PFECWS 015076
AMAMS2= 000000	CKSWR = 104406	EM105 023606	EM62 022543	PIRQ = 177772
AMAMS3= 000000	CLK 013630	EM106 023631	EM63 022610	PIRQVE= 000240
AMAMS4= 000000	CLKCNT 002444	EM107 023666	EM64 022634	PROMPT 002566
AMSGAD= 000000	COMPARE 007330	EM11 020373	EM65 022660	PRO = 000000
AMSGLG= 000000	CNTLP 002564	EM110 023720	EM66 022726	PR1 = 000040
AMSGTY= 000000	COMP 013432	EM111 023750	EM67 022764	PR2 = 000100
AMTYP1= 000000	CONT 006246	EM112 024010	EM7 020324	PR3 = 000140
AMTYP2= 000000	CONT41 012354	EM113 024025	EM70 023027	PR4 = 000200
AMTYP3= 000000	CPSAVE 015674	EM114 024063	EM71 023074	PR5 = 000240
AMTYP4= 000000	CR = 000015	EM115 024122	EM72 023243	PR6 = 000300
APASS = 000000	CRBUF 002656	EM116 024161	EM73 023136	PR7 = 000340
APRIOR= 000000	CRCSR 002654	EM117 024220	EM74 023174	PS = 177776
APTCSU= 000040	CRLF = 000200	EM12 020436	EM75 023243	PSW = 177776
APTENV= 000001	CRPSW 002666	EM13 020467	EM76 023272	PUTLIN 014030
APTSIZ= 000200	CRVECT 002664	EM14 020522	EM77 023316	PWRVEC= 000024
APTSPO= 000100	CTBUF 002662	EM15 020563	ENDADD= 025222	RBUF 002636
APTSZD 003636	CTCSR 002660	EM16 020626	ENDADR 025222	RBUF58= 176502
ASWREG= 000000	CTPSW 002672	EM17 020657	ENDB7 004676	RCSR 002634
ATESTN= 000000	CTSTFL 002624	EM2 020130	ENDEV 014272	RCSR58= 176500
AUNIT = 000000	CTVECT 002670	EM20 020712	ENDMG 014416	RCV 013614
AUSWR = 000400	DDISP = 177570	EM21 020753	ENDSTK 002412	RCVCNT 002440
AVECT1= 000300	DEVADR 003014	EM22 021016	ERROR = 104000	RCVDON 010346
AVECT2= 000000	DH1 024265	EM23 021053	ERRVEC= 000004	RDCHR = 104407
BDVECT 002622	DH10 024411	EM24 021104	FIRST 002336	RDLIN = 104410
BEGIN 003774	DH103 024536	EM25 021137	FLAG44 003002	RESTTE 014236

RESVEC=	000010	TBUF	002642	TST5	004714	\$ERFLG	001003	\$PASS	001074
RPSW	002646	TBUF58=	176506	TST6	004760	\$ERMAX	001015	\$PASTM	000506
RSTRT	014472	TCLOCK	005536	TST7	005012	\$ERROR	014544	\$POWER	015364
RTCPSW	002700	TCONS	004020	TURBUF	003006	\$ERRPC	001016	\$PWRDN	015212
RTCVT	002676	TCSR	002640	TURCSR	003004	\$ERRTB	001146	\$PWRMG	015346
RVECT	002644	TCSR58=	176504	TUTBUF	003012	\$ERRTY	014726	\$PWRUP	015264
R6	=%000006	TIMER	014162	TUTCSR	003010	\$ERTTL	001012	\$QUES	001062
R7	=%000007	TKVEC =	000060	TVECT	002650	\$ESCAP	001060	\$RDCHR	017220
SAVEPS	002434	TMP1	002626	TYPE =	104401	\$ETABL	001106	\$RDLIN	017350
SAVETE	014202	TMP2	002630	TYPOC =	104402	\$ETEND	001146	\$RDSZ =	000010
SAVEO	002344	TMP3	002632	TYPON =	104404	\$FATAL	001070	\$RTNAD	014410
SAVLOC	002346	TOLER	007342	TYPOS =	104403	\$FFLG	016142	\$SAVR6	015362
SCOPE =	000004	TPSW	002652	VCTADR	003670	\$FILLC	001056	\$SCOPE	015374
SCPSW	002452	TPVEC =	000064	VCTTBL	002742	\$FILLS	001055	\$SETUP=	000137
SECND	002616	TRAPVE=	000034	VECT	004152	\$GDADR	001020	\$STUP =	177777
SETADR	004100	TRTVEC=	000014	WACTV	010230	\$GDDAT	001024	\$SVLAD	015624
SHIFT	003740	TSTDEV	004050	WAIT	017732	\$GET42	014366	\$SVPC =	000500
SIZE	003512	TSTDVM	003646	WDONE	010362	\$GTSWR	017006	\$SWR =	161000
SRPSW	002450	TST1	004172	WRAP	013640	\$HIBTS	000500	\$SWREG	001110
STACK =	001100	TST10	005044	WRPSW	014502	\$ICNT	001004	\$SWRMK=	000000
START	003046	TST11	005252	WT	010316	\$ILLUP	015356	\$TESTN	001072
STKLMT=	177774	TST12	005404	XCONT	013604	\$INTAG	001035	\$TKB	001046
STPSW	002446	TST13	005544	XMIT	013542	\$ITEMB	001014	\$TKS	001044
SWR	001040	TST14	005614	XMTCNT	002442	\$LF	001064	\$TN =	000047
SWREG	000176	TST15	005764	XRET	013610	\$LFLG	016141	\$TPB	001052
SWO =	000001	TST16	006154	\$APTHD	000500	\$LPADR	001006	\$TPFLG	001057
SWO0 =	000001	TST17	006262	\$ATYC	015722	\$LPERR	001010	\$TPS	001050
SWO1 =	000002	TST2	004244	\$ATY1	015676	\$MADR1	001120	\$TRAP	017522
SWO2 =	000004	TST20	006522	\$ATY3	015704	\$MADR2	001124	\$TRAP2	017544
SWO3 =	000010	TST21	006720	\$ATY4	015714	\$MADR3	001130	\$TRP =	000011
SWO4 =	000020	TST22	007054	\$AUTOB	001034	\$MADR4	001134	\$TRPAD	017556
SWO5 =	000040	TST23	007222	\$BASE	001142	\$MAIL	001066	\$STSM	000504
SWO6 =	000100	TST24	007410	\$BDADR	001022	\$MAMS1	001116	\$STSTM	001002
SWO7 =	000200	TST25	007520	\$BDDAT	001026	\$MAMS2	001122	\$TTYIN	017456
SWO8 =	000400	TST26	007646	\$CHARC	016504	\$MAMS3	001126	\$TYPE	016144
SWO9 =	001000	TST27	010012	\$CKSWR	016736	\$MAMS4	001132	\$TYPEC	016356
SW1 =	000002	TST3	004316	\$CMTAG	001000	\$MBADR	000502	\$TYPEX	016506
SW10 =	002000	TST30	010156	\$CM3 =	000000	\$MFLG	016140	\$TYPOC	016534
SW11 =	004000	TST31	010462	\$CNTLG	017473	\$MNEW	017511	\$TYPON	016550
SW12 =	010000	TST32	010554	\$CNTLU	017466	\$MSGAD	001102	\$TYPOS	016510
SW13 =	020000	TST33	010700	\$CPUOP	001114	\$MSGLG	001104	\$UNIT	001100
SW14 =	040000	TST34	011102	\$CRLF	001063	\$MSGTY	001066	\$UNITM	000510
SW15 =	100000	TST35	011304	\$DEVCT	001076	\$MSWR	017500	\$USWR	001112
SW2 =	000004	TST36	011512	\$DEVVM	001144	\$MTYP1	001117	\$VECT1	001136
SW3 =	000010	TST37	011656	\$DOAGN	014406	\$MTYP2	001123	\$VECT2	001140
SW4 =	000020	TST4	004466	\$ENDAD	014376	\$MTYP3	001127	\$XOFF =	000023
SW5 =	000040	TST40	012014	\$ENDCT	014356	\$MTYP4	001133	\$XON =	000021
SW6 =	000100	TST41	012200	\$ENULL	014412	\$NULL	001054	\$XTSTR	015406
SW7 =	000200	TST42	012414	\$ENV	001106	\$NWTST=	000001	\$GET4=	000000
SW8 =	000400	TST43	012576	\$ENVM	001107	\$OCNT	016732	\$OFILL	016733
SW9 =	001000	TST44	012744	\$EOP	014326	\$OMODE	016734	.\$ERRT	001146
TA	002610	TST45	013062	\$EOPCT	014350	\$OVER	015660	.\$X =	000500
TBITVE=	000014	TST46	013636						

. ABS. 025224 000  
 000000 001  
 ERRORS DETECTED: 0

VIRTUAL MEMORY USED: 40488 WORDS ( 159 PAGES)  
DYNAMIC MEMORY: 20034 WORDS ( 77 PAGES)  
ELAPSED TIME: 00:17:06  
CZDLDH.BIN,CZDLDH/CR/-SP/NL:TOC=CZDLDH.MLB/ML,CZDLDH.P11

SYMBOL	CROSS REFERENCE	VALUE	REFERENCES			
ABASE	=	176500	#17-1435	21-1482	21-1482	
ACDW1	=	000000	21-1482			
ACDW2	=	000000	21-1482			
ACPUOP	=	000000	21-1482	21-1482		
ADDW0	=	000000	21-1482			
ADDW1	=	000000	21-1482			
ADDW10	=	000000	21-1482			
ADDW11	=	000000	21-1482			
ADDW12	=	000000	21-1482			
ADDW13	=	000000	21-1482			
ADDW14	=	000000	21-1482			
ADDW15	=	000000	21-1482			
ADDW2	=	000000	21-1482			
ADDW3	=	000000	21-1482			
ADDW4	=	000000	21-1482			
ADDW5	=	000000	21-1482			
ADDW6	=	000000	21-1482			
ADDW7	=	000000	21-1482			
ADDW8	=	000000	21-1482			
ADDW9	=	000000	21-1482			
ADEVCT	=	000000	21-1482	21-1482		
ADEVMT	=	000000	21-1482	21-1482		
ADR	=	004132	#30-2085	30-2088		
ADRTBL	=	002702	#30-1931	30-1942	30-2083	
AENV	=	000000	21-1482	21-1482		
AENVMT	=	000000	21-1482	21-1482		
AFATAL	=	000000	21-1482	21-1482		
AMADR1	=	000000	21-1482	21-1482		
AMADR2	=	000000	21-1482	21-1482		
AMADR3	=	000000	21-1482	21-1482		
AMADR4	=	000000	21-1482	21-1482		
AMAMS1	=	000000	21-1482	21-1482		
AMAMS2	=	000000	21-1482	21-1482		
AMAMS3	=	000000	21-1432	21-1482		
AMAMS4	=	000000	21-1482	21-1482		
AMSGAD	=	000000	21-1482	21-1482		
AMSGLG	=	000000	21-1482	21-1482		
AMSGTY	=	000000	21-1482	21-1482		
AMTYP1	=	000000	21-1482	21-1482		
AMTYP2	=	000000	21-1482	21-1482		
AMTYP3	=	000000	21-1482	21-1482		
AMTYP4	=	000000	21-1482	21-1482		
APASS	=	000000	21-1482	21-1482		
APRIOR	=	000000	21-1482			
APTCSU	=	000040	#77-3731	78-3735		
APTENV	=	000001	73-3592	77-3686	#77-3729	78-3735
APTSIZ	=	000200	30-1967	#77-3728		
APTSPO	=	000100	77-3688	#77-3730	78-3735	
APTSZD	=	003636	30-1988	30-1990	#30-2016	
ASWREG	=	000000	21-1482	21-1482		
ATESTN	=	000000	21-1482	21-1482		
AUNIT	=	000000	21-1482	21-1482		



SYMBOL CROSS REFERENCE		REFERENCES									
SYMBOL	VALUE										
BUF	002454	#30-1886	67-3248	67-3282							
CATCH	014520	17-1458	#72-3558								
CHKSUM	014010	69-3369	69-3385	#69-3410							
CKSWR	= 104406	73-3602	76-3669	#80-3745							
CLK	013630	67-3237	#67-3326								
CLKCNT	002444	#30-1882	*67-3220	67-3276	*67-3326						
CMPARE	007330	49-2641	#49-2644								
CNTLP	002564	#30-1887	69-3372								
COMP	013432	#67-3284	67-3292								
CONT	006246	#44-2476	44-2479								
CONT41	012354	63-3099	#63-3107								
CPSAVE	015674	74-3618	76-3669	*76-3669	76-3669	*76-3669	#76-3669				
CR	= 000015	#17-1433	30-1888	30-1888	30-1889	71-3535	78-3735	78-3735	81-3796	81-3797	
		81-3798	81-3799	83-3930	83-3932						
CRBUF	002656	#30-1918	81-3760	81-3761	82-3832						
CRCR	002654	#30-1917	30-2007	30-2064	81-3758	82-3826					
CRLF	= 000200	#17-1433	78-3735	78-3735							
CRPSW	002666	#30-1922									
CRVECT	002664	#30-1921									
CTBUF	002662	#30-1920	81-3757	81-3760	81-3791	82-3819					
CTCSR	002660	#30-1919	34-2180	34-2193	34-2203	53-2775	54-2793	54-2812	54-2822	54-2830	
		54-2838	54-2843	55-2852	56-2877	57-2893	57-2897	57-2907	57-2912	58-2934	
		58-2944	58-2948	59-2966	60-3004	62-3066	63-3104	63-3110	63-3115	64-3137	
		65-3177	65-3181	81-3789	82-3817						
CTPSW	002672	#30-1924									
CTSTFL	002624	#30-1900	*30-1957	*30-1983	*30-2063	31-2103	32-2120	33-2132	33-2150	34-2178	
		34-2191	38-2258	38-2268	38-2284	41-2375	42-2387	44-2459	45-2483	46-2531	
		47-2569	48-2595	49-2624	62-3051	63-3083	64-3123	67-3233	67-3272	*70-3531	
CTVECT	002670	#30-1923									
DDISP	= 177570	#17-1433	21-1482	30-1967							
DEVADR	003014	#30-1942	30-1992	30-2024	69-3336						
DH1	024265	22-1485	22-1495	22-1500	22-1505	23-1524	23-1529	23-1534	23-1539	23-1544	
		23-1549	23-1554	24-1558	24-1563	24-1568	24-1573	24-1578	24-1583	27-1695	
		27-1700	27-1705	27-1710	27-1715	27-1720	27-1725	#83-3918			
DH10	024411	23-1519	25-1612	25-1617	25-1622	25-1627	25-1632	25-1642	26-1646	26-1651	
		26-1656	26-1661	26-1666	26-1671	26-1676	26-1681	26-1686	30-1846	#83-3922	
DH103	024536	29-1807	#83-3925								
DH105	024573	29-1817	30-1821	30-1836	#83-3926						
DH110	024637	30-1831	#83-3927								
DH112	024703	30-1841	#83-3928								
DH2	024312	22-1490	#83-3919								
DH40	024435	25-1637	#83-3923								
DH53	024471	27-1690	#83-3924								
DH6	024337	22-1510	24-1588	24-1593	24-1598	25-1602	25-1607	27-1730	28-1734	28-1739	
		28-1744	28-1749	28-1754	28-1759	28-1764	28-1769	28-1774	29-1778	29-1782	
		29-1787	29-1792	29-1798	29-1803	29-1812	30-1826	30-1851	30-1856	30-1861	
		30-1866	#83-3920								
DH7	024364	23-1514	#83-3921								
DISPLA	001042	#21-1482	*30-1967	*30-1967	73-3582	76-3669					
DISPRE	000174	#17-1469	30-1967								
DSWR	= 177570	#17-1433	21-1482	30-1967							
DT1	025046	22-1486	22-1496	22-1501	22-1506	23-1525	23-1530	23-1535	23-1540	23-1545	





SYMBOL CROSS REFERENCE		REFERENCES	
SYMBOL	VALUE		
EM27	021243	24-1592	#83-3860
EM3	020154	22-1494	#83-3840
EM30	021302	24-1597	#83-3861
EM31	021333	25-1601	#83-3862
EM32	021366	25-1606	#83-3863
EM33	021434	25-1611	#83-3864
EM34	021476	25-1616	#83-3865
EM35	021534	25-1621	#83-3866
EM36	021564	25-1626	#83-3867
EM37	021616	25-1631	#83-3868
EM4	020221	22-1499	#83-3841
EM40	021656	25-1636	#83-3869
EM41	021704	25-1641	#83-3870
EM42	021744	26-1645	#83-3871
EM43	021776	26-1650	#83-3872
EM44	022027	26-1655	#83-3873
EM45	022063	26-1660	#83-3874
EM46	022120	26-1665	#83-3876
EM47	022120	26-1670	#83-3875
EM5	020243	22-1504	#83-3842
EM50	022154	26-1675	#83-3877
EM51	022201	26-1680	#83-3878
EM52	022231	26-1685	#83-3879
EM53	022306	27-1689	#83-3880
EM54	022332	27-1694	#83-3881
EM55	022470	27-1699	#83-3884
EM56	022370	27-1704	#83-3882
EM57	022426	27-1709	#83-3883
EM6	020300	22-1509	#83-3843
EM60	022470	27-1714	#83-3885
EM61	022517	27-1719	#83-3886
EM62	022543	27-1724	#83-3887
EM63	022610	27-1729	#83-3888
EM64	022634	28-1733	#83-3889
EM65	022660	28-1738	#83-3890
EM66	022726	28-1743	#83-3891
EM67	022764	28-1748	#83-3892
EM7	020324	23-1513	#83-3844
EM70	023027	28-1753	#83-3893
EM71	023074	28-1758	#83-3894
EM72	023243	28-1763	#83-3897
EM73	023136	28-1768	#83-3895
EM74	023174	28-1773	#83-3896
EM75	023243	29-1777	#83-3898
EM76	023272	29-1781	#83-3899
EM77	023316	29-1786	#83-3900
ENDADD	= 025222	#69-3337	69-3413 69-3505 69-3522
ENDADR	025222	69-3337	#83-3947
ENDB7	004676	34-2198	34-2215 #34-2217
ENDEV	014272	67-3305	#70-3526
ENDMG	014416	71-3534	#71-3535
ENDSTK	002412	#30-1876	

SYMBOL CROSS REFERENCE		REFERENCES								
SYMBOL	VALUE	REFERENCES								
ERROR	= 104000	#17-1433	31-2105	32-2122	33-2134	33-2144	33-2152	33-2163	34-2181	34-2194
		35-2230	36-2242	37-2251	38-2270	38-2279	38-2286	38-2296	39-2306	39-2311
		39-2318	39-2324	39-2332	40-2343	40-2348	40-2355	40-2362	40-2370	41-2382
		42-2395	42-2400	42-2407	42-2413	42-2420	43-2452	44-2464	44-2473	44-2481
		45-2502	45-2515	45-2524	46-2549	46-2562	47-2591	48-2618	49-2657	50-2673
		50-2681	51-2702	51-2715	52-2734	52-2748	53-2776	54-2795	54-2802	54-2813
		54-2823	54-2831	54-2839	55-2855	56-2872	56-2878	57-2899	57-2908	58-2936
		58-2950	59-2973	59-2989	60-3016	61-3043	62-3069	62-3074	63-3105	63-3112
		64-3141	64-3146	65-3178	66-3205	66-3217	67-3262	67-3268	67-3279	67-3297
		72-3562								
ERRVEC	= 000004	#17-1433	30-1967	*30-1967	*30-1967	76-3669	*76-3669	*76-3669	*76-3669	
FIRST	002336	#30-1871	*49-2642	49-2644	83-3942					
FLAG44	003002	#30-1933	*30-1960	*30-1976	38-2256	49-2648	54-2782	54-2826	56-2860	62-3049
		63-3081	64-3121	65-3159	66-3194	67-3287	67-3309			
GETB	014126	69-3381	#69-3466	69-3468	69-3472					
GETLIN	014062	69-3376	#69-3443	69-3453						
GNS	= *****	80-3745	80-3745	80-3745	80-3745	80-3745	80-3745	80-3745	80-3745	80-3745
		80-3745	80-3745	80-3745	80-3745	80-3745	80-3745	80-3745	80-3745	80-3745
GOAGIN	014432	71-3534	#72-3537							
GTSWR	= 104405	#80-3745								
HT	= 000011	#17-1433	78-3735	78-3735						
ID	004620	30-1984	34-2187	34-2195	#34-2202					
INIT	003462	30-1982	#30-1985							
IOTVEC	= 000020	#17-1433	*30-1967	*30-1967						
JIMSTK	002432	#30-1877	69-3362							
LF	= 000012	#17-1433	30-1888	30-1888	30-1889	71-3535	78-3735	78-3735	81-3796	81-3797
		81-3798	81-3799	83-3930	83-3932					
LKS	002674	#30-1927	41-2378	42-2393	42-2403	42-2404	42-2410	42-2411	42-2415	42-2417
		43-2432	44-2461	44-2466	44-2467	44-2469	44-2470	44-2476	45-2490	45-2491
		45-2492	45-2494	45-2495	45-2496	45-2504	45-2508	45-2518	45-2519	45-2520
		45-2527	46-2539	46-2540	46-2541	46-2543	46-2544	46-2545	46-2565	47-2574
		47-2575	47-2576	47-2578	47-2579	47-2580	47-2587	48-2600	48-2601	48-2602
		48-2604	48-2605	48-2606	48-2609	48-2613	49-2625	49-2630	49-2631	49-2633
		49-2634	49-2637	67-3239	83-3940	83-3942				
LOC1	002340	#30-1872	*69-3364	*69-3484						
LOC2	002342	#30-1873	*69-3365	*69-3486						
MANL	003502	30-1986	#30-1989							
MFPT	= 000007	#17-1434	30-1973							
MFR	017763	81-3772	#81-3797							
MOR	017775	81-3777	#81-3798							
MPAR	017752	81-3767	#81-3796							
MSG	017724	81-3766	81-3771	81-3776	81-3781	#81-3787	82-3823			
MSTOP	020010	81-3782	#81-3799							
M1	024754	34-2211	#83-3930							
M2	025014	34-2213	#83-3932							
M2A	025016	*30-2055	#83-3933							
NOEOP	014316	70-3529	#70-3531							
OLDPC	002620	#30-1893	*72-3561	83-3946						
OLDSUM	002436	#30-1879	*69-3370	69-3386						
OUTTST	020020	17-1475	#82-3815	82-3829	82-3834					
PFECDF	015206	74-3618	#74-3618							
PFECDH	015146	74-3618	#74-3618							

SYMBOL CROSS REFERENCE

SYMBOL	CROSS REFERENCE	VALUE	REFERENCES
PFECDT		015176	74-3618 #74-3618
PFECM		015106	74-3618 #74-3618
PFECWS		015076	74-3618 #74-3618
PIRQ	=	177772	#17-1433
PIRQVE	=	000240	#17-1433
PROMPT		002566	#30-1888 69-3375
PRO	=	000000	#17-1433
PR1	=	000040	#17-1433
PR2	=	000100	#17-1433
PR3	=	000140	#17-1433
PR4	=	000200	#17-1433
PR5	=	000240	#17-1433
PR6	=	000300	#17-1433
PR7	=	000340	#17-1433
PS	=	177776	#17-1433 17-1433
PSW	=	177776	#17-1433 69-3360 *69-3361 *69-3390
PUTLIN		014030	69-3373 69-3379 #69-3426 69-3432
PWRVEC	=	000024	#17-1433 *30-1967 *30-1967 *75-3626 *75-3627 *75-3636 *75-3644 *75-3653 *75-3654
RBUF		002636	#30-1907 37-2247 54-2786 55-2847 55-2851 56-2865 57-2885 58-2921 59-2962
			60-3000 60-3005 61-3027 62-3059 62-3067 62-3072 63-3091 63-3095 63-3107
			64-3131 64-3138 64-3143 65-3156 65-3168 66-3207 67-3241 67-3321 83-3939
RBUF58	=	176502	#69-3333 69-3450 69-3469
RCSR		002634	#30-1906 30-2065 30-2084 36-2238 40-2341 40-2351 40-2352 40-2358 40-2359
			40-2365 40-2367 43-2434 54-2789 54-2799 54-2809 54-2818 54-2828 54-2835
			55-2849 55-2853 56-2863 56-2867 56-2869 56-2874 56-2875 57-2883 57-2891
			57-2902 57-2911 58-2925 58-2927 58-2938 59-2964 59-2967 59-2983 60-2998
			60-3002 60-3012 61-3025 61-3029 61-3038 63-3093 63-3098 64-3134 65-3166
			66-3196 66-3200 67-3243 83-3938 83-3943 83-3944 83-3945 83-3946
RCSR58	=	176500	#69-3332 69-3448 69-3467
RCV		013614	67-3231 #67-3321
RCVCNT		002440	#30-1880 *67-3245 67-3264 *67-3322 83-3945
RCVDON		010346	54-2783 54-2810 #54-2815
RDCHR	=	104407	79-3740 #80-3745
RDLIN	=	104410	#80-3745
RESTTE		014236	69-3383 #69-3516
RESVEC	=	000010	#17-1433
RPSW		002646	#30-1911 59-2956 59-2958 59-2985 67-3227 67-3232 67-3303
RSTRT		014472	72-3544 #72-3548
RTCPSW		002700	#30-1929 45-2487 45-2489 45-2529 46-2534 46-2536 46-2567 67-3228 67-3238
			67-3304
RTCVT		002676	#30-1928 42-2389 42-2390 42-2422 45-2486 45-2488 45-2505 45-2517 45-2528
			46-2533 46-2535 46-2552 46-2566 47-2572 47-2573 47-2593 48-2598 48-2599
			48-2620 67-3225 67-3237 67-3301
RVECT		002644	#30-1910 40-2337 40-2338 40-2372 57-2886 57-2887 57-2901 57-2913 58-2918
			58-2919 58-2939 58-2952 59-2955 59-2957 59-2976 59-2984 59-2991 60-2996
			60-2997 60-3018 61-3023 61-3024 61-3045 67-3224 67-3231 67-3300
R6	=	%000006	#17-1433 *30-1967 *30-1967 30-1967
R7	=	%000007	#17-1433
SAVEPS		002434	#30-1878 *69-3360 69-3390
SAVETE		014202	69-3367 #69-3499
SAVEO		002344	#30-1874 *69-3499 69-3516
SAVLOC		002346	#30-1875 69-3501 69-3518

SYMBOL CROSS REFERENCE		REFERENCES									
SYMBOL	VALUE										
SCOPE	= 000004	#17-1433	31-2096	32-2113	33-2127	34-2167	35-2219	36-2235	37-2244	38-2253	
		39-2298	40-2335	41-2374	42-2386	43-2423	44-2458	45-2482	46-2530	47-2568	
		48-2594	49-2623	50-2662	51-2686	52-2718	53-2753	54-2781	55-2844	56-2859	
		57-2881	58-2914	59-2953	60-2992	61-3019	62-3046	63-3078	64-3116	65-3149	
		66-3185	67-3218	69-3328	71-3534						
SCPSW	002452	#30-1885	*67-3228	67-3304							
SECND	002616	#30-1892	*49-2656	83-3942							
SETADR	004100	30-2073	#30-2079								
SHIFT	003740	#30-2045	30-2054								
SIZE	003512	#30-1992									
SRPSW	002450	#30-1884	*67-3227	67-3303							
STACK	= 001100	#17-1433									
START	003046	17-1473	#30-1954								
STKMT	= 177774	#17-1433									
STPSW	002446	#30-1883	*67-3226	67-3302							
SWR	001040	#21-1482	30-1967	*30-1967	30-1967	*30-1967	*30-1967	30-1981	30-1989	34-2214	
		38-2254	38-2261	41-2375	42-2387	43-2430	44-2459	45-2483	46-2531	47-2569	
		48-2595	49-2624	49-2659	62-3047	62-3054	63-3079	63-3086	64-3117	64-3119	
		64-3126	66-3186	67-3235	67-3274	69-3357	73-3587	73-3599	73-3603	75-3634	
		75-3646	76-3669	76-3669	76-3669	76-3669	76-3669	79-3740	79-3740		
SWREG	000176	#17-1470	30-1967	79-3740	79-3740						
SW0	= 000001	#17-1433									
SW00	= 000001	#17-1433	17-1433								
SW01	= 000002	#17-1433	17-1433								
SW02	= 000004	#17-1433	17-1433								
SW03	= 000010	#17-1433	17-1433								
SW04	= 000020	#17-1433	17-1433								
SW05	= 000040	#17-1433	17-1433								
SW06	= 000100	#17-1433	17-1433								
SW07	= 000200	#17-1433	17-1433								
SW08	= 000400	#17-1433	17-1433								
SW09	= 001000	#17-1433	17-1433								
SW1	= 000002	#17-1433									
SW10	= 002000	#17-1433									
SW11	= 004000	#17-1433									
SW12	= 010000	#17-1433									
SW13	= 020000	#17-1433									
SW14	= 040000	#17-1433									
SW15	= 100000	#17-1433									
SW2	= 000004	#17-1433									
SW3	= 000010	#17-1433									
SW4	= 000020	#17-1433									
SW5	= 000040	#17-1433									
SW6	= 000100	#17-1433									
SW7	= 000200	#17-1433									
SW8	= 000400	#17-1433									
SW9	= 001000	#17-1433									
TA	002610	#30-1889	69-3378								
TBITVE	= 000014	#17-1433									
TBUF	002642	#30-1909	32-2116	34-2169	34-2174	35-2221	35-2224	53-2761	53-2764	54-2788	
		54-2817	55-2848	56-2866	57-2890	58-2924	59-2963	60-3001	61-3028	62-3061	
		63-3092	64-3133	65-3165	66-3198	67-3249	67-3317	83-3937			

SYMBOL	CROSS REFERENCE	VALUE	REFERENCES
TBUF58	=	176506	#69-3335 *69-3431
TCLOCK		005536	34-2216 #40-2373
TCONS		004020	30-2061 #30-2063
TCSR		002640	#30-1908 31-2099 33-2129 33-2140 33-2141 33-2146 33-2147 33-2157 33-2159
			33-2162 34-2168 34-2170 34-2175 34-2186 35-2220 35-2222 35-2227 38-2266
			38-2275 38-2276 38-2281 38-2282 38-2291 38-2293 38-2295 39-2304 39-2314
			39-2315 39-2321 39-2322 39-2327 39-2329 50-2663 50-2666 50-2676 50-2684
			51-2687 51-2692 51-2694 51-2704 52-2719 52-2726 52-2728 52-2744 53-2754
			53-2759 53-2760 53-2762 53-2771 54-2785 54-2805 54-2815 55-2846 56-2864
			57-2882 57-2884 58-2920 59-2961 60-2999 61-3026 62-3058 62-3062 63-3090
			63-3096 63-3103 63-3109 63-3114 64-3130 64-3132 64-3136 65-3152 67-3240
			67-3242 67-3258 67-3315 83-3936
TCSR58	=	176504	#69-3334 69-3429
TIMER		014162	69-3428 69-3447 69-3466 #69-3484
TKVEC	=	000060	#17-1433
TMP1		002626	#30-1901 *30-2059 *30-2075 30-2081 *30-2082
TMP2		002630	#30-1902 *30-1993 *30-2001 *30-2011 *30-2016 *30-2020 30-2040 70-3528 72-3543
TMP3		002632	#30-1903 *30-2058 30-2072 *30-2074 *30-2080
TOLER		007342	49-2646 #49-2648
TPSW		002652	#30-1913 30-2067 52-2721 52-2723 52-2751 67-3226 67-3230 67-3302
TPVEC	=	000064	#17-1433
TRAPVE	=	000034	#17-1433 *30-1967 *30-1967
TRTVEC	=	000014	#17-1433
TSTDEV		004050	30-2062 #30-2072 30-2077 70-3532
TSTDVM		003646	#30-2018 30-2023
TST1		004172	30-2069 30-2095 #31-2096 72-3546
TST10		005044	#38-2253
TST11		005252	#39-2298
TST12		005404	#40-2335
TST13		005544	#41-2374
TST14		005614	41-2375 41-2375 #42-2386
TST15		005764	42-2387 42-2387 #43-2423
TST16		006154	#44-2458
TST17		006262	44-2459 44-2459 44-2477 #45-2482
TST2		004244	#32-2113
TST20		006522	45-2483 45-2483 #46-2530
TST21		006720	46-2531 46-2531 #47-2568
TST22		007054	47-2569 47-2569 #48-2594
TST23		007222	48-2595 48-2595 #49-2623
TST24		007410	49-2624 49-2624 49-2660 #50-2662
TST25		007520	#51-2686
TST26		007646	#52-2718
TST27		010012	#53-2753
TST3		004316	#33-2127
TST30		010156	#54-2781
TST31		010462	#55-2844
TST32		010554	#56-2859
TST33		010700	#57-2881
TST34		011102	#58-2914
TST35		011304	#59-2953
TST36		011512	#60-2992
TST37		011656	#61-3019





SYMBOL	CROSS REFERENCE	VALUE	REFERENCES	REF	V01						
\$MAMS2		001122	#21-1482								
\$MAMS3		001126	#21-1482								
\$MAMS4		001132	#21-1482								
\$MBADR		000502	#19-1480								
\$MFLG		016140	*77-3678	77-3684	*77-3719	#77-3723					
\$MNEW		017511	79-3740	#79-3740							
\$MSGAD		001102	#21-1482	*77-3694	77-3697						
\$MSGLG		001104	#21-1482	*77-3699							
\$MSGTY		001066	#21-1482	*30-1955	77-3692	*77-3700	77-3712	*77-3715			
\$MSWR		017500	79-3740	#79-3740							
\$MTYP1		001117	#21-1482								
\$MTYP2		001123	#21-1482								
\$MTYP3		001127	#21-1482								
\$MTYP4		001133	#21-1482								
\$NULL		001054	#21-1482	78-3735	78-3735	78-3735					
\$NWTST	=	000001	#30-2096	30-2096	#31-2096	#31-2113	31-2113	#32-2113	#32-2127	32-2127	#33-2127
			#33-2167	33-2167	#34-2167	#34-2219	34-2219	#35-2219	#35-2235	35-2235	#36-2235
			#36-2244	36-2244	#37-2244	#37-2253	37-2253	#38-2253	#38-2298	38-2298	#39-2298
			#39-2335	39-2335	#40-2335	#40-2374	40-2374	#41-2374	#41-2386	41-2386	#42-2386
			#42-2423	42-2423	#43-2423	#43-2458	43-2458	#44-2458	#44-2482	44-2482	#45-2482
			#45-2530	45-2530	#46-2530	#46-2568	46-2568	#47-2568	#47-2594	47-2594	#48-2594
			#48-2623	48-2623	#49-2623	#49-2662	49-2662	#50-2662	#50-2686	50-2686	#51-2686
			#51-2718	51-2718	#52-2718	#52-2753	52-2753	#53-2753	#53-2781	53-2781	#54-2781
			#54-2844	54-2844	#55-2844	#55-2859	55-2859	#56-2859	#56-2881	56-2881	#57-2881
			#57-2914	57-2914	#58-2914	#58-2953	58-2953	#59-2953	#59-2992	59-2992	#60-2992
			#60-3019	60-3019	#61-3019	#61-3046	61-3046	#62-3046	#62-3078	62-3078	#63-3078
			#63-3116	63-3116	#64-3116	#64-3149	64-3149	#65-3149	#65-3185	65-3185	#66-3185
			#66-3218	66-3218	#67-3218	#68-3328	68-3328	#69-3328			
\$SOCNT		016732	*78-3736	*78-3736	#78-3736						
\$SOMODE		016734	*78-3736	*78-3736	78-3736	*78-3736	*78-3736	#78-3736			
\$OVER		015660	76-3669	76-3669	#76-3669						
\$PASS		001074	#21-1482	*30-1967	34-2206	*71-3534	*71-3534	71-3534	71-3534		
\$PASTM		000506	#19-1480								
\$POWER		015364	75-3659	#75-3665							
\$PWRDN		015212	30-1967	#75-3626	75-3653						
\$PWRMG		015346	#75-3659								
\$PWRUP		015264	75-3636	#75-3644							
\$QUES		001062	#21-1482	78-3735	78-3735	79-3740	79-3740	79-3740	79-3740		
\$RDCHR		017220	#79-3740	80-3745	80-3745						
\$RDDEC	=	*****	80-3745								
\$RDLIN		017350	#79-3740	80-3745	80-3745						
\$RDOCT	=	*****	80-3745								
\$RDSZ	=	000010	#79-3740	79-3740							
\$RTNAD		014410	#71-3534								
\$R2A	=	*****	80-3745								
\$SAVRE	=	*****	80-3745								
\$SAVR6		015362	*75-3635	75-3645	*75-3655	*75-3656	#75-3664				
\$SCOPE		015374	30-1967	#76-3669							
\$SETUP	=	000137	#17-1465	17-1465	#17-1465	17-1465	#17-1465	17-1465	#17-1465	17-1465	#17-1465
			17-1465	#17-1465	17-1465	#17-1465	30-1967	30-1967	30-1967	30-1967	30-1967
			30-1967	30-1967	30-1967	30-1967	30-1967	30-1967	30-1967	30-1967	30-1967
			76-3669	79-3740	79-3740				71-3534	71-3534	71-3534



SYMBOL CROSS REFERENCE

SYMBOL	CROSS REFERENCE	REFERENCES	REFERENCES	REFERENCES	REFERENCES	REFERENCES	REFERENCES	REFERENCES	REFERENCES	REFERENCES
SYMBOL	VALUE									
\$STUP	= 177777	#17-1465	#17-1465	17-1465	#17-1465	#17-1465	17-1465	#17-1465	#17-1465	17-1465
		#17-1465	#17-1465	17-1465	#17-1465	#17-1465	17-1465	#17-1465	#17-1465	17-1465
\$SVLAD	015624	76-3669	76-3669	#76-3669						
\$SVPC	= 000500	#18-1478	18-1478							
\$SWR	= 161000	#17-1439	21-1482	21-1482	21-1482	30-1967	30-1967	30-1967	30-1967	30-1967
		31-2096	32-2113	33-2127	34-2167	35-2219	36-2235	37-2244	38-2253	39-2298
		40-2335	41-2374	42-2386	43-2423	44-2458	45-2482	46-2530	47-2568	48-2594
		49-2623	50-2662	51-2686	52-2718	53-2753	54-2781	55-2844	56-2859	57-2881
		58-2914	59-2953	60-2992	61-3019	62-3046	63-3078	64-3116	65-3149	66-3185
		67-3218	69-3328	71-3534	71-3534	71-3534	71-3534	71-3534	76-3669	76-3669
		76-3669	76-3669	76-3669	76-3669	76-3669	76-3669	76-3669	76-3669	76-3669
		76-3669	76-3669	76-3669	76-3669	76-3669	76-3669	76-3669	76-3669	76-3669
\$SWREG	001110	#21-1482	30-1967							
\$SWRMK	= 000000	76-3669								
\$TESTN	001072	#21-1482	*30-1956	74-3618	*76-3669	83-3936	83-3937	83-3938	83-3939	83-3940
		83-3941	83-3942	83-3943	83-3944	83-3945	83-3946			
\$TKB	001046	#21-1482	78-3735	78-3735	78-3735	78-3735	79-3740	79-3740	79-3740	79-3740
		79-3740	79-3740							
\$TKS	001044	#21-1482	78-3735	78-3735	78-3735	78-3735	79-3740	79-3740	79-3740	79-3740
		79-3740	79-3740							
\$TN	= 000047	#17-1438	30-2096	31-2096	#31-2096	31-2113	32-2113	#32-2113	32-2127	33-2127
		#33-2127	33-2167	34-2167	#34-2167	34-2219	35-2219	#35-2219	35-2235	36-2235
		#36-2235	36-2244	37-2244	#37-2244	37-2253	38-2253	#38-2253	38-2298	39-2298
		#39-2298	39-2335	40-2335	#40-2335	*40-2373	40-2374	41-2374	#41-2374	41-2375
		41-2386	42-2386	#42-2386	42-2387	42-2423	43-2423	#43-2423	43-2458	44-2458
		#44-2458	44-2459	44-2482	45-2482	#45-2482	45-2483	45-2530	46-2530	#46-2530
		46-2531	46-2568	47-2568	#47-2568	47-2569	47-2594	48-2594	#48-2594	48-2595
		48-2623	49-2623	#49-2623	49-2624	49-2662	50-2662	#50-2662	50-2686	51-2686
		#51-2686	51-2718	52-2718	#52-2718	52-2753	53-2753	#53-2753	53-2781	54-2781
		#54-2781	54-2844	55-2844	#55-2844	55-2859	56-2859	#56-2859	56-2881	57-2881
		#57-2881	57-2914	58-2914	#58-2914	58-2953	59-2953	#59-2953	59-2992	60-2992
		#60-2992	60-3019	61-3019	#61-3019	61-3046	62-3046	#62-3046	62-3078	63-3078
		#63-3078	63-3116	64-3116	#64-3116	64-3149	65-3149	#65-3149	65-3185	66-3185
		#66-3135	66-3218	67-3218	#67-3218	68-3328	69-3328	#69-3328		
\$TPB	001052	#21-1482	78-3735	78-3735	78-3735					
\$TPFLG	001057	#21-1482	78-3735	78-3735	78-3735					
\$TPS	001050	#21-1482	58-2922	78-3735	78-3735	78-3735				
\$TRAP	017522	30-1967	#80-3745							
\$TRAP2	017544	#80-3745	80-3745							
\$TRP	= 000011	#80-3745	80-3745	80-3745	80-3745	80-3745	#80-3745	80-3745	80-3745	80-3745
		80-3745	#80-3745	80-3745	80-3745	80-3745	80-3745	#80-3745	80-3745	80-3745
		80-3745	80-3745	#80-3745	80-3745	80-3745	80-3745	80-3745	#80-3745	80-3745
		80-3745	80-3745	80-3745	#80-3745	80-3745	80-3745	80-3745	80-3745	#80-3745
		80-3745	80-3745	80-3745	80-3745	80-3745	80-3745	80-3745	80-3745	#80-3745
		80-3745	80-3745	80-3745	80-3745	#80-3745				
\$TRPAD	017556	80-3745	#80-3745							
\$STSM	000504	#19-1480								
\$STSNM	001002	#21-1482	*40-2373	*70-3526	*71-3534	73-3582	76-3669	*76-3669	76-3669	76-3669
		76-3669	76-3669							
\$TTYIN	017456	79-3740	79-3740	79-3740	#79-3740					
\$TYPBN	= *****	80-3745								
\$TYPDS	= *****	80-3745								
\$TYPE	016144	77-3705	#78-3735	80-3745	80-3745					

SYMBOL	CROSS REFERENCE	VALUE	REFERENCES	SEQUENCE	CREF	V01
\$TYPEC		016356	78-3735	78-3735	#78-3735	79-3740
\$TYPEX		016506	78-3735	78-3735	#78-3735	
\$TYPOC		016534	#78-3736	80-3745	80-3745	
\$TYPON		016550	78-3736	#78-3736	80-3745	
\$TYPOS		016510	#78-3736	80-3745		
\$UNIT		001100	#21-1482	*30-1959	*30-2076	*30-2079 *72-3548
\$UNITM		000510	#19-1480			
\$USWR		001112	#21-1482	30-1961	*30-1963	65-3169 66-3208
\$VECT1		001136	#21-1482	30-2029		
\$VECT2		001140	#21-1482			
\$XOFF	=	000023	78-3735	78-3735		
\$XON	=	000021	78-3735	78-3735	78-3735	79-3740
\$XTSTR		015406	#76-3669			
\$GET4	=	000000	#71-3534	71-3534		
\$OF ILL		016733	*78-3736	*78-3736	78-3736	#78-3736
\$4OCAT	=	*****	76-3669			
.\$ERRT		001146	#22-1483			
.\$X	=	000500	#19-1480	19-1480		



