

# KWV11A

DIAGNOSTIC  
MD-11-DVKWA-A

EP-DVKWA-A-DL-A

COPYRIGHT © 1976

FICHE 1 OF 1

NOV 1976

digital

MADE IN U.S.A.

[Chart 1]	[Chart 2]	[Chart 3]	[Chart 4]	[Chart 5]	[Chart 6]	[Chart 7]	[Chart 8]	[Chart 9]	[Chart 10]	[Chart 11]	[Chart 12]
[Chart 13]	[Chart 14]	[Chart 15]	[Chart 16]	[Chart 17]	[Chart 18]	[Chart 19]	[Chart 20]	[Chart 21]	[Chart 22]	[Chart 23]	[Chart 24]
[Chart 25]	[Chart 26]	[Chart 27]	[Chart 28]	[Chart 29]	[Chart 30]	[Chart 31]	[Chart 32]	[Chart 33]	[Chart 34]	[Chart 35]	[Chart 36]
[Chart 37]	[Chart 38]	[Chart 39]	[Chart 40]	[Chart 41]	[Chart 42]	[Chart 43]	[Chart 44]	[Chart 45]	[Chart 46]	[Chart 47]	[Chart 48]
[Chart 49]	[Chart 50]	[Chart 51]	[Chart 52]	[Chart 53]	[Chart 54]	[Chart 55]	[Chart 56]	[Chart 57]	[Chart 58]	[Chart 59]	[Chart 60]
[Chart 61]	[Chart 62]	[Chart 63]	[Chart 64]	[Chart 65]	[Chart 66]	[Chart 67]	[Chart 68]	[Chart 69]	[Chart 70]	[Chart 71]	[Chart 72]
[Chart 73]	[Chart 74]	[Chart 75]	[Chart 76]	[Chart 77]	[Chart 78]	[Chart 79]	[Chart 80]	[Chart 81]	[Chart 82]	[Chart 83]	[Chart 84]
[Chart 85]	[Chart 86]	[Chart 87]	[Chart 88]	[Chart 89]	[Chart 90]	[Chart 91]	[Chart 92]	[Chart 93]	[Chart 94]	[Chart 95]	[Chart 96]
[Chart 97]	[Chart 98]	[Chart 99]	[Chart 100]	[Chart 101]	[Chart 102]	[Chart 103]	[Chart 104]	[Chart 105]	[Chart 106]	[Chart 107]	[Chart 108]
[Chart 109]	[Chart 110]	[Chart 111]	[Chart 112]	[Chart 113]	[Chart 114]	[Chart 115]	[Chart 116]	[Chart 117]	[Chart 118]	[Chart 119]	[Chart 120]

[Small illegible text]

.REM x

IDENTIFICATION

PRODUCT CODE:	MAINDEC-11-DVKWA-A-0
PRODUCT NAME:	KWV11A DIAGNOSTIC
DATE CREATED:	OCTOBER 1976
MAINTAINER:	DIAGNOSTIC ENGINEERING

COPYRIGHT (C) 1976  
DIGITAL EQUIPMENT CORPORATION

THIS SOFTWARE IS FURNISHED UNDER A LICENSE FOR USE ONLY ON A SINGLE COMPUTER SYSTEM AND MAY BE COPIED ONLY WITH THE INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE, OR ANY OTHER COPIES THEREOF, MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON EXCEPT FOR USE ON SUCH SYSTEM AND TO ONE WHO AGREES TO THESE LICENSE TERMS. TITLE TO AND OWNERSHIP OF THE SOFTWARE SHALL AT ALL TIMES REMAIN IN DEC.

THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION.

DEC ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DEC.

10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25  
26  
27  
28  
29  
30  
31  
32  
33  
34  
35  
36  
37  
38  
39  
40  
41  
42  
43  
44  
45  
46  
47  
48  
49  
50



100.98  
101  
102  
103  
104  
105  
106  
107  
108  
109  
110  
111  
112  
113  
114  
115  
116  
117  
118  
119  
120  
121  
122  
123  
124  
125  
126  
127  
128  
129  
130  
131  
132  
133  
134  
135  
136  
137  
138  
139  
140

1.0 ABSTRACT  
-----

THIS PROGRAM ALLOWS THE USER CHECK-OUT OR DEBUG THE KVV11A, PROGRAMMABLE REAL-TIME CLOCK. THE LOGIC TEST IS SELF CONTAINED AND NEEDS NO EXTERNAL MAINTENANCE HARDWARE OR OPERATOR INTERVENTION WITH ONLY ONE EXCEPTION: IF THE CUSTOMER HARDWARE CONNECTED TO THE KVV11 COULD INJECT SIGNALS ON ST2, ST1, OR SLAVE IN INPUTS, IT MUST BE DISCONNECTED.

EVEN THOUGH THE KVV11 IS A Q BUS OPTION, THIS PROGRAM WAS DESIGNED TO RUN ON ANY PDP-11 FAMILY COMPUTER. IF THE USER IS UNFAMILAR WITH AN LSI-11 HE SHOULD REVIEW SECTIONS 8.4 AND 8.5. A SOFTWARE SWITCH REGISTER IS INCLUDED WITH THIS PROGRAM. IT CAN BE USED ON AN LSI-11 OR BY CPU'S THAT HAVE HARDWARE SWITCH REGISTERS, SEE SECTION 8.6.

EVERY EFFORT WAS MADE TO MAKE THIS PROGRAM CONFORM TO LSI-11 PROGRAMMING RESTRICTIONS, HOWEVER; THE USER SHOULD READ SECTIONS 7.2 AND 7.3.

2.0 REQUIREMENTS  
-----

2.1 EQUIPMENT

1. PDP-11 FAMILY COMPUTER WITH 4K OF MEMORY (OR MORE) AND I/O FACILITIES (A SWITCH REGISTER OR TTY).
2. KVV11 UNDER TEST.

2.2 STORAGE

THIS PROGRAM OCCUPIES AND USES ONLY THE LOWER 4K OF MEMORY.

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

3.0 LOADING PROCEDURE

3.1 METHOD

STANDARDS PROCEDURE FOR NORMAL BINARY TAPES SHOULD BE FOLLOWED.

1. ABSOLUTE LOADER MUST BE IN MEMORY.
2. PLACE BINARY TAPE IN READER.
3. \*\* LOAD ADDRESS \*7500 (5\* DETERMINE BY LOCATION OF LOADER).
4. \*\* PRESS "START" (PROGRAM WILL BE LOADED INTO MEMORY).

THE PROGRAM CAN ALSO BE LOADED BY XXDP, ACT, OR APT.

\*\* THESE STEPS VARY ON AN LSI-11 (OR EQUAL) CPU. SEE SECTIONS 8.4 AND 8.5.

3.2 NON-STANDARD ADDRESS, VECTOR, OR USE OF SOFTWARE SWITCH REGISTER

THIS PROGRAM IS SET TO TEST A KVV11 WITH A STANDARD ADDRESS AND VECTOR. IF ANY OF THESE ARE DIFFERENT ON THE KVV11 YOU ARE TESTING, CHANGE THE CORRESPONDING LOCATION IN MEMORY BEFORE STARTING THIS TEST.

LOCATION	TAG	CURRENT CONTENTS	COMMENTS
1250	\$BASE:	170420	:: BASE ADDRESS OF EQUIPMENT :: UNDER TEST
1244	\$VECT1:	000440	:: INTERRUPT VECTOR #1
176	\$SWREG:	000000	:: MANUAL SWR.
1157	\$TPFLG:	.BYTE 0	:: "TERMINAL AVAILABLE" :: FLAG (BIT<0:7>=0=YES)

NOTE

IF NO HARDWARE SWITCH REGISTER EXISTS, YOU MAY SET ANY BIT IN "\$SWREG" AS YOU WOULD HAVE SET IT IN THE SWR.

192  
193  
194  
195  
196  
197  
198  
199  
200  
201  
202  
203  
204  
205  
206  
207  
208  
209  
210  
211  
212  
213  
214  
215  
216  
217  
218  
219  
220  
221  
222  
223  
224  
225  
226  
227  
228  
229  
230  
231  
232  
233  
234  
235  
236  
237  
238  
239  
240  
241  
242  
243  
244  
245  
246

4.0 STARTING PROCEDURE  
-----

4.1 CONTROL SWITCH SETTING

BEFORE STARTING THE DIAGNOSTIC, SET ALL SWITCH REGISTER BITS AS DESIRED, SEE SECTION 5.1.

4.2 STARTING ADDRESSES

200 START OF LOGIC TESTS  
204 RESTART ADDRESS FOR LOGIC TEST  
210 I/O SIGNAL TEST #1  
214 I/O SIGNAL TEST #2  
220 I/O SIGNAL TEST #3  
230 PRODUCTION STARTING ADDRESS  
240 TESTOR STARTING ADDRESS

4.3 PROGRAM AND/OR OPERATOR ACTION

4.3.1 WITH A HARDWARE SWITCH REGISTER

1. LOAD PROGRAM INTO MEMORY.
2. SET SWITCH REGISTER TO STARTING ADDRESS.
3. LOAD ADDRESS.
4. SET SWITCHES TO DESIRED SETTINGS - SEE SECTION 5.1.
5. PRESS START.

4.3.2 WITHOUT A HARDWARE SWITCH REGISTER

1. LOAD PROGRAM INTO MEMORY.
2. ENTER KEYBOARD "ODT".
3. ALTER LOCATION "SWREG" TO REFLECT DESIRED OPTIONS OF A SWITCH REGISTER - SEE SECTION 5.1.
4. TYPE STARTING ADDRESS, FOLLOWED BY "G" TO START PROGRAM.

27  
28  
29  
30  
31  
32  
33  
34  
35  
36  
37  
38  
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

5.0 OPERATING PROCEDURE

5.1 SWITCH REGISTER FUNCTION

SWR BIT	OCTAL	FUNCTION WHEN SET
15	100000	HALT ON ERROR
14	040000	LOOP ON TEST
13	020000	INHIBIT ERROR TYPEOUT
12	010000	ENABLE LINE FREQ. RATE TESTING
11	004000	INHIBIT ITERATIONS (SHORT PASS)
10	002000	BELL ON ERROR
09	001000	LOOP ON ERROR
08	000400	LOOP ON TEST IN SWR <7:0>

5.2 SCOPE LOOPS

5.2.1 SCOPE LOOPS WITH A HARDWARE SWITCH REGISTER

IF AN ERROR OCCURS AND THE USER WISHES TO SCOPE THE ERROR, HE (OR SHE) SHOULD SET SW15=1 TO HALT ON ERROR, THEN WHEN THE PROGRAM HALTS ON ERROR, SW15=0, SET SW14=1. TO LOOP ON CURRENT TEST, SET SW13=1 TO INHIBIT ERROR PRINTOUT, AND PRESS CONTINUE ON THE CPU'S CONSOLE.

5.2.2 SCOPE LOOPS WITHOUT A HARDWARE SWITCH REGISTER

IF AN ERROR OCCURS AND THE USER WISHES TO SCOPE THE ERROR, "\$SWREG" SHOULD BE ALTERED TO "100000" AT THE START OF THE TEST TO HALT ON ERROR, THEN WHEN THE PROGRAM HALTS ON ERROR AND THE CPU ENTERS "ODT", "\$SWREG" SHOULD BE ALTERED TO "060000" TO LOOP ON CURRENT TEST AND INHIBIT ERROR TYPEOUT, THEN TYPE "P" TO CONTINUE PROGRAM EXECUTION.

293  
294  
295  
296  
297  
298  
299  
300  
301  
302  
303  
304  
305  
306  
307  
308  
309  
310  
311  
312  
313  
314  
315  
316  
317  
318  
319  
320  
321  
322  
323  
324  
325  
326  
327  
328  
329  
330  
331  
332  
333

5.3 PROGRAM AND/OR OPERATOR ACTION

5.3.1 LOGIC TEST

THE FIRST PASS THROUGH THE PROGRAM WILL BE MADE WITH ITERATIONS INHIBITED. SUCCESSIVE PASSES WILL ENABLE ITERATIONS IF SWR11=0.

IF NOT INHIBITED BY APT, THE PROGRAM WILL LOOK FOR MORE KVV11'S TO EXERCISE, ONE PASS WILL EXERCISE ALL KVV11'S.

IF FOUR UNITS ARE DETECTED, THE FOLLOWING WILL BE TYPED:

UNIT #000001 COMPLETED TESTING UNIT #000002  
UNIT #000002 COMPLETED TESTING UNIT #000003  
UNIT #000003 COMPLETED TESTING UNIT #000004  
UNIT #000004 COMPLETED

AT END OF PASS WHEN ALL UNITS HAVE BEEN TESTED, THE FOLLOWING TYPEOUT WILL OCCUR:

"ENDPASS 12 - TOTAL ERRORS 4 - GOOD UNITS 000000000001011"

THIS INDICATES THAT THE PROGRAM HAS COMPLETED 12 OCTAL (10 DECIMAL) PASSES. DURING THAT TIME 4(OCTAL) ERRORS WERE DETECTED. ALSO WE TESTED 4 UNITS AND THE THIRD UNIT WAS THE ONLY UNIT TO FAIL.

5.4 INHIBITING AUTO-SIZE FEATURE

THIS PROGRAM WILL AUTOMATICALLY AUTO-SIZE AND TEST EACH KVV11 IT DETECTS ON THE SYSTEM. TO INHIBIT THIS FEATURE, SET BIT 15 OF LOCATION "SENV11". ALSO, TO TEST AN INDIVIDUAL KVV11 IN A GROUP, SET THIS BIT AND REFER TO SECTION 3.2 FOR CHANGING THE BASE ADDRESS OF THE KVV11 UNDER TEST.

334  
335  
336  
337  
338  
339  
340  
341  
342  
343  
344  
345  
346  
347  
348  
349  
350  
351  
352  
353  
354  
355  
356  
357  
358  
359  
360  
361  
362  
363  
364  
365  
366  
367  
368  
369  
370  
371  
372  
373  
374  
375  
376  
377  
378  
379  
380  
381  
382  
383

6.0 ERRORS  
-----

6.1 ERROR PRINTOUT

PRINTOUT VARIES WITH THE ERROR DETECTED. THE ERROR PC TYPED OUT IS THE ACTUAL LOCATION OF THE ERROR CALL.

A HALT AT LOCATION "STYPE"+10 WHEN RUNNING WITH NO TERMINAL INDICATES AN ERROR HAS OCCURRED. TO FIND OUT THE NUMBER OF THE ERROR, EXAMINE LOCATION "STSTNM". THIS IS THE ITEM NUMBER OF THE ERROR. TO FIND OUT WHAT THE ERROR TYPEOUT WOULD HAVE BEEN GOTO TO THE ERROR POINTER TABLE BEGINNING AT LOCATION "ERRTB".

6.1.1 EXAMPLE

IF WE EXAMINED LOCATION "STSTNM" AND FOUND A 5(101) WE GO TO LOCATION "ERRTB" AND LOOK THROUGH THE ERROR POINTER TABLE UNTIL WE FOUND ITEM 5. THE INFORMATION WOULD LOOK LIKE:

```
;ITEM 5
      EMS      :CLOCK SR DATA ERROR
      DHS      :ERRPC ASR WAS S/B
      DTS      :SERRPC,ASR,$BODAT,$GDOAT
      DFO      :ALL NUMBERS ARE IN OCTAL FORM
```

TO FIND OUT THE INFORMATION SPECIFIED BY DTS (SERRPC,BSR,\$BODAR,\$BODAR) FOLLOW THESE STEPS:

1. LOOK UP THE ADDRESS OF THE LABEL (I.E., SERRPC) IN THE SYMBOL TABLE WHICH FOLLOWS THE LISTING.
2. \* PUT THIS ADDRESS IN THE SWITCH REGISTER AND DEPRESS THE LOAD ADDRESS SWITCH ON THE PROCESSOR'S CONSOLE.
3. \* NOW DEPRESS THE EXAMINE SWITCH.
4. \* THE DATA DISPLAYED IN THE DATA LIGHTS IS THE INFORMATION THAT WOULD HAVE BEEN PRINTED FOR HIS LABEL IF YOU HAD A INPUT/OUTPUT TERMINAL.

\* SEE SECTION 8.4 FOR LSI-11 ODT COMMANDS.

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

## 6.2 NON-STANDARD ERROR HALTS

ANY HALT IN THE TRAP CATCHER AREA LOCATIONS 00C000-001C00, INDICATES:

1. THE KHV11 INTERRUPTED TO A WRONG VECTOR ADDRESS,  
OR
2. TIME-OUT OR ILLEGAL INSTRUCTION HARDWARE TRAP.

## 7.0 RESTRICTIONS

-----

### 7.1 EXTERNAL INPUTS

EXTERNAL INPUTS SUCH AS "SLAVE IN", "ST1" AND "ST2" MUST NOT BE CONNECTED TO ANY CUSTOMER HARDWARE THAT MIGHT GENERATE THESE SIGNAL WHILE THE DIAGNOSTIC IS RUNNING.

### 7.2 STARTING RESTRICTION

IF A FREE-RUNNING CLOCK, SUCH AS 60HZ FROM THE POWER SUPPLY, IS ATTACHED TO THE "BEVNT" BUS LINE ON BOTH REV LEVEL C/D AND E SYSTEMS, AN INTERRUPT TO LOCATION 100 WILL OCCUR WHEN USING THE "G" AND "L" COMMANDS PRIOR TO EXECUTING THE FIRST INSTRUCTION. THEREFORE THIS PROGRAM CAN NOT DISABLE THE BEVNT BUS LINE BY INHIBITING INTERRUPTS.

USER SYSTEMS REQUIRING A FREE-RUNNING CLOCK ATTACHED TO THE BEVNT BUS LINE CAN TEMPORARILY AVOID THIS SITUATION BY SETTING THE PSW(RS) TO 200, LOADING THE PC WITH THE STARTING ADDRESS INSTEAD OF USING THE "G" COMMAND, AND THEN USING THE "P" COMMAND. BEFORE USING THE "L" COMMAND, THE PSW(RS) CAN BE SET TO 200, THEREBY INHIBITING INTERRUPTS, TO AVOID RECEIVING THE EVENT INTERRUPT AFTER LOADING THE ABS LOADER.

4  
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

7.3 POSSIBLE PROGRAM "BOMBS"

THE FIRST TWO TESTS OF THIS PROGRAM CHECK TO SEE IF THE KVV11 RESPONDS TO THE ADDRESS THE PROGRAM THINKS ITS AT. IF THE KVV11 DOES NOT RESPOND, A BUS ERROR OCCURS. ALSO BUS ERRORS CAN OCCUR DURING THE TIME THE PROGRAM SIZES TO SEE HOW MANY KVV11 ARE ON YOU SYSTEM.

FOR MORE INFORMATION ON THE NEXT SUBJECT, SEE JAN. 1976 LSI-11 ENGINEERING BULLETIN ISSUED BY THE DIGITAL COMPONENTS GROUP.

BUS ERRORS MAY ALTER THE PRESET CONTENTS OF LOCATION 4 BEFORE THE TRAP IS EXECUTED, THEREBY TRANSFERRING PROGRAM CONTROL TO AREA IN THE PROGRAM THAT WAS NOT SET UP TO HANDLE THE TRAP. IF THIS HAPPENS, THE PROGRAM WILL "BOMB" AND POSSIBLY REWRITE PARTS OF ITSELF.

8.0 MISCELLANEOUS  
-----

8.1 POWER FAIL

AFTER A POWER FAILURE OCCURS, THE PROGRAM EXECUTION WILL CONTINUE AT THE POINT WHERE THE POWER OCCURRED. THE PROGRAM WILL TYPE "POWER".

8.2 XXDP, ACT, APT

THE PROGRAM IS CHAINABLE UNDER XXDP, ACT, OR APT. ALTHOUGH "APT HOOKS" HAVE BEEN INSTALLED, THEY HAVE NOT BEEN TESTED.

8.3 EXECUTION TIME

0.5 MINUTES (30 SEC) ITERATION INHIBITED - NO ERRORS  
2.5 MINUTES (150 SEC) WITH ITERATIONS - NO ERRORS

475  
476  
477  
478  
479  
480  
481  
482  
483  
484  
485  
486  
487  
488  
489  
490  
491  
492  
493  
494  
495  
496  
497  
498  
499  
500  
501  
502  
503  
504  
505  
506  
507  
508  
509  
510  
511  
512  
513  
514  
515  
516  
517  
518  
519  
520  
521  
522  
523  
524  
525  
526  
527  
528  
529  
530

8.4 LSI-11 "ODT" COMMANDS

<u>FORMAT</u>	<u>DESCRIPTION</u>
<CR> RETURN	CLOSE OPENED LOCATION AND ACCEPT NEXT COMMAND.
<LF> LINE FEED	CLOSE CURRENT LOCATION; OPEN NEXT SEQUENTIAL LOCATION.
↑(UPARROW)	OPEN PREVIOUS LOCATION.
← (LEFT ARROW)	TAKE CONTENTS OF OPENED LOCATION, INDEXED BY CONTENTS OF PC, AND OPEN THAT LOCATION.
⌘	TAKE CONTENTS OF OPENED LOCATION AS ABSOLUTE ADDRESS AND OPEN THAT LOCATION.
R/	OPEN THE WORD AT LOCATION R.
/	REOPEN THE LAST LOCATION.
SN/ OR RN/	OPEN GENERAL REGISTER N(0-7) OR S(PS REGISTER).
R;G OR RG	GOTO LOCATION R AND START PROGRAM.
NL	EXECUTE BOOTSTRAP LOADER USING N AS DEVICE CSR. CONSOLE DEVICE IS 177560.
;P OR P	PROCEED WITH PROGRAM EXECUTION.
RUBOUT	ERASES PREVIOUS NUMERIC CHARACTER. RESPONSE IS A BACKSLASH ( ).

8.5 ENTERING LSI-11 "ODT"

THE HALT OR ODT MICROCODE STATE OF THE KDIIF (LSI-11 MODULE) CAN BE ENTERED IN FIVE DIFFERENT WAYS (OTHERS ARE A SUBSET OF THESE) FROM THE RUN STATE:

1. EXECUTION OF A LSI-11 HALT INSTRUCTION,
2. A DOUBLE BUS ERROR,
3. AS A POWER UP OPTION,
4. ASCII BREAK WITH DLV11 FRAMING ERROR ASSERTING THE B HALT LINE (ENABLED BY JUMPER OF DLV11).

MO1

MAINDEC-11-DVKWA-A  
DVKWA.P11

MACY11 27(732) 04-OCT-76 14:57 PAGE 13

531  
532

UPON ENTERING THE HALT STATE, THE KD11F RESPONDS THROUGH THE SET  
OF COMMAND LISTED IN SECTION 8.4.

533  
534  
535  
536  
537  
538  
539  
540  
541  
542  
543  
544  
545  
546  
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  
585  
586  
588

8.6 USE OF PROGRAM SOFTWARE SWR

THE PROGRAM SOFTWARE SWITCH REGISTER IS ENABLED IF

1. NO HARDWARE SWR EXISTS;
2. IF YOU START WITH ALL ONES (SWR=177777) IN THE SWITCH REGISTER.

THE SOFTWARE SWITCH REGISTER MAY BE CHANGED BY TYPING  $\uparrow$ G (CONTROL AND LETTER G KEYS TYPED SIMULTANEOUSLY). WHEN  $\uparrow$ G IS TYPED, THE PROGRAM RESPONDS BY TYPING "SWR=XXXXXX" WHERE XXXXXX EQUALS THE FORMER CONTENTS OF THE SWITCH REGISTER.

IF YOU WISH TO KEEP THE CURRENT VALUE, TYPE <CR>. IF YOU WISH TO CHANGE THE VALUE, TYPE THE NEW VALUE FOLLOWED BY A <CR>.

IT IS IMPORTANT TO NOTE THAT THE DIAGNOSTIC IS NOT RUNNING AFTER THE  $\uparrow$ G UNTIL A <CR> IS TYPED.

8.7 SPECIAL I/O SIGNAL TESTS

THREE TESTS WERE INCLUDED TO ENABLE CHECKOUT OF I/O SIGNALS: ST1, ST2, AND CLOCK OVERFLOW. THESE TESTS HAVE A SPECIAL STARTING ADDRESS. SINCE END-PASSES ARE IMMEDIATE, NO "END OF PASS" MESSAGE IS REPORTED. ERRORS ARE REPORTED BY TYPING OUT THE PC WHERE THE ERROR WAS DETECTED. WHEN STARTED, THE PROGRAM REMAINS IN A LOOP GENERATING AND DETECTING THE SPECIFIED SIGNALS. HALT ON ERROR AND INHIBIT ERROR TIMEOUT OPTIONS MAY BE USED.

LOGIC TEST MUST HAVE ALREADY BEEN RUN ON THE K1V11.

8.7.1 I/O SIGNAL TEST #1 ST1 IN, ST2 OUT

SWITCH PACK S2 MUST BE SET UP AS FOLLOWS:

SWITCH STATE

```

-----
1    OFF
2    ON
3    OFF
4    OFF
5    ON
6    ON
7    NOT USED

```

THE FOLLOWING JUMPER MUST BE INSTALLED.

MAINDEC-11-DVKWA-A  
DVKWA.P11

MACY11 27(732) 04-OCT-76 14:57 PAGE 15

802

589

J1-SS (ST2 OUT) TO J1-WV (ST1 IN)

590  
591  
592  
593  
594  
595  
596  
597  
598  
599  
600  
601  
602  
603  
604  
605  
606  
607  
608  
609  
610  
611  
612  
613  
614  
615  
616  
617  
618  
619  
620  
621  
622  
623  
624  
625  
626  
627  
628  
629  
630  
631  
632  
633  
634  
635  
636  
637  
638  
639

LOAD AND START THE PROGRAM AT 210.

8.7.2 I/O SIGNAL TEST #2 CLOCK OVERFLOW TEST  
SWITCH PACK S2 MUST BE SET UP AS FOLLOWS:

SWITCH STATE

1	OFF
2	OFF
3	OFF
4	OFF
5	OFF
6	OFF
7	NOT USED

THE FOLLOWING JUMPER MUST BE INSTALLED.

J1-RR (CLOCK OVERFLOW) TO J1-TT (ST2 IN)

LOAD AND START AT LOCATION 214.

8.7.3 I/O SIGNAL TEST #3 ST1 OUT AND ST2 IN  
SWITCH PACK S2 MUST BE SET UP AS FOLLOWS:

SWITCH STATE

1	OFF
2	OFF
3	OFF
4	ON
5	ON
6	ON
7	NOT USED

THE FOLLOWING JUMPER MUST BE INSTALLED:

J1-UU (ST1 OUT) TO J1-TT (ST2 IN)

LOAD AND START AT LOCATION 220.

640  
641  
642  
643  
644  
645  
646  
647  
648  
649  
650  
651  
652  
653  
654  
655  
656  
657  
658  
659  
660  
661  
662  
663  
664  
665  
666  
667  
668  
669  
670  
671  
672  
673  
674  
675  
676  
677  
678  
679  
680  
681  
682  
683  
684  
685  
686  
687  
688  
689  
690  
691  
692  
693  
694  
695

8.8 PRODUCTION STARTING ADDRESS

A SPECIAL STARTING ADDRESS HAS BEEN PROVIDED FOR IN-HOUSE PRODUCTION TO USE TO START THE LOGIC DIAGNOSTIC AND INFORM THE TEST THAT PRODUCTION IS USING IT.

IN THE FIELD ONLY ENOUGH ADDRESSES WERE ALLOTTED FOR 4 SEQUENTIAL KWV11S. WHEN THE LOGIC TESTS ARE STARTED AT LOCATION 200, WE ONLY AUTO-SIZE UP TO 4 KWV11S.

IN HOUSE TESTING MAY WISH TO EXERCISE UP TO 16 KWV11S AT ONE TIME. THE LOGIC TESTS MAY BE STARTED AT LOCATION 230 AND THE PROGRAM WILL AUTO SIZE UP TO 16 KWV11S.

8.9 TESTOR STARTING ADDRESS

A SPECIAL STARTING ADDRESS HAS BEEN PROVIDED FOR MANUFACTURING TO USE TO START THE LOGIC DIAGNOSTIC AND INFORM THE PROGRAM THAT THE CLOCK MODULE IS CABLED TO AN IN-HOUSE TESTOR.

MANUAL INTERVENTION IS NEEDED IN THIS SEQUENCE OF TESTING. THE PROGRAM WILL TYPE OUT ALL INSTRUCTIONS. A CABLE SHOULD CONNECT J1 ON THE CLOCK MODULE TO J10 ON THE TESTOR. SWITCHES 1 AND 3 OF S2 (ON THE CLOCK MODULE) SHOULD BE ON, ALL OTHER SWITCHES ON S2 SHOULD BE OFF.

```
%  
.MLIST MC,MD,CND  
.LIST ME  
.ENABL ABS  
.ENABL AMA  
.MCALL .HEADER,.SETUP,.SETTRAP,.TRNTRP,.$STRAP,.$RDOCT,.$STYPBIN  
.MCALL TYPOCS,.$POWER,.$SCATCH,.$STYPOCT,.$EQUAT,.$CMTAG,.$SWRHI  
.MCALL .SEOP,.$ERROR,.$ERRTYP,.$STYPDEC,.$SCOPE,.$SREAD,.$STYPE  
.MCALL .$ACT11,.$SAPTHOR,.$SAPTYPE  
$SWR= 167400
```

167400

```
.TITLE MAINDEC-11-DVKWA-A  
*COPYRIGHT (C) 1976  
*DIGITAL EQUIPMENT CORP.  
*MAYNARD, MASS. 01754  
*  
*PROGRAM BY EDWARD C. BADGER  
*  
*THIS PROGRAM WAS ASSEMBLED USING THE POP-11 MAINDEC SYSMAC  
*PACKAGE (MAINDEC-11-DZQAC-CO), MAR 21, 1976.
```

```

696          000001      ;*
697          ;$TN=1
698
699 000000      ;SWRH:
700          .SBTTL TRAP CATCHER
701
702          000000      ;=0
703          ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A "+2,HALT"
704          ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
705          ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
706          000174      ;=174
707 000174 000000      DISPRG: .WORD 0      ;; SOFTWARE DISPLAY REGISTER
708 000176 000000      SWREG:  .WORD 0      ;; SOFTWARE SWITCH REGISTER
709          000100      ;=100
710 000100 000102 000002 .WORD 102,2      ; IF "B EVENT" ON Q-BUS IS CONNECTED
711          ; WE NEED WAY OF IGNORING ITS INTERRUPTS.
712
713          000200      ;=200
714 000200 000137 001472      JMP      @#START
715 000204 000137 002070      JMP      @#RSTART
716 000210 000137 013322      JMP      @#IOTST1
717 000214 000137 013400      JMP      @#IOTST2
718 000220 000137 013446      JMP      @#IOTST3
719
720          000230      ;=230
721 000230 000137 001456      JMP      @#WSTART      ; WESTFIELD STARTING ADDRESS
722          000240      ;=240
723 000240 000137 001442      JMP      @#TSTSTR      ; ALL TESTER TESTS
724          ; IF STARTED HERE.
725          ; ALLOWS PRODUCTION TO EXERCISE
726          ; UP TO 16 CLOCKS.NORMAL=4.
727
728          .SBTTL BASIC DEFINITIONS
729
730          ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
731          001100      STACK= 1100
732          .EQUIV EMT,ERROR      ;; BASIC DEFINITION OF ERROR CALL
733          .EQUIV IOT,SCOPE      ;; BASIC DEFINITION OF SCOPE CALL
734
735          ;*MISCELLANEOUS DEFINITIONS
736          000011      HT= 11      ;; CODE FOR HORIZONTAL TAB
737          000012      LF= 12      ;; CODE FOR LINE FEED
738          000015      CR= 15      ;; CODE FOR CARRIAGE RETURN
739          000200      CRLF= 200      ;; CODE FOR CARRIAGE RETURN-LINE FEED
740          177776      PS= 177776      ;; PROCESSOR STATUS WORD
741          .EQUIV PS,PSW
742          177774      STKLMT= 177774      ;; STACK LIMIT REGISTER
743          177772      PIRQ= 177772      ;; PROGRAM INTERRUPT REQUEST REGISTER
744          177570      DSWR= 177570      ;; HARDWARE SWITCH REGISTER
745          177570      DOISP= 177570      ;; HARDWARE DISPLAY REGISTER
746
747          ;*GENERAL PURPOSE REGISTER DEFINITIONS
748          000000      R0= %0      ;; GENERAL REGISTER
749          000001      R1= %1      ;; GENERAL REGISTER
750          000002      R2= %2      ;; GENERAL REGISTER
751          000003      R3= %3      ;; GENERAL REGISTER

```

752	000004	R4=	%4	:: GENERAL REGISTER
753	000005	R5=	%5	:: GENERAL REGISTER
754	000006	R6=	%6	:: GENERAL REGISTER
755	000007	R7=	%7	:: GENERAL REGISTER
756		.EQUIV	R6, SP	:: STACK POINTER
757		.EQUIV	R7, PC	:: PROGRAM COUNTER

758

759 : \*PRIORITY LEVEL DEFINITIONS

760	000000	PR0=	0	:: PRIORITY LEVEL 0
761	000040	PR1=	40	:: PRIORITY LEVEL 1
762	000100	PR2=	100	:: PRIORITY LEVEL 2
763	000140	PR3=	140	:: PRIORITY LEVEL 3
764	000200	PR4=	200	:: PRIORITY LEVEL 4
765	000240	PR5=	240	:: PRIORITY LEVEL 5
766	000300	PR6=	300	:: PRIORITY LEVEL 6
767	000340	PR7=	340	:: PRIORITY LEVEL 7

768

769 : \*"SWITCH REGISTER" SWITCH DEFINITIONS

770	100000	SW15=	100000
771	040000	SW14=	40000
772	020000	SW13=	20000
773	010000	SW12=	10000
774	004000	SW11=	4000
775	002000	SW10=	2000
776	001000	SW09=	1000
777	000400	SW08=	400
778	000200	SW07=	200
779	000100	SW06=	100
780	000040	SW05=	40
781	000020	SW04=	20
782	000010	SW03=	10
783	000004	SW02=	4
784	000002	SW01=	2
785	000001	SW00=	1
786		.EQUIV	SW09, SW9
787		.EQUIV	SW08, SW8
788		.EQUIV	SW07, SW7
789		.EQUIV	SW06, SW6
790		.EQUIV	SW05, SW5
791		.EQUIV	SW04, SW4
792		.EQUIV	SW03, SW3
793		.EQUIV	SW02, SW2
794		.EQUIV	SW01, SW1
795		.EQUIV	SW00, SW0

796

797 : \*DATA BIT DEFINITIONS (BIT00 TO BIT15)

798	100000	BIT15=	100000
799	040000	BIT14=	40000
800	020000	BIT13=	20000
801	010000	BIT12=	10000
802	004000	BIT11=	4000
803	002000	BIT10=	2000
804	001000	BIT09=	1000
805	000400	BIT08=	400
806	000200	BIT07=	200
807	000100	BIT06=	100

808 000040  
809 000020  
810 000010  
811 000004  
812 000002  
813 000001  
814  
815  
816  
817  
818  
819  
820  
821  
822  
823  
824  
825  
826 000004  
827 000010  
828 000014  
829 000014  
830 000014  
831 000020  
832 000024  
833 000030  
834 000034  
835 000060  
836 000064  
837 000240  
838  
839 170420  
840 000440  
841 000200  
842  
843 167400  
844 000001  
845  
846  
847  
848  
849  
850 000244  
851 000046  
852 000046 013302  
853 000052 000052  
854 000052 000000  
855 000244 000244  
856 001000 001000  
857  
858  
859  
860  
861  
862 001000  
863 000024

BIT05= 40  
BIT04= 20  
BIT03= 10  
BIT02= 4  
BIT01= 2  
BIT00= 1  
.EQUIV BIT09,BIT9  
.EQUIV BIT08,BIT8  
.EQUIV BIT07,BIT7  
.EQUIV BIT06,BIT6  
.EQUIV BIT05,BIT5  
.EQUIV BIT04,BIT4  
.EQUIV BIT03,BIT3  
.EQUIV BIT02,BIT2  
.EQUIV BIT01,BIT1  
.EQUIV BIT00,BIT0

::#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  
TPVEC= 64 ;: TTY PRINTER VECTOR  
PIRQVEC=240 ;: PROGRAM INTERRUPT REQUEST VECTOR

ABASE= 170420  
AVECT1= 440  
APRIOR= 200  
SSWR= 167400  
STN= 1

.SBTTL ACT11 HOOKS

::\*\*\*\*\*  
;HOOKS REQUIRED BY ACT11  
\$SVPC= ;SAVE PC  
.=46  
SENDAD ;:1)SET LOC.46 TO ADDRESS OF SENDAD IN .SEOP  
.=52  
.WORD 0 ;:2)SET LOC.52 TO ZERO  
.\$SVPC ;: RESTORE PC  
.=1000

.SBTTL APT PARAMETER BLOCK

::\*\*\*\*\*  
;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT  
::\*\*\*\*\*  
.\$X= ;:SAVE CURRENT LOCATION  
.=24 ;:SET POWER FAIL TO POINT TO START OF PROGRAM

864	000024	000200
865		000044
866	000044	001000
867		001000
868		
869		
870		
871		
872	001000	
873	001000	000000
874	001002	001174
875	001004	000002
876	001006	000170
877	001010	000170
878	001012	000031

```

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-POP11 DIAGNOSTIC
; INTERFACE SPEC.

$APTHD:
$HIBTS: .WORD 0      ;; TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
$MBAOR: .WORD $MAIL  ;; ADDRESS OF APT MAILBOX (BITS 0-15)
$TSTM:  .WORD 2      ;; RUN TIM OF LONGEST TEST
$PASTM: .WORD 120.   ;; RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
$UNITM: .WORD 120.   ;; ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
          .WORD $ETEND-$MAIL/2 ;; LENGTH MAILBOX-ETABLE (WORDS)

```

879  
880  
881  
882  
883  
884  
885 001100  
886 001100  
887 001100 000000  
888 001102 000  
889 001103 000  
890 001104 000000  
891 001106 000000  
892 001110 000000  
893 001112 000000  
894 001114 000  
895 001115 001  
896 001116 000000  
897 001120 000000  
898 001122 000000  
899 001124 000000  
900 001126 000000  
901 001130 000000  
902 001132 000000  
903 001134 000  
904 001135 000  
905 001136 000000  
906 001140 177570  
907 001142 177570  
908 001144 177560  
909 001146 177562  
910 001150 177564  
911 001152 177566  
912 001154 000  
913 001155 002  
914 001156 012  
915 001157 000  
916 001160 000000  
917 001162 000000  
918 001164 177607 000377  
919 001170 077  
920 001171 015  
921 001172 000012  
922  
923  
924  
925  
926  
927 001174  
928 001174 000000  
929 001176 000000  
930 001200 000000  
931 001202 000000  
932 001204 000000  
933 001206 000000  
934 001210 000000

.SBTTL COMMON TAGS

\*\*\*\*\*  
\*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS  
\*USED IN THE PROGRAM.

. =1100

\$CMTAG: ; START OF COMMON TAGS  
 \$STNM: .WORD 0 ; CONTAINS THE TEST NUMBER  
 \$ERFLG: .BYTE 0 ; CONTAINS ERROR FLAG  
 \$ICNT: .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  
 \$GDADR: .WORD 0 ; CONTAINS 'GOOD' DATA  
 \$BDADR: .WORD 0 ; CONTAINS 'BAD' DATA  
 ; RESERVED--NOT TO BE USED  
 \$AUTOB: .BYTE 0 ; AUTOMATIC MODE INDICATOR  
 \$INTAG: .BYTE 0 ; INTERRUPT MODE INDICATOR  
 \$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)  
 \$TIMES: 0 ; MAX. NUMBER OF ITERATIONS  
 \$ESCAPE: 0 ; ESCAPE ON ERROR ADDRESS  
 \$BELL: .ASCIZ <207><377><377> ; CODE FOR BELL  
 \$QUES: .ASCII '?' ; QUESTION MARK  
 \$CRLF: .ASCII <15> ; CARRIAGE RETURN  
 \$LF: .ASCIZ <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

935	001212	000000	\$MSGLG: .WORD	RMSGLG	:: MESSAGE LENGTH
936	001214		\$ETABLE:		:: APT ENVIRONMENT TABLE
937	001214	000	\$ENV: .BYTE	AENV	:: ENVIRONMENT BYTE
938	001215	000	\$ENVM: .BYTE	AENVM	:: ENVIRONMENT MODE BITS
939	001216	000000	\$SWREG: .WORD	ASWREG	:: APT SWITCH REGISTER
940	001220	000000	\$USWA: .WORD	AUSWA	:: USER SWITCHES
941	001222	000000	\$CPUOP: .WORD	ACPUOP	:: CPU TYPE, OPTIONS
942			*		BITS 15-11=CPU TYPE
943			*		11/04=01, 11/05=02, 11/20=03, 11/40=04, 11/45=05
944			*		11/70=06, PDQ=07, Q=10
945			*		BIT 10=REAL TIME CLOCK
946			*		BIT 9=FLOATING POINT PROCESSOR
947			*		BIT 8=MEMORY MANAGEMENT
948	001224	000	\$MAMS1: .BYTE	AMAMS1	:: HIGH ADDRESS, M.S. BYTE
949	001225	000	\$MTYP1: .BYTE	AMTYP1	:: MEM. TYPE, BLK#1
950			*		MEM. TYPE BYTE -- (HIGH BYTE)
951			*		900 NSEC CORE=001
952			*		300 NSEC BIPOLAR=002
953			*		500 NSEC MOS=003
954	001226	000000	\$MADR1: .WORD	AMADR1	:: HIGH ADDRESS, BLK#1
955			*		MEM. LAST ADDR.=3 BYTES, THIS WORD AND LOW OF "TYPE" ABOVE
956	001230	000	\$MAMS2: .BYTE	AMAMS2	:: HIGH ADDRESS, M.S. BYTE
957	001231	000	\$MTYP2: .BYTE	AMTYP2	:: MEM. TYPE, BLK#2
958	001232	000000	\$MADR2: .WORD	AMADR2	:: MEM. LAST ADDRESS, BLK#2
959	001234	000	\$MAMS3: .BYTE	AMAMS3	:: HIGH ADDRESS, M.S. BYTE
960	001235	000	\$MTYP3: .BYTE	AMTYP3	:: MEM. TYPE, BLK#3
961	001236	000000	\$MADR3: .WORD	AMADR3	:: MEM. LAST ADDRESS, BLK#3
962	001240	000	\$MAMS4: .BYTE	AMAMS4	:: HIGH ADDRESS, M.S. BYTE
963	001241	000	\$MTYP4: .BYTE	AMTYP4	:: MEM. TYPE, BLK#4
964	001242	000000	\$MADR4: .WORD	AMADR4	:: MEM. LAST ADDRESS, BLK#4
965	001244	000440	\$VECT1: .WORD	AVECT1	:: INTERRUPT VECTOR#1, BUS PRIORITY#1
966	001246	000000	\$VECT2: .WORD	AVECT2	:: INTERRUPT VECTOR#2, BUS PRIORITY#2
967	001250	170420	\$BASE: .WORD	ABASE	:: BASE ADDRESS OF EQUIPMENT UNDER TEST
968	001252	000000	\$DEVN: .WORD	ADEVN	:: DEVICE MAP
969	001254	000000	\$CDW1: .WORD	ACDW1	:: CONTROLLER DESCRIPTION WORD#1
970	001256		\$ETEND:		
971			.MEXIT		

.SB\*TL 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 SITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.  
;\*NOTE1: IF SITEMB IS 0 THE ONLY PERTINENT DATA IS (SERRPC).  
;\*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

SERRTB:

;ITEM 1

972  
973  
974  
975  
976  
977  
978  
979  
980  
981  
982  
983  
984  
985  
986 001256  
987  
988  
989  
990  
991 001256 016364 EM1 ;CLOCK SR FUNCTION ERROR  
992 001260 016716 DH1 ;ERRPC ASR WAS S/B  
993 001262 017114 DT1 ;SERRPC, ASR, SBDDAT, SGDDAT  
994 001264 017202 DF0 ;ALL NUMBERS ARE IN OCTAL FORM  
995  
996  
997

;ITEM 2

998  
999 001266 016416 EM2 ;CLOCK SR DATA ERROR  
1000 001270 016716 DH1 ;ERRPC ASR WAS S/B  
1001 001272 017114 DT1 ;SERRPC, ASR, SBDDAT, SGDDAT  
1002 001274 017202 DF0 ;ALL NUMBERS ARE IN OCTAL FORM  
1003  
1004

;ITEM 3

1005  
1006  
1007 001276 016444 EM3 ;CLOCK BR DATA ERROR  
1008 001300 016742 DH3 ;ERRPC ABR WAS  
1009 001302 017126 DT3 ;SERRPC, ABR, SBDDAT, SGDDAT  
1010 001304 017202 DF0 ;ALL NUMBERS ARE IN OCTAL FORM  
1011  
1012

;ITEM 4

1013  
1014  
1015 001306 000000 0 ;RESERVED FOR FUTURE ERROR.  
1016 001310 000000 0 ;RESERVED.  
1017 001312 000000 0  
1018 001314 017202 DF0 ;ALL NUMBERS ARE IN OCTAL FORM  
1019  
1020

;ITEM 5

1021  
1022  
1023 001316 016472 EM5 ;CLOCK COUNT REG ERROR  
1024 001320 016766 DH4 ;ERRPC ACR WAS S/B  
1025 001322 017114 DT1 ;SERRPC, ACR, SBDDAT, SGDDAT  
1026 001324 017202 DF0 ;ALL NUMBERS ARE IN OCTAL FORM  
1027

1028					
1029			;ITEM	6	
1030					
1031	001326	016542		EM12	;CLOCK COUNT FUNCTION ERROR
1032	001330	017012		DH12	;ERRPC ASR
1033	001332	017140		DT12	;ERRPC ASR
1034	001334	017202		DFO	;ALL NUMBERS ARE IN OCTAL FORM
1035					
1036					
1037			;ITEM	7	
1038					
1039	001336	016577		EM16	;CLOCK INTERRUPT ERROR
1040	001340	017012		DH12	;ERRPC ASR
1041	001342	017140		DT12	;ERRPC ASR
1042	001344	017202		DFO	;ALL NUMBERS ARE IN OCTAL FORM
1043					
1044					
1045			;ITEM	10	
1046					
1047	001346	016630		EM20	;CLOCK REPEATABILITY ERROR
1048	001350	017027		DH20	;ERROR ASR 2ND CNT 1ST CNT 3RD CNT
1049	001352	017146		DT20	;ERRPC ASR \$BDDAT \$GDDAT \$TMPO
1050	001354	017202		DFO	;ALL NUMBERS ARE IN OCTAL FORM
1051					
1052					
1053			;ITEM	11	
1054					
1055	001356	016515		EM11	;CLOCK COUNT ERROR
1056	001360	016766		DH4	;ERRPC ASR WAS S/B
1057	001362	017162		DT22	;ERRPC ASR \$BDDAT \$TMPO
1058	001364	017202		DFO	;ALL NUMBERS ARE IN OCTAL FORM
1059					
1060					
1061			;ITEM	12	
1062					
1063	001366	016665		EM26	;CLOCK ADDRESSING ERROR
1064	001370	017070		DH26	;ERRPC CLOCK ADDR.
1065	001372	017174		DT26	;ERRPC \$TMPO
1066	001374	017202		DFO	;ALL NUMBERS ARE IN OCTAL FORM
1067					
1068					
1069	001376	170420	ASR:	.WORD	ABASE
1070	001400	170422	ASR:	.WORD	ABASE+2
1071	001402	000440	VECT1:	.WORD	AVECT1
1072	001404	000442	VECTP:	.WORD	AVECT1+2
1073	001406	000444	VECT2:	.WORD	AVECT1+4
1074	001410	000446	VECT2P:	.WORD	AVECT1+6
1075	001412	000200	PRIOR:	.WORD	APRIOR
1076	001414	167774	DR:	.WORD	167774
1077	001416	167772	DR2:	.WORD	167772
1078	001420	000000	\$TMPO:	.WORD	0
1079	001422	000000	\$TMP1:	.WORD	0
1080	001424	000000	\$TMP3:	.WORD	0
1081	001426	000000	ROTATE:	.WORD	0
1082	001430	000000	UTEST:	.WORD	0
1083	001432	000000	ERCNT:	.WORD	0

; VECTOR ADDR. OF ST2 INTR.

; TEMP STORAGE.  
; TMP STORAGE.

; POINT TO DEVICE UNDER TEST.  
; KEEPS TRACK OF GOOD UNITS.  
; COUNTS ERRORS.

# MO2

```

1084 001434 000000 MDEVCT: .WORD - ;COUNTS DEVICES TESTED.
1085 001436 000000 TSTCNT: .WORD 0 ;MAX DEVICES TO BE TESTED.
1086 001440 000000 EXS: .WORD 0 ;=0, NORMAL: =1 SPECIAL TESTOR START, BY L+S 2 2
1087
1088
1089 001442 005237 001440 TSTSTR: INC EXS ;SET FOR TESTOR.
1090 001446 012737 000020 001436 MOV #16.,TSTCNT ;ALLOW 16 UNITS
1091 001454 000413 BR 1$
1092
1093 001456 012737 000020 001436 WSTART=. MOV #16.,TSTCNT ;TEST UP TO 16 UNITS.
1094 001464 005037 001440 CLR EXS
1095 001-70 000405 BR 1$
1096
1097 001472 012737 000004 001436 START=. MOV #4,TSTCNT ;TEST UP TO FOUR UNITS.
1098 001500 005037 001440 CLR EXS
1099
1100
1101 .SBTTL INITIALIZE THE COMMON TAGS
1102 ;;CLEAR THE COMMON TAGS (SCMTAG) AREA
1103 MOV #SCMTAG,R6 ;;FIRST LOCATION TO BE CLEARED
1104 CLR (R6)+ ;;CLEAR MEMORY LOCATION
1105 CMP #SWR,R6 ;;DONE?
1106 BNE -6 ;;LOC? BACK IF NO
1107 MOV #STACK,SP ;;SETUP THE STACK POINTER
1108 ;;INITIALIZE A FEW VECTORS
1109 MOV #SCOPE,#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
1110 MOV #340,#IOTVEC+2 ;;LEVEL 7
1111 MOV #ERROR,#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
1112 MOV #340,#EMTVEC+2 ;;LEVEL 7
1113 MOV #TRAP,#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
1114 MOV #340,#TRAPVEC+2 ;;LEVEL 7
1115 MOV #SPWRDN,#PWRVEC ;;POWER FAILURE VECTOR
1116 MOV #340,#PWRVEC+2 ;;LEVEL 7
1117 CLR #TIMES ;;INITIALIZE NUMBER OF ITERATIONS
1118 CLR #ESCAPE ;;CLEAR THE ESCAPE ON ERROR ADDRESS
1119 MOV #1,#SERMAX ;;ALLOW ONE ERROR PER TEST
1120 MOV #.,#SLPADR ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
1121 MOV #.,#SLPERR ;;SETUP THE ERROR LOOP ADDRESS
1122 ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
1123 ;;EQUAL TO A "-1" SETUP FOR A SOFTWARE SWITCH REGISTER.
1124 MOV #ERRVEC,-(SP) ;;SAVE ERROR VECTOR
1125 MOV #64$,#ERRVEC ;;SET UP ERROR VECTOR
1126 MOV #DSWR,#SWR ;;SETUP FOR A HARDWARE SWICH REGISTER
1127 MOV #DDISP,#DISPLAY ;;AND A HARDWARE DISPLAY REGISTER
1128 CMP #-1,#SWR ;;TRY TO REFERENCE HARDWARE SWR
1129 BNE 66$ ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
1130 BR 65$ ;;AND THE HARDWARE SWR IS NOT = -1
1131 64$: MOV #65$,(SP) ;;BRANCH IF NO TIMEOUT
1132 RTI ;;SET UP FOR TRAP RETURN
1133 65$: MOV #SWREG,#SWR ;;POINT TO SOFTWARE SWR
1134 MOV #DISPREG,#DISPLAY
1135 66$: MOV (SP)+,#ERRVEC ;;RESTORE ERROR VECTOR
1136
1137 CLR #PASS ;;CLEAR PASS COUNT
1138 BITB #APTSIZE,#ENVM ;;TEST USER SIZE UNDER APT
1139 BEQ 67$ ;;YES,USE NON-APT SWITCH
  
```

```

1140 001740 012737 001216 001140      MOV      #SSWREG,SWR      ;;NO,USE APT SWITCH REGISTER
1141 001746                                67$:
1142
1143
1144 001746 012746 000340      MOV      #340,-(SP)      ;SET CPU PRIORITY ON RETURN.
1145 001752 012746 001760      MOV      #68$,-(SP)      ;SHOW RETURN ADDRESS.
1146 001756 000002      RTI                          ;CAUSE A RETURN(PUTS STATUS IN STATUS REG.).
1147 001760                                68$:
1148
1149 001760 013737 001244 001402      MOV      $VECT1,VECT1    ;NOW FIX VECTOR ADDR.
1150 001766 013737 001250 001376      MOV      $BASE,ASR      ;FIX ADDRESS OF CSR.
1151
1152      .SBTTL  TYPE PROGRAM NAME
1153      ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
1154 001774 005227 177777      INC      #-1              ;FIRST TIME?
1155 002000 001033      BNE      69$              ;BRANCH IF NO
1156 002002 104400 002050      TYPE     70$              ;TYPE ASCIZ STRING
1157      .SBTTL  GET VALU FOR SOFTWARE SWITCH REGISTER
1158 002006 005737 000042      TST     #42              ;ARE WE RUNNING UNDER XXDP/ACT?
1159 002012 001012      BNE      71$              ;BRANCH IF YES
1160 002014 123727 001214 000001      CMPB    $ENV,#1          ;ARE WE RUNNING UNDER APT?
1161 002022 001406      BEQ     71$              ;BRANCH IF YES
1162 002024 023727 001140 000176      CMP     SWR,#SWREG      ;SOFTWARE SWITCH REG SELECTED?
1163 002032 001005      BNE      72$              ;BRANCH IF NO
1164 002034 104405      GTSWR                                ;GET SOFT-SWR SETTINGS
1165 002036 000403      BR     72$
1166 002040 112737 000001 001134 71$:  MOVB    #1,$AUTOB      ;;SET AUTO-MODE INDICATOR
1167 002046                                72$:
1168 002046 000410      BR     69$              ;GET OVER THE ASCIZ
1169      ;;70$: .ASCIZ <CRLF>#MD11-DVKWA-A<CRLF>
1170 002070      69$:
1171 002070      RSTART:
1172 002070 005737 001440      TST     EXS              ;TESTOR MODE ENABLED??
1173 002074 001441      BEQ     1$              ;NO DON'T TYPE NEXT MESSAGE.
1174 002076 104400 002104      TYPE     65$              ;TYPE ASCIZ STRING
1175 002102 000436      BR     64$              ;GET OVER THE ASCIZ
1176      ;;65$: .ASCIZ <15><12>#TESTOR MODE ENABLED--SEE DOCUMENTATION FOR INSTRUCTIONS.#
1177 002200      64$:
1178 002200      1$:
1179 002200 104400 002206      TYPE     67$              ;TYPE ASCIZ STRING
1180 002204 000411      BR     66$              ;GET OVER THE ASCIZ
1181      ;;67$: .ASCIZ <15><12>#TEST RUNNING...#
1182 002230      66$:
1183 002230 005037 001434      CLR     MDEVCT          ;TESTING FIRST UNIT.
1184 002234 005037 001432      CLR     ERcnt           ;NO ERRORS.
1185 002240 005037 001202      CLR     $PASS          ;NO PASSES.
1186 002244 012737 000001 001426      MOV     #1,ROTATE      ;POINT TO FIRST UNIT.
1187 002252 013737 001426 001430      MOV     ROTATE,UTEST
1188 002260                                LOOP:
1189
1190 002260 042737 170000 001402      BIC     #170000,VECT1   ;CLEAR OUT PRIORITY BITS.
1191 002266 013737 001402 001404      MOV     VECT1,VECTP    ;NOW FIX VECTOR +2 ADDR.
1192 002274 062737 000002 001404      ADD     #2,VECTP
1193 002302 013737 001402 001406      MOV     VECT1,VECT2    ;LETS FIX ST2 VECTOR ADDR.
1194 002310 062737 000004 001406      ADD     #4,VECT2      ;ITS 4 GREATER THEN THE 1ST.
1195 002316 013737 001406 001410      MOV     VECT2,VECT2P   ;VECTOR +2 ADDR.
  
```

```

1196 002324 062737 000002 001410      AUD      #2, VECT2P
1197
1198
1199 002332 013737 001376 001400      MOV      ASR, ABR      ;FIX ADDR OF PRESET REG=
1200 002340 062737 000002 001400      ADD      #2, ABR      ;CSR + 2
1201
1202
1203
1204
1205
1206
1207 002346 000240
1208 002350 012737 000050 001160
1209 002356 012737 002400 001106
1210 002364 112737 000001 001102
1211 002372 012737 002400 001110
1212
1213
1214
1215 002400 013746 0010004 1S:      MOV      2#ERRVEC, -(SP) ;SAVE CONTENTS OF ADDRS 6.
1216 002404 012737 002420 000004      MOV      #25, 2#ERRVEC ;SET TIME-OUT TRAP VECTOR TO HANDLER IN CASE.
1217
1218
1219 002412 005777 1176760      TST      2ASR          ;ADDRESS THE CLOCK!
1220
1221
1222
1223
1224
1225
1226 002420 062706 000004 2S:
1227 002424 013737 001376 001420      ADD      #4, SP          ;/ADD #4 TO STACK POINTER.
1228
1229
1230
1231
1232 002432 104012      ERROR 12 ;REPORT ERROR=CLOCK CSR FAILED TO RETURN
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242 002434 012637 000004 3S:      MOV      (SP)+, 2#ERRVEC
1243

```

;\*\*\*\*\*  
;TEST 1 \*TEST THE ADDRESSABILITY OF CLOCK CSR  
;\*\*\*\*\*  
↑ST1: NOP  
MOV #50, \$TIMES ;DO 50 ITERATIONS  
MOV #1\$, \$LPAOR ;SET SCOPE LOOP ADDRESS  
MOV #1, \$TSTNM ;SET TEST #1.  
MOV #1\$, \$LPERA  
  
;THE CLOCK WAS THERE! EXIT SUB-TEST.  
  
;/ADD #4 TO STACK POINTER.  
;FOR ERROR TYPEOUT.  
  
;\*\*\*\*\*  
;\*\*\*\*\*  
;\*\*\*\*\*  
;\*\*\*\*\*  
;\*\*\*\*\*  
;\*\*\*\*\*

```

1274
1275
1276
1277
1278 002440 000004
1279
1280
1281 002442 013746 000004 1S:  MOV  @#ERRVEC, -(SP) ;SAVE CONTENTS OF ADDRS 6.
1282 002446 012737 002462 000004  MOV  #2S,@#ERRVEC ;SET TIME-OUT TRAP VECTOR TO HANDLER IN CASE.
1283
1284
1285 002454 005777 176720      TST  @ABR ;ADDRESS THE CLOCK!
1286
1287
1288
1289 002460 000406      BR   3S ;THE CLOCK WAS THERE! EXIT SUB-TEST.
1290
1291 002462      2S:
1292 002462 062706 000004      ADD  #4,SP ;/ADD #4 TO STACK POINTER.
1293 002466 013737 001400 001420  MOV  ABR,$TMP0 ;FOR ERROR TYPEOUT.
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368 002474 104012      ERROR 12 ;REPORT ERROR=CLOCK BUFFER REG. FAILED TO RETURN
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428 002476 012637 000004 3S:  MOV  (SP)+,@#ERRVEC
1429

```

D03

MAINDEC-11-DVKWA-A  
DVKWA.P11 T2

MACY11 27(732) 04-OCT-76 14:57 PAGE 30  
\*TEST THE ADDRESSABILITY OF CLOCK BUFFER REG.

1280  
1281  
1282  
1283

# E03

MAINDEC-11-DVKWA-A  
DVKWA.P11 T2

MAY11 27(732) 04-OCT-76 14:57 PAGE 31  
\*TEST THE ADDRESSABILITY OF CLOCK BUFFER REG.

1294  
1295  
1296  
1297  
1298  
1299  
1300  
1301  
1302

```

:*****:
:TEST 3      *TEST THAT CLOCK A STATUS REGISTER BIT 14 CAN BE SET AND CLEARED
:*
:*CLOCK STATUS REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
:*F/FS OR GATES
:*
:*****:

```

1293 002502 000004  
1294 002504 012737 000100 001160  
1295  
1296 002512 005077 176660  
1297 002516 052777 040000 176652  
1298 002524 012737 040000 001124  
1299 002532 017737 176640 001126  
1300 002540 023737 001124 001126  
1301 002546 001402  
1302

```

:*****:
:ST3: SCOPE
:MOV      #100,$TIMES      ;;DO 100 ITERATIONS
:CLR      2ASR             ;/CLEAR THE STATUS REGISTER
:BIS      #BIT14,2ASR      ;/SET BIT 14.
:MOV      #BIT14,$GDDAT    ;/SET FOR ERROR TYPEOUT S/B.
:MOV      2ASR,$BDDAT      ;/READ THE STATUS REGISTER.
:CMR      $GDDAT,$BDDAT    ;/DID BIT 14 AND ONLY BIT 14 SET?
:BEQ      15               ;/IF SO-LETS TRY CLEARING IT.

```

;;\*\*\*\*\*>> ERROR <<\*\*\*\*\*

1305 002550 104002  
1306  
1307

```

ERROR 2 ;/ERROR CLOCK AS STATUS REGISTER
;/BIT 14 FAILED TO BIT SET.

```

;;\*\*\*\*\*>> ERROR <<\*\*\*\*\*

1310 002552 000412  
1311  
1312 002554 042777 040000 176614 15:  
1313 002562 005037 001124  
1314 002566 017737 176604 001126  
1315 002574 001401  
1316  
1317

```

BR 25 ;/BR TO END SUBTEST.
BIC #BIT14,2ASR ;/TRY CLEARING BIT 14.
CLR $GDDAT ;/CLEAR S/G FOR TYPEOUT IF ANY.
MOV 2ASR,$BDDAT ;/NOW READ IT BACK.
BEQ 25 ;/IF ZERO - NO ERROR!

```

;;\*\*\*\*\*>> ERROR <<\*\*\*\*\*

1320 002576 104002  
1321  
1322  
1323

```

ERROR 2 ;/ERROR - CLOCK A STATUS REGISTER.
;/BIT 14 FAILED TO CLEAR.

```

;;\*\*\*\*\*>> ERROR <<\*\*\*\*\*

1326 002600  
1327

25:

F03

MAINDEC-11-DVKWA-A  
DVKWA.P11 T3

MACY11 27(732) 04-OCT-76 14:57 PAGE 32  
\*TEST THAT CLOCK A STATUS REGISTER BIT 14 CAN BE SET AND CLEARED

1328



H03

MAINDEC-11-DVQWA-A  
DVQWA.P11 T4

MACY11 27(732) 04-OCT-76 14:57 PAGE 34  
\*TEST THAT CLOCK A STATUS REGISTER BIT 13 CAN BE SET AND CLEARED

1373

1374  
1375  
1376  
1377  
1378  
1379  
1380  
1381  
1382  
1383  
1384  
1385  
1386  
1367  
1388  
1389  
1390  
1391  
1392

.....  
\*TEST 5 \*TEST THAT CLOCK A STATUS REGISTER BIT 11 CAN BE SET AND CLEARED  
\*  
\*CLOCK STATUS REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL  
\*F/FS OR GATES  
\*  
.....

002676 000004  
002700 012737 000100 001160  
002706 005077 176464  
002712 052777 004000 176456  
002720 012737 004000 001124  
002726 017737 176444 001126  
002734 023737 001124 001126  
002742 001402

ST5: SCOPE  
MOV #100,STIMES ;DO 100 ITERATIONS  
CLR @ASR ;/CLEAR THE STATUS REGISTER.  
BIS #BIT11,@ASR ;/SET BIT 11.  
MOV #BIT11,\$GDDAT ;/SET FOR ERROR TYPEOUT S/B.  
MOV @ASR,\$SDDAT ;/READ THE STATUS REGISTER.  
CMP \$GDDAT,\$SDDAT ;/DID BIT 11 AND ONLY BIT 11 SET?  
BEQ IS ;/IF SO-LETS TRY CLEARING IT.

;;SSSSSSSSSSSSSSSSSSSSSSSSSS>>> ERROR <<<SSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

1395  
1396  
1397

ERROR 2 ;/ERROR CLOCK AS STATUS REGISTER  
;/BIT 11 FAILED TO BIT SET.

;;SSSSSSSSSSSSSSSSSSSSSSSSSS>>> ERROR <<<SSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

1400  
1401  
1402  
1403  
1404  
1405  
1406  
1407

002746 000412  
002750 042777 004000 176410  
002756 005037 001124  
002762 017737 176410 001126  
002770 001401

BR 2S ;/BR TO END SUBTEST.  
IS: BIC #BIT11,@ASR ;/TRY CLEARING BIT 11.  
CLR \$GDDAT ;/CLEAR S/B FOR TYPEOUT IF ANY.  
MOV @ASR,\$SDDAT ;/NOW READ IT BACK.  
BEQ 2S ;/IF ZERO - NO ERROR!

;;SSSSSSSSSSSSSSSSSSSSSSSSSS>>> ERROR <<<SSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

1410  
1411  
1412  
1413

002772 104002

ERROR 2 ;/ERROR - CLOCK A STATUS REGISTER.  
;/BIT 11 FAILED TO CLEAR.

;;SSSSSSSSSSSSSSSSSSSSSSSSSS>>> ERROR <<<SSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

1416  
1417

002774

2S:

J03

MAINDEC-11-DVKWA-A  
DVKWA.P11 TS

MACY11 27(732) 04-OCT-76 14:57 PAGE 36  
\*TEST THAT CLOCK A STATUS REGISTER BIT 11 CAN BE SET AND CLEARED

1418

K03

MAINDEC-11-DVKWA-A  
DVKWA.P11 TS

MACY11 27(732) 04-OCT-76 14:57 PAGE 37  
\*TEST THAT CLOCK A STATUS REGISTER BIT 11 CAN BE SET AND CLEARED

1419  
1420  
1421  
1422  
1423  
1424  
1425  
1426  
1427  
1428  
1429  
1430  
1431  
1432  
1433  
1434  
1435  
1436  
1437

```

:*****;/8*****
: *TEST 6 *TEST THAT CLOCK A STATUS REGISTER BIT 6 CAN BE SET AND CLEARED
: *
: *CLOCK STATUS REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
: *F/FS OR GATES
: *

```

002774 000004  
002776 012737 000100 001160  
003004 005077 176366  
003010 052777 000100 176360  
003016 012737 000100 001124  
003024 017737 176346 001126  
003032 023737 001124 001126  
003040 001402

```

:*****
: *ST6: SCOPE
: *NOV #100,STIMES ;;DO 100 ITERATIONS
: *CLR #ASR ;/CLEAR THE STATUS REGISTER.
: *BIS #BIT6,#ASR ;/SET BIT 6.
: *MOV #BIT6,$GDDAT ;/SET FOR ERROR TIMEOUT S/B.
: *MOV #ASR,$BDDAT ;/READ THE STATUS REGISTER.
: *CMP $GDDAT,$BDDAT ;/DID BIT 6 AND ONLY BIT 6 SET?
: *BEQ 15 ;/IF SO-LETS TRY CLEARING IT.

```

;;\*\*\*\*\*>> ERROR <<<\*\*\*\*\*

1440 003042 104002  
1441  
1442

```

: *ERROR 2 ;/ERROR CLOCK AS STATUS REGISTER
: * ;/BIT 6 FAILED TO BIT SET.

```

;;\*\*\*\*\*>> ERROR <<<\*\*\*\*\*

1445 003044 000412  
1446  
1447 003046 042777 000100 176322  
1448 003054 005037 001124  
1449 003060 017737 176312 001126  
1450 003066 001401  
1451  
1452

```

: *BR 25 ;/BR TO END SUBTEST.
: *15: BIC #BIT6,#ASR ;/TRY CLEARING BIT 6.
: *CLR $GDDAT ;/CLEAR S/B FOR TIMEOUT IF ANY.
: *MOV #ASR,$BDDAT ;/NOW READ IT BACK.
: *BEQ 25 ;/IF ZERO - NO ERROR!

```

;;\*\*\*\*\*>> ERROR <<<\*\*\*\*\*

1455 003070 104002  
1456  
1457  
1458

```

: *ERROR 2 ;/ERROR - CLOCK A STATUS REGISTER.
: * ;/BIT 6 FAILED TO CLEAR.

```

;;\*\*\*\*\*>> ERROR <<<\*\*\*\*\*

1461 003072  
1462

25:

L03

MAINDEC-11-DVKWA-A  
D-KWA.P11 T6

MACY11 27(732) 04-OCT-76 14:57 PAGE 38  
\*TEST THAT CLOCK A STATUS REGISTER BIT 6 CAN BE SET AND CLEARED

1463

M03

MAINDEC-11-DVKWA-A  
DVKWA.P11 T6

MACY11 27(732) 04-OCT-76 14:57 PAGE 39  
\*TEST THAT CLOCK A STATUS REGISTER BIT 6 CAN BE SET AND CLEARED

```

1464 ;/
1465 ;*****
1466 ;TEST 7 *TEST THAT CLOCK A STATUS REGISTER BIT 5 CAN BE SET AND CLEARED
1467 ;*
1468 ;*CLOCK STATUS REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
1469 ;*F/FS OR GATES
1470 ;*
1471 ;*****
1472 ;*****
1473 003072 000004 1ST7: SCOPE
1474 003074 012737 000100 001160 MOV #100, $TIMES ;;DO 100 ITERATIONS
1475
1476 003102 005077 176270 CLR @ASR ;/CLEAR THE STATUS REGISTER.
1477 003106 052777 000040 176262 BIS #BITS, @ASR ;/SET BIT 5.
1478 003114 012737 000040 001124 MOV #BITS, $GDDAT ;/SET FOR ERROR TYPEOUT S/B.
1479 003122 017737 176250 001126 MOV @ASR, $BDDAT ;/READ THE STATUS REGISTER.
1480 003130 023737 001124 001126 CMP $GDDAT, $BDDAT ;/DID BIT 5 AND ONLY BIT 5 SET?
1481 003136 001402 BEQ IS ;/IF SO-LETS TRY CLEARING IT.
1482
;;*****
1485 003140 104002 ERROR 2 ;/ERROR CLOCK AS STATUS REGISTER
1486 ;/BIT 5 FAILED TO BIT SET.
1487
;;*****
1490 003142 000412 BR 2S ;/BR TO END SUBTEST.
1491
1492 003144 042777 000040 176224 IS: BIC #BITS, @ASR ;/TRY CLEARING BIT 5.
1493 003152 005037 001124 CLR $GDDAT ;/CLEAR S/B FOR TYPEOUT IF ANY.
1494 003156 017737 176214 001126 MOV @ASR, $BDDAT ;/NOW READ IT BACK.
1495 003164 001401 BEQ 2S ;/IF ZERO - NO ERROR!
1496
1497
;;*****
1500 003166 104002 ERROR 2 ;/ERROR - CLOCK A STATUS REGISTER.
1501 ;/BIT 5 FAILED TO CLEAR.
1502
1503
;;*****
1506 003170 2S:
1507

```

N03

MAINDEC-11-DVKWA-A  
DVKWA.P11 T7

M3CY11 27(732) 04-OCT-76 14:57 PAGE 40  
\*TEST THAT CLOCK A STATUS REGISTER BIT 5 CAN BE SET AND CLEARED

1508

1..

:

```

1509 ;/8
1510 ;*****
1511 ;*TEST 10 *TEST THAT CLOCK A STATUS REGISTER BIT 4 CAN BE SET AND CLEARED
1512 ;*
1513 ;*CLOCK STATUS REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
1514 ;*F/FS OR GATES
1515 ;*
1516 ;*****
1517 ;*****
1518 003170 000004
1519 003172 012737 000100 001160 †ST10: SCOPE
1520 ;
1521 003200 005077 176172 CLR 2ASR ;/CLEAR THE STATUS REGISTER.
1522 003204 052777 003C20 176164 BIS #BIT4,2ASR ;/SET BIT 4.
1523 003212 012737 000020 001124 MOV #BIT4,$GDOAT ;/SET FOR ERROR TIMEOUT S/B.
1524 003220 017737 176152 001126 MOV 2ASR,$BDOAT ;/READ THE STATUS REGISTER.
1525 003226 023737 001124 001126 CMP $GDOAT,$BDOAT ;/DID BIT 4 AND ONLY BIT 4 SET?
1526 003234 001402 BEQ 15 ;/IF SO-LETS TRY CLEARING IT.
1527 ;
;; *****
ERROR <<< *****
ERROR 2 ;/ERROR CLOCK AS STATUS REGISTER
;/BIT 4 FAILED TO BIT SET.
;; *****
ERROR <<< *****
1535 003240 000412 BR 25 ;/BR TO END SUBTEST.
1536 ;
1537 003242 042777 000020 176126 15: BIC #BIT4,2ASR ;/TRY CLEARING BIT 4.
1538 003250 005037 001124 CLR $GDOAT ;/CLEAR S/B FOR TIMEOUT IF ANY.
1539 003254 017737 176116 001126 MOV 2ASR,$BDOAT ;/NOW READ IT BACK.
1540 003262 001401 BEQ 25 ;/IF ZERO - NO ERROR!
1541 ;
1542 ;
;; *****
ERROR <<< *****
1545 003264 104002 ERROR 2 ;/ERROR - CLOCK A STATUS REGISTER.
1546 ;/BIT 4 FAILED TO CLEAR.
1547 ;
1548 ;
;; *****
ERROR <<< *****
1551 003266 25:
1552 ;

```

C04

MAINDEC-11-DVKWA-A  
DVKWA.P11 T10

MACY11 27(732) 04-OCT-76 14:57 PAGE 42  
\*TEST THAT CLOCK A STATUS REGISTER BIT 4 CAN BE SET AND CLEARED

1553

```

1554 ;/8
1555 ;*****
1556 ;*TEST 11 *TEST THAT CLOCK A STATUS REGISTER BIT 3 CAN BE SET AND CLEARED
1557 ;*
1558 ;*CLOCK STATUS REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
1559 ;*F/FS OR GATES
1560 ;*
1561 ;*****
1562 ;*****
1563 003266 000004          TST11: SCOPE
1564 003270 012737 000100 001160      MOV      #100,$TIMES      ;;DO 100 ITERATIONS
1565
1566 003276 005077 176074          CLR      2ASR            ;/CLEAR THE STATUS REGISTER.
1567 003302 052777 000010 176066      BIS      #BIT3,2ASR     ;/SET BIT 3.
1568 003310 012737 000010 001124      MOV      #BIT3,$GDDAT   ;/SET FOR ERROR TIMEOUT S/B.
1569 003316 017737 176054 001126      MOV      2ASR,$BDDAT    ;/READ THE STATUS REGISTER.
1570 003324 023737 001124 001126      CMP      $GDDAT,$BDDAT  ;/DID BIT 3 AND ONLY BIT 3 SET?
1571 003332 001402          BEQ      15              ;/IF SO-LETS TRY CLEARING IT.
1572
;;*****
1575 003334 104002          ERROR 2                ;/ERROR CLOCK AS STATUS REGISTER
1576                                     ;/BIT 3 FAILED TO BIT SET.
1577
;;*****
1580 003336 000412          BR      25              ;/BR TO END SUBTEST.
1581
1582 003340 042777 000010 176030 15:   BIC      #BIT3,2ASR     ;/TRY CLEARING BIT 3.
1583 003346 005037 001124          CLR      $GDDAT        ;/CLEAR S/B FOR TIMEOUT IF ANY.
1584 003352 017737 176020 001126      MOV      2ASR,$BDDAT   ;/NOW READ IT BACK.
1585 003360 001401          BEQ      25              ;/IF ZERO - NO ERROR!
1586
1587
;;*****
1590 003362 104002          ERROR 2                ;/ERROR - CLOCK A STATUS REGISTER.
1591                                     ;/BIT 3 FAILED TO CLEAR.
1592
1593
;;*****
1596 003364          25:
1597

```

E04

MAINDEC-11-DVKWA-A  
DVKWA.P11 T11

MACY11 27(732) 04-OCT-76 14:57 PAGE 44  
\*TEST THAT CLOCK A STATUS REGISTER BIT 3 CAN BE SET AND CLEARED

1598



G04

MAINDEC-11-DMIA-A  
DMIA.P11 T12

MACY11 27(732) 04-OCT-76 14:57 PAGE 46  
\*TEST THAT CLOCK A STATUS REGISTER BIT 2 CAN BE SET AND CLEARED

1643

1644  
1645  
1646  
1647  
1648  
1649  
1650  
1651  
1652  
1653  
1654  
1655  
1656  
1657  
1658  
1659  
1660  
1661  
1662  
  
1665  
1666  
1667  
  
1670  
1671  
1672  
1673  
1674  
1675  
1676  
1677  
  
1680  
1681  
1682  
1683  
  
1686  
1687

003462 000004  
003464 012737 000100 001160  
  
003472 005077 175700  
003476 052777 000002 175672  
003504 012737 000002 001124  
003512 017737 175660 001126  
003520 023737 001124 001126  
003526 001402  
  
003530 104002  
  
003532 000412  
003534 042777 000002 175634  
003542 005037 001124  
003546 017737 175624 001126  
003554 001401  
  
003556 104002  
  
003560

```

;*****
;TEST 13      *TEST THAT CLOCK A STATUS REGISTER BIT 1 CAN BE SET AND CLEARED
;
;*CLOCK STATUS REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
;*F/FS OR GATES
;*
;*****
TST13:  SCOPE
        MOV     #100,$TIMES      ;;DO 100 ITERATIONS

        CLR     @ASR             ;/CLEAR THE STATUS REGISTER.
        BIS     @BIT1,@ASR      ;/SET BIT 1.
        MOV     @BIT1,$GDDAT    ;/SET FOR ERROR TYPEOUT S/B.
        MOV     @ASR,$BDDAT     ;/READ THE STATUS REGISTER.
        CMP     $GDDAT,$BDDAT   ;/DID BIT 1 AND ONLY BIT 1 SET?
        BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.

;*****
;*****
        ERROR 2                  ;/ERROR CLOCK AS STATUS REGISTER
;*****
;*****
;*****
        BR     2$                ;/BR TO END SUBTEST.

1$:     BIC     @BIT1,@ASR      ;/TRY CLEARING BIT 1.
        CLR     $GDDAT         ;/CLEAR S/B FOR TYPEOUT IF ANY.
        MOV     @ASR,$BDDAT    ;/NOW READ IT BACK.
        BEQ     2$              ;/IF ZERO - NO ERROR!

;*****
;*****
        ERROR 2                  ;/ERROR - CLOCK A STATUS REGISTER.
;*****
;*****
2$:

```

MAINDEC-11-DVKWA-A  
DVKWA.P11 T13

MACY11 27(732) 04-OCT-76 14:57 PAGE 48  
\*TEST THAT CLOCK A STATUS REGISTER BIT 1 CAN BE SET AND CLEARED

1688











805

MAINDEC-11-DVKWA-A  
DVKWA.P11 T22

MACY11 27(732) 04-OCT-76 14:57 PAGE 54  
\*TEST CLOCKS COUNTER REGISTER WITH 052525 PATTERN

```

1957
1958 004342 052777 000001 175026      BIS      #BIT0, @ASR      ;SINCE THIS IS MODE 0.
1959                                       ;SET B0 BIT(ALLOWS BUFFER-COUNT REG XFER).
1960 004350 012737 052525 001124      MOV      #052525, $GDOAT ;SET EXPECTED TO PATTERN IN CASE OF
1961                                       ;NEED OF ERROR TYPEOUT.
1962 004356 017746 175014                MOV      @ASR, -(6)    ;SAVE CSR
1963 004352 011637 001424                MOV      (6), $TMP3    ;GET CSR.
1964 004366 042737 177707 001424      BIC      #177707, $TMP3 ;SAVE RATE BITS.
1965 004374 052737 004005 001424      BIS      #BIT1!BIT2!BIT0, $TMP3 ;SET MODE 2, NO RATE.DISABLE INTERNAL OSC
1966 004402 013777 001424 174766      MOV      $TMP3, @ASR   ;LOAD CSR.
1967                                       ;THIS MUST BE DONE IN
1968                                       ;ORDER TO XFERR COUNTER
1969                                       ;TO BUFFER ON ST2.
1970 004410 052777 001000 174760      BIS      #BIT9, @ASR   ;GENERATE ON ST2 PULSE
1971 004416 017737 174756 001126      MOV      @ASR, $BDOAT ;READ THE PRESET BUFFER,
1972                                       ;PREVIOUS COUNTER
1973 004424 012677 174746                MOV      (6)+, @ASR    ;CONTENTS ARE IN $BDOAT.
1974 004430 005737 001126                TST      $BDOAT        ;RESTORE CSR
1975
1976 004434 023737 001124 001126      CMP      $GDOAT, $BDOAT ;DID ALL THE BITS AND NO OTHER BITS
1977                                       ;COME THROUGH?
1978 004442 001401                        BEQ      IS            ;BR IF YES TO NEXT TEST.
1979
1980

```

;; \$)) ERROR (<< \$

```

1983 004444 104005      ERROR 5      ;DATA ERROR CLOCK - PATTERN "052525"
1984                                       ;FAILED TO TRANSFER PROPERLY BETWEEN
1985                                       ;BUFFER AND COUNT REGISTERS.
1986
1987

```

;; \$)) ERROR (<< \$

1990 004446 18

```

1991
1992
1993
1994
1995
1996
1997
1998
1999

```

```

*****
;TEST 23      *TEST THAT INIT CLEARS STATUS REGISTER
;
;TESTING OF THE INIT LOGIC AS RECEIVED FROM THE QBUS AND BUFFERED
;TO STATUS REGISTER F/FS.
;
*****

```

```

2000 004446 000004      $T23: SCOPE
2001 004450 012737 000005 001160      MOV      #5, $TIMES    ;;DO 5 ITERATIONS
2002
2003 004456 005037 001124                CLR      $GDOAT        ;EXPECTED DATA IS ZERO.
2004 004462 012777 177777 174706      MOV      #177777, @ASR ;SET ALL BITS IN THE STATUS REG.
2005
2006 004470 000005                RESET                 ;SYSTEM INITIALIZE.
2007
2008 004472 017737 174700 001126      MOV      @ASR, $BDOAT ;READ THE STATUS REG., ALL BITS SHOULD
2009                                       ;HAVE BEEN CLEARED BY INIT.
2010 004500 001402                        BEQ      IS            ;BR IF YES TO NEXT TEST.
2011
2012

```



MAINDEC-11-DVKWA-A  
DVKWA.P11 T24

MAY11 27(732) 04-OCT-76 14:57 PAGE 56  
\*TEST THAT INIT CLEARS BUFFER REGISTER

2069 004576 001403  
2070 004600 052777 016000 174610

BEQ TST25  
BIS #BIT11!BIT12!BIT10,20R2 ;ENABLE THEM

2071  
2072  
2073  
2074  
2075 004606 000004  
2076  
2077 004610 012777 000001 174560  
2078 004616 052777 001000 174552  
2079  
2080 004624 005777 174546  
2081 004630 100402  
2082  
2083  
::\*\*\*\*\*  
;\*TEST 25 \*TEST THE SETTING OF MAINTENANCE ST2 IN CLOCK BIT 15 TO SET  
\*\*\*\*\*

2084  
2085  
2086 004632 104001  
2087  
2088  
2089  
2090  
2091 004634 000410  
2092 004636 005737 001440  
2093 004642 001405  
2094 004644 032777 000004 174542  
2095 004652 001001  
2096  
2097

TST25: SCOPE  
MOV #1,20SR ;SET THE GO BIT(ENABLES ST2 TO SET FLG).  
BIS #BIT9,20SR ;SET MAINTENANCE ST2.  
TST 20SR ;DID BIT15 (ST2 FLAG) SET?  
BNI 15 ;BR IF YES - NEXT TEST

::\*\*\*\*\*  
::\*\*\*\*\*  
::\*\*\*\*\*

ERROR 1 ;ERROR - MAINTENANCE ST2 (BIT9)  
;DID NOT SET BIT15 (ST2 FLAG).

::\*\*\*\*\*  
::\*\*\*\*\*  
::\*\*\*\*\*

BR TST26 ;;  
TST EXS ;TEST EXTERNAL SIGNALS?  
BEQ TST26 ;;  
BIT #BIT2,20R ;DID EXTERNAL ST2 GET SET?  
BNE TST26 ;;

::\*\*\*\*\*  
::\*\*\*\*\*  
::\*\*\*\*\*

ERROR 6 ;ST2 OUT NOT DETECTED  
;BY TESTOR

::\*\*\*\*\*  
::\*\*\*\*\*  
::\*\*\*\*\*

2105  
2106  
2107  
2108  
2109  
2110 004656 000004  
2111  
2112 004660 012777 020000 174510  
2113 004666 052777 001000 174502  
2114 004674 032777 000001 174474  
2115 004702 001001  
2116  
2117

\*\*\*\*\*  
\*TEST 26 \*TEST THAT BIT00 IN CLOCK STATUS REG. WILL SET WHEN BIT13 AND MAIN. ST2  
\*\*\*\*\*

TST26: SCOPE  
MOV #BIT13,20SR ;SET "ST2 ENB COUNTER" IN CLK STATUS REG.  
BIS #BIT9,20SR ;GENERATE A MAINTENANCE ST2.  
BIT #BIT00,20SR ;DID BIT00 (GO) SET?  
BNE 15 ;BR IF YES - NEXT TEST.

::\*\*\*\*\*  
::\*\*\*\*\*  
::\*\*\*\*\*

ERROR 1 ;ERROR - BIT00 OF CLOCK'S STATUS REGISTER  
;FAILED TO SET WHEN BIT13 WAS SET  
;AND A MAINTENANCE ST2 GENERATED.

2120 004704 104001  
2121  
2122  
2123  
2124



# F05

MAINDEC-11-DVKWA-A  
DVKWA.P11

T30

MACY11 27(732) 04-OCT-76 14:57 PAGE 58  
\*SEE IF CLOCK WILL COUNT UP FROM A ZERO BASE, RATE:ST1

```
2181 005042 012737 000000 001124      MOV    #0,$GDDAT       ;CLEAR EXPECTED.
2182 005050 012737 005212 001110      MOV    #2,$SLPERR
2183
2184 005056 052777 000061 174312 1$:    BIS    #BITS!BIT4!BIT0,$ASR ;START CLOCK, RATE:ST1.
2185
2186 005064 052777 000400 174304      BIS    #BIT8,$ASR      ;GENERATE A MAINTENANCE ST1
2187                               ;CLOCK SHOULD COUNT ONCE.
2188 005072 005737 001440      TST    EXS             ;TEST EXTERNAL SIGNALS?
2189 005076 001406      BEQ    10$            ;NO
2190 005100 032777 000002 174306      BIT    #BIT1,$DR       ;YES - DID ST1 GET SET?
2191 005106 001002      BNE    10$            ;YES
2192

;;*****>>> ERROR <<<*****

2195 005110 104006      ERROR 6                ;ST1 OUT NOT DETECTED
2196                               ;BY TESTOR
2197

;;*****>>> ERROR <<<*****

2200 005112 000447      BR     TST31           ;;
2201 005114
2202 005114 017746 174256 10$:    MOV    $ASR,-(6)      ;/SAVE CSR
2203 005120 011637 001424      MOV    (6),$TMP3      ;/GET CSR.
2204 005124 042737 177707 001424      BIC    #177707,$TMP3  ;/SAVE RATE BITS.
2205 005132 052737 004005 001424      BIS    #BIT11!BIT2!BIT0,$TMP3 ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
2206 005140 013777 001424 174230      MOV    $TMP3,$ASR    ;/LOAD CSR.
2207                               ;/THIS MUST BE DONE IN
2208                               ;/ORDER TO XFERR COUNTER
2209                               ;/TO BUFFER ON ST2.
2210 005146 052777 001000 174222      BIS    #BIT9,$ASR    ;/GENERATE ON ST2 PULSE
2211 005154 017737 174220 001126      MOV    $ABR,$BDDAT   ;/READ THE PRESET BUFFER,
2212                               ;/PREVIOUS COUNTER
2213 005162 012677 174210      MOV    (6)+,$ASR     ;/CONTENTS ARE IN $BDDAT.
2214 005166 005737 001126      TST    $BDDAT        ;/RESTORE CSR
2215 005172 005237 001124      INC    $GDDAT        ;COUNT=OLD COUNT+1
2216 005176 023737 001124 001126      CMP    $GDDAT,$BDDAT ;COUNT READ=COUNT EXP'ED?
2217 005204 001402      BEQ    2$            ;YES - SEE IF WE'RE THROUGH.
2218

;;*****>>> ERROR <<<*****

2221 005206 104011      ERROR 11               ;CLOCK FAILED TO COUNT UP PROPERLY.
2222

;;*****>>> ERROR <<<*****

2225 005210 000410      BR     3$              ;GOTO SCOPE LOOP.
2226
2227 005212 005077 174160 2$:    CLR    $ASR
2228 005216 013777 001124 174154      MOV    $GDDAT,$ABR
2229 005224 005737 001124      TST    $GDDAT        ;ALL DONE?
2230 005230 001312      BNE    1$            ;NO DO NEXT INCREMENT.
2231 005232 3$:
2232
2233 ;;*****
2234 ;*TEST 31 *TEST THAT OVERFLOW (CSR BIT07) WILL SET ON OVERFLOW
2235 ;;*****
2236 005232 000004      TST31: SCOPE
```





MAINDEC-11-DVKWA-A  
DVKWA.P11 T34

MACY11 27(732) 04-OCT-76 14.57 PAGE 61  
\*TEST THE ABILITY OF CLOCK TO COUNT AT 1MHZ RATE

```
2349 005502 017746 173670      MOV    @ASR,-(6)           ;/SAVE CSR
2350 005506 011637 001424      MOV    (6), $TMP3         ;/GET CSR.
2351 005512 042737 177707 001424  BIC    @177707, $TMP3      ;/SAVE RATE BITS.
2352 005520 052737 004005 001424  BIS    @BIT1!BIT2!BIT0, $TMP3 ;/SET MODE 2, NO RATE, DISABLE INTERNAL OSC
2353 005526 013777 001424 173642  MOV    $TMP3, @ASR        ;/LOAD CSR.
2354                                     ;/THIS MUST BE DONE IN
2355                                     ;/ORDER TO XFERR COUNTER
2356                                     ;/TO BUFFER ON ST2.
2357 005534 052777 001000 173634  SIS    @BIT9, @ASR         ;/GENERATE ON ST2 PULSE
2358 005542 017737 173632 001126  MOV    @ABR, $BDDAT       ;/READ THE PRESET BUFFER,
2359                                     ;/PREVIOUS COUNTER
2360 005550 012677 173622      MOV    (6)+, @ASR         ;/CONTENTS ARE IN $BDDAT.
2361 005554 005737 001126      TST    $BDDAT             ;/RESTORE CSR
2362 005560 001004      BNE    2$                 ;/YES - NEXT TEST.
2363 005562 105766 177776      TSTB   -2(6)              ;/AT HIGH RATE MAY HAVE HAD OVERFLOW
2364                                     ;/NOTE: CSR HAD BEEN PUT ON STACK.
2365 005566 100401      BMI    2$                 ;/NEXT TEST IF OVERFLOW.
```

:: \$ ERROR <<< \$

```
2370 005570 104006      ERROR 6                   ;/CLOCK FAILED TO COUNT AT
2371                                     ;/RATE: 1MHZ
2372
```

:: \$ ERROR <<< \$

```
2375 005572 005077 173600      2$: CLR    @ASR           ;/CLEAR THE CLOCK.
2376
2377
2378
2379
```

:: \*\*\*\*\*  
;\*TEST 35 \*TEST THE ABILITY OF CLOCK TO COUNT AT 100KHZ RATE

```
2380
2381
2382
2383
2384
2385
2386
2387
2388 005576 000004      ST35: SCOPE
2389 005600 012737 000005 001160  MOV    @5, $TIMES         ;/DO 5 ITERATIONS
2390
2391
2392 005606 005077 173564      CLR    @ASR               ;/CLEAR CLOCK
2393 005612 005077 173562      CLR    @ABR               ;/CLEAR PRESET BUFFER
2394 005616 012777 000021 173552  MOV    @BIT0!20, @PCR      ;/START CLOCK, MODE0, RATE: 100KHZ
2395 005624 005000      CLR    R0                  ;/NOW WE'LL DO A LITTLE DELAY. THIS DELAY
2396
2397 005626 005200      1$: INC    R0               ;/WILL AMOUNT TO APPROXIMATELY
2398 005630 001376      BNE    1$                 ;/369 MS.
```

```
2400 005632 017746 173540      MOV    @ASR,-(6)           ;/SAVE CSR
2401 005636 011637 001424      MOV    (6), $TMP3         ;/GET CSR.
2402 005642 042737 177707 001424  BIC    @177707, $TMP3      ;/SAVE RATE BITS.
2403 005650 052737 004005 001424  BIS    @BIT1!BIT2!BIT0, $TMP3 ;/SET MODE 2, NO RATE, DISABLE INTERNAL OSC
2404 005656 013777 001424 173512  MOV    $TMP3, @ASR        ;/LOAD CSR.
```









# N05

MAINDEC-11-DVKWA-A  
DVKWA.P11 T41

MACY11 27(732) 04-OCT-76 14:57 PAGE 66  
\*TEST THE ABILITY OF CLOCK TO COUNT AT LINEFREQ RATE

```

2629 006470 104006                ERROR  5                ;/CLOCK FAILED TO COUNT AT
2630                                     ;/RATE:LINEFREQ
2631
;;SSSSSSSSSSSSSSSSSSSSSSSSSSSS)) ERROR <<SSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

2634 006472 005077 17270L          2S:  CLR  QASR                ;/CLEAR THE CLOCK.
2635
2636
2637
2638
2639
2640
2641
2642
2643
2644
2645
2646
2647 006476 000004
2648 006500 012737 000005 001160  ST42: SCOPE
2649                                     MOV  #5,$TIMES                ;;DO 5 ITERATIONS
2650
2651 006506 005077 172664          CLR  QASR                ;/CLEAR CLOCK
2652 006512 005077 172662          CLR  QABR                ;/CLEAR PRESET BUFFER
2653 006516 012777 000061 172652  MOV  #BIT0!60,QASR        ;/START CLOCK, MODE0, RATE:ST1
2654 006524 052777 000400 172644  BIS  #BIT8,QASR
2655 006532 017746 172640          MOV  QASR,-(6)            ;/SAVE CSR
2656 006536 011637 001424          MOV  (6),$TMP3           ;/GET CSR.
2657 006542 042737 177707 001424  BIC  #177707,$TMP3       ;/SAVE RATE BITS.
2658 006550 052737 004005 001424  BIS  #BIT11!BIT2!BIT0,$TMP3 ;/SET MODE 2, NO RATE,DISAPLE INTERNAL OSC
2659 006556 013777 001424 172612  MOV  $TMP3,QASR          ;/LOAD CSR.
2660
2661                                     ;/THIS MUST BE DONE IN
2662                                     ;/ORDER TO XFERR COUNTER
2663 006564 052777 001000 172604  BIS  #BIT9,QASR          ;/GENERATE ON ST2 PULSE
2664 006572 017737 172602 001126  MOV  QABR,$BDDAT         ;/READ THE PRESET BUFFER,
2665                                     ;/PREVIOUS COUNTER
2666 006600 012677 172572          MOV  (6)+,QASR           ;/CONTENTS ARE IN $BDDAT.
2667 006604 005737 001126          TST  $BDDAT              ;/RESTORE CSR
2668 006610 001004          BNE  2S                  ;/YES - NEXT TEST.
2669 006612 105766 177776          TSTB -2(6)              ;/AT HIGH RATE MAY HAVE HAD OVERFLOW
2670                                     ;/NOTE: CSR HAD BEEN PUT ON STACK.
2671 006616 1004C1          BMI  2S                  ;/NEXT TEST IF OVERFLOW.
2672
2673
;;SSSSSSSSSSSSSSSSSSSSSSSSSSSS)) ERROR <<SSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

2676 006620 104006                ERROR  6                ;/CLOCK FAILED TO COUNT AT
2677                                     ;/RATE:ST1
2678
;;SSSSSSSSSSSSSSSSSSSSSSSSSSSS)) ERROR <<SSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

2681 006622 005077 172550          2S:  CLR  QASR                ;/CLEAR THE CLOCK.
2682
2683
2684

```









MAINDEC-11-DVKWA-A  
DVKWA.P11 T51

MACY11 27(732) 04-OCT-76 14:57 PAGE 71  
\*TEST THAT FOR BIT WILL CLEAR IF GO BIT IS SET

2909									
2910	007440	042777	000001	171730	BIC	#BITO,@ASR		;FOR BIT SETS HERE.	
2911								;CLEAR GO BIT.	
2912	007446	052777	000001	171722	BIS	#BITO,@ASR		;SET THE "GO" BIT AGAIN -	

G06

MAINDEC-11-DVKWA-A  
DVKWA.P11 T51

MACY11 27(732) 04-OCT-76 14:57 PAGE 72  
\*TEST THAT FOR BIT WILL CLEAR IF GO BIT IS SET

2913  
2914  
2915

007454 017737 171716 001126

MOV 2ASR, \$BDOAT

; SHOULD CLEAR FOR BIT.

; READ THE CSR.



MAINDEC-11-DVKWA-A  
DVKWA.P11 T52

MACY11 27(732) 04-OCT-76 14:57 PAGE 74  
\*TEST THAT WE CAN DISABLE THE INTERNAL OSC

2929				
2930				
2931				
2932	007506	000004		
2933	007510	012737	000005	001160
2934				
2935	007516	005077	171654	

```

:*****
: *TEST 52      *TEST THAT WE CAN DISABLE THE INTERNAL OSC
:*****
†ST52: SCOPE
      MOV      #5,$TIMES      ;;DO 5 ITERATIONS
      CLR      @ASR          ;CLEAR THE CSR

```

J06

MAINDEC-11-DVKWA-A  
DVKWA.P11 T52

MFCY11 27(732) 04-OCT-76 14:57 PAGE 75  
\*TEST THAT WE CAN DISABLE THE INTERNAL OSC

2936 007522 005077 171652

CLR 2ABR

;CLEAR THE PRESET BUFFER

K06

MAINDEC-11-DVKWA-A  
DVKWA.P11 T52

MACY11 27(732) 04-OCT-76 14:57 PAGE 76  
\*TEST THAT WE CAN DISABLE THE INTERNAL OSC

2937 007526 005037 001124  
2938

CLR \$GDDAT

;CLEAR EXPED.

```

2939 007532 012777 004000 171636      MOV      #BIT11,%ASR      ;DISABLE THE INTERNAL OSC.
2940 007540 052777 000011 171630      BIS      #BIT3!BIT0,%ASR ;START CLOCK:RATE 1MHZ.
2941 007546 005000                   CLR      RO
2942 007550 105200                   IS:     INCB      RO      ;DELAY A SHORT TIME.
2943 007552 001376                   BNE     IS
2944 007554 017746 171616      MOV      %ASR,-(6)      ;/SAVE CSR
2945 007560 011637 001424      MOV      (6),%TMP3      ;/GET CSR.
2946 007564 042737 177707 001424      BIC      #177707,%TMP3  ;/SAVE RATE BITS.
2947 007572 052737 004005 001424      BIS      #BIT11!BIT2!BIT0,%TMP3 ;/SET MODE 2, NO RATE.DISABLE INTERNAL OSC
2948 007600 013777 001424 171570      MOV      %TMP3,%ASR      ;/LOAD CSR.
2949                                     ;/THIS MUST BE DONE IN
2950                                     ;/ORDER TO XFERR COUNTER
2951                                     ;/TO BUFFER ON ST2.
2952 007606 052777 001000 171562      BIS      #BIT9,%ASR      ;/GENERATE ON ST2 PULSE
2953 007614 017737 171560 001126      MOV      %ASR,%BD0AT    ;/READ THE PRESET BUFFER,
2954                                     ;/PREVIOUS COUNTER
2955 007622 012677 171550      MOV      (6)+,%ASR      ;/CONTENTS ARE IN %BD0AT.
2956 007626 005737 001126      TST      %BD0AT         ;/RESTORE CSR
2957 007632 001401      BEQ     %25             ;NO - GOOD - NEXT TEST.
2958                                     ;;#####)) ERROR ((#####
2961 007634 104011                   ERROR    11             ;CLOCK DISABLE INTERNAL
2962                                     ;OSC. DID NOT WORK.
2963                                     ;;#####)) ERROR ((#####
2966 007636 005077 171534      %25:    CLR      %ASR      ;CLEAR THE CSR.
2967                                     ;*****
2968                                     ;*TEST 53 *TEST THAT CLOCK CAN BE COUNTED USING MAINTENANCE OSC
2969                                     ;*****
2970                                     ;*ST53: SCOPE
2971 007642 000004                   %25:    CLR      %ASR      ;CLEAR THE CSR.
2972                                     ;CLEAR THE PRESET BUFFER.
2973 007644 005077 171526      CLR      %ABR          ;CLEAR THE PRESET BUFFER.
2974 007650 005077 171524      CLR      %ABR          ;CLEAR THE PRESET BUFFER.
2975 007654 052777 004000 171514      BIS      #BIT11,%ASR    ;DISABLE THE INTERNAL OSC.
2976 007662 052777 000011 171506      BIS      #BIT3!BIT0,%ASR ;START CLOCK, 1MHZ, GO.
2977 007670 012700 177754      MOV      #-20,%RO      ;SET TO COUNT 20 TIMES
2978                                     ;*****
2979 007674 052777 002000 171474      IS:     BIS      #BIT10,%ASR ;GENERATE 1 MAINTENANCE OSC.
2980                                     ;NOTE: AT 1MHZ, IT TAKES 10
2981                                     ;MAINT. OSC TO EQUAL 1 COUNT
2982 007702 005200                   INC      RO             ;DO 20 MAINTENANCE OSC.
2983 007704 001373                   BNE     IS
2984                                     ;*****
2985 007706 017746 171464      MOV      %ASR,-(6)      ;/SAVE CSR
2986 007712 011637 001424      MOV      (6),%TMP3      ;/GET CSR.
2987 007716 042737 177707 001424      BIC      #177707,%TMP3  ;/SAVE RATE BITS.
2988 007724 052737 004005 001424      BIS      #BIT11!BIT2!BIT0,%TMP3 ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
2989 007732 013777 001424 171436      MOV      %TMP3,%ASR      ;/LOAD CSR.
2990                                     ;/THIS MUST BE DONE IN
2991                                     ;/ORDER TO XFERR COUNTER
2992                                     ;/TO BUFFER ON ST2.
2993 007740 052777 001000 171430      BIS      #BIT9,%ASR      ;/GENERATE ON ST2 PULSE
2994 007746 017737 171426 001126      MOV      %ABR,%BD0AT    ;/READ THE PRESET BUFFER,

```





```

3056 010166 005300          D&C      R0          ;/WHAT WE WANT TO CHECK
3057 010170 001373          BNE      48          ;/1MHZ PULSE ON 9 OSC PULSES.
3058
3059
3070 010172 017746 171200    MOV      2ASR,-(6)    ;/SAVE CSR
3071 010176 011637 001424    MOV      (6),%TMP3   ;/GET CSR.
3072 010202 042737 177707 001424  BIC     %177707,%TMP3 ;/SAVE RATE BITS.
3073 010210 052737 004005 001424  BIS     %BIT11,%BIT2,%BIT0,%TMP3 ;/SET MODE 2, NO RATE, DISABLE INTERNAL OSC
3074 010216 013777 001424 171152  MOV     %TMP3,2ASR    ;/LOAD CSR.
3075
3076
3077
3078 010224 052777 001000 171144  BIS     %BIT9,2ASR   ;/THIS MUST BE DONE IN
3079 010232 017737 171142 001126  MOV     2ASR,%SBOAT  ;/ORDER TO XFERR COUNTER
3080
3081 010240 012677 171132          MOV     (6)+,2ASR   ;/TO BUFFER ON ST2.
3082 010244 005737 001126          TST    %SBOAT       ;/GENERATE ON ST2 PULSE
3083 010250 022737 001124 001126  CMP    %SBOAT,%SBOAT ;/READ THE PRESET BUFFER,
3084 010256 001401          BEQ    55           ;/PREVIOUS COUNTER
3085
3086
3087
3088
3089
3090
3091
3092
3093
3094
3095 010262 005077 171110    SS:     CLR     2ASR    ;/CLEAR THE CSR.
3096
3097
3098
3099
3100
3101 010266 000004          ;/WE SEEM TO HAVE GENERATED
3102 010270 012737 000005 001160  MOV     #5,%TIMES   ;/ANOTHER 1MHZ PULSE ON
3103
3104 010276 005077 171074          ;/ONLY 9 MAINTENANCE
3105 010302 005077 171072          ;/OSC PULSES.
3106 010306 052777 004000 171062  SS:     CLR     2ASR    ;/CLEAR THE CSR.
3107 010314 052777 000021 171054  CLR     2ASR        ;/CLEAR THE PRESET BUFFER.
3108
3109
3110
3111 010322 012700 177634    10S:    MOV     #-100.,R0 ;/SET TO GENERATE 100 OSC PULSES.
3112
3113 010326 052777 002000 171042  1S:     BIS     %BIT10,2ASR ;/GENERATE ONE OSC PULSE.
3114 010334 005200          INC     R0           ;/DONE 100 OSC PULSES?
3115 010336 001373          BNE    1S           ;/NO - DO ANOTHER ONE.
3116
3117
3118 010340 012737 000001 001124  2S:     MOV     %1,%SBOAT ;/SET FOR ERROR TYPEOUT - IF ANY.
3119 010346 017746 171024          MOV     2ASR,-(6)   ;/SAVE CSR
3120 010352 011637 001424          MOV     (6),%TMP3   ;/GET CSR.
3121 010356 042737 177707 001424  BIC     %177707,%TMP3 ;/SAVE RATE BITS.

```

;; \*\*\*\*\* ERROR <<< \*\*\*\*\*

ERROR 11 ;/WE SEEM TO HAVE GENERATED  
;/ANOTHER 1MHZ PULSE ON  
;/ONLY 9 MAINTENANCE  
;/OSC PULSES.

;; \*\*\*\*\* ERROR <<< \*\*\*\*\*

\*\*\*\*\*  
\*TEST 55 \*TEST THE CLOCK'S 100KHZ DIVIDER  
\*\*\*\*\*

†ST55: SCOPE ;/DO 5 ITERATIONS  
MOV #5,%TIMES ;/DO 5 ITERATIONS  
CLR 2ASR ;/CLEAR THE CSR.  
CLR 2ASR ;/CLEAR THE PRESET BUFFER.  
BIS %BIT11,2ASR ;/DISABLE THE INTERNAL OSC.  
BIS %1:20,2ASR ;/ENABLE CLOCK, RATE:100KHZ  
MOV #-100.,R0 ;/SET TO GENERATE 100 OSC PULSES.  
BIS %BIT10,2ASR ;/GENERATE ONE OSC PULSE.  
INC R0 ;/DONE 100 OSC PULSES?  
BNE 1S ;/NO - DO ANOTHER ONE.  
MOV %1,%SBOAT ;/SET FOR ERROR TYPEOUT - IF ANY.  
MOV 2ASR,-(6) ;/SAVE CSR  
MOV (6),%TMP3 ;/GET CSR.  
BIC %177707,%TMP3 ;/SAVE RATE BITS.



















```

3626                                     ;/THIS MUST BE DONE IN
3627                                     ;/ORDER TO XFERR COUNTER
3628                                     ;/TO BUFFER ON ST2
3629 012504 052777 001000 166664          BIS      #BI*9, QASR      ;/GENERATE ON ST2 PULSE
3630 012512 017737 166662 001126          MOV      QABR, $BDDAT ;/READ THE PRESET BUFFER,
3631                                     ;/PREVIOUS COUNTER
3632 012520 012677 166652          MOV      (6)+, QASR   ;/CONTENTS ARE IN $BDDAT.
3633 012524 005737 001126          TST      $BDDAT      ;/RESTORE CSR
3634                                     ;/OF NON-ZERO - REPORT ERROR
3635 012530 001402          BEQ      2$
3636                                     ;/*****
;;*****>>> ERROR <<<*****
3639 012532 104002          ERROR      2          ;INCORRECT # OF ST1'S
3640                                     ;RECEIVED BY CLOCK
3641                                     ;/*****
;;*****>>> ERROR <<<*****
3644 012534 000404          BR      ENDP
3645
3646 012536          2$:          TST      QASR          ;DID ST2 FLG SET?
3647 012536 005777 166634          BMI      ENDP
3648 012542 100401
3649                                     ;/*****
;;*****>>> ERROR <<<*****
3652 012544 104006          ERROR      6          ;ST2 FLAG DID SET EVEN THOUGH
3653                                     ;SWITCH WAS TOGGLED.
3654                                     ;/*****
;;*****>>> ERROR <<<*****
3657
3658                                     ;
3659                                     ;NOW WE'LL DETERMINE IF WE ARE ALLOWED TO AUTO-SIZE.
3660                                     ;IF SO, WE'LL FIND OUT IF THERE ARE OTHER CLOCKS OUT THERE
3661                                     ;TO TEST.
3662                                     ;
3663
3664 012546 000004          ENDP:   SCOPE
3665
3666
3667 012550 105737 001215          TSTB     $ENVM          ;SEE IF APT WILL LET UP AUTO-SIZE.
3668 012554 100537          BMI      2$          ;NO - EXIT.
3669
3670
3671 012556 023737 001434 001436          CMP      MDEVCT, TSTCNT ;TESTED MAX. UNITS?
3672 012564 001507          BEQ      4$          ;YES EXIT.
3673 012566 006337 001426          ASL     ROTATE        ;POINT NEXT UNIT.
3674 012572 005237 001434          INC     MDEVCT
3675
3676 012576 062737 000004 001376          ADD     #4, ASR        ;YES, ADD T. BASE ADDR.
3677 012604 013746 000004          MOV     ERARVC, -(6)   ;SAVE CONTENTS OF LOC 4.
3678 012610 012737 012764 000004          MOV     #1$, ERARVC   ;SET UP IN CASE NO MORE CLOCKS.
3679
3680 012616 005777 166554          TST     QASR          ;TIME OUT HERE IF NO MORE CLOCKS.
3681

```

```

3682
3683 012622 005737 001202          TST      $PASS          ;IF HERE, ANOTHER CLOCK FOUND.
3684 012626 001003          BNE      3$            ;IS THIS 1ST PASS?
3685 012630 053737 001426 001430    BIS      ROTATE,UTEST  ;NO-GET OUT.
3686 012636          3$:
3687 012636 104400 012644          TYPE     65$          ;:TYPE ASCIZ STRING
3688 012642 000405          BR       64$          ;:GET OVER THE ASCIZ
3689
3690 012656          ;:65$: .ASCIZ <15><12>"UNIT #"  
3691 012656 013746 001204          MOV      $DEVCT,-(SP) ;:SAVE $DEVCT FOR TYPEOUT
3692 012662 104401          TYPOC   ;:GO TYPE--OCTAL ASCII(ALL DIGITS)
3693 012664 104400 012672          TYPE     67$          ;:TYPE ASCIZ STRING
3694 012670 000406          BR       66$          ;:GET OVER THE ASCIZ
3695
3696 012706          ;:67$: .ASCIZ " COMPLETED "  
3697 012706 005237 001204          INC      $DEVCT
3698 012712 104400 012720          TYPE     69$          ;:TYPE ASCIZ STRING
3699 012716 000410          BR       68$          ;:GET OVER THE ASCIZ
3700
3701 012740          ;:69$: .ASCIZ " TESTING UNIT #"  
3702 012740 013746 001204          MOV      $DEVCT,-(SP) ;:SAVE $DEVCT FOR TYPEOUT
3703 012744 104401          TYPOC   ;:GO TYPE--OCTAL ASCII(ALL DIGITS)
3704 012746 012637 000004          MOV      (6)+,ERRVEC ;:RESTORE LOC 4.
3705 012752 062737 000010 001402          ADD      #10,VECT1   ;:UPDATE VECTOR ADDR.
3706 012760 000137 002260          JMP      LOOP        ;:TEST NEW UNIT.
3707
3708 012764          1$:
3709 012764 062706 000004          ADD      #4,SP       ;:/ADD #4 TO STACK POINTER.
3710 012770 012637 000004          MOV      (6)+,ERRVEC ;:RESTORE LOC 4
3711 012774 022737 000001 001204          CMP      #1,$DEVCT  ;:TESTED ONLY ONE UNIT?
3712 013002 001424          BEQ     2$          ;:YES - NO NEED FOR TYPEOUT.
3713
3714 013004          4$:
3715 013004 104400 013012          TYPE     71$          ;:TYPE ASCIZ STRING
3716 013010 000405          BR       70$          ;:GET OVER THE ASCIZ
3717
3718 013024          ;:71$: .ASCIZ <15><12>"UNIT #"  
3719 013024 013746 001204          MOV      $DEVCT,-(SP) ;:SAVE $DEVCT FOR TYPEOUT
3720 013030 104401          TYPOC   ;:GO TYPE--OCTAL ASCII(ALL DIGITS)
3721 013032 104400 013040          TYPE     73$          ;:TYPE ASCIZ STRING
3722 013036 000406          BR       72$          ;:GET OVER THE ASCIZ
3723
3724 013054          ;:73$: .ASCIZ " COMPLETED "  
3725
3726 013054 013737 001250 001376 2$: MOV      $BASE,ASR
3727 013062 013737 001244 001402    MOV      $VECT1,VECT1
3728 013070 012737 000001 001204    MOV      #1,$DEVCT
3729
3730 013076 005037 001434          CLR      $DEVCT      ;:BEGIN TESTING 1ST UNIT.
3731 013102 012737 000001 001426    MOV      #1,ROTATE   ;:POINT TO IT.
3732
3733
3734 .SBTTL  END OF PASS ROUTINE
3735
3736 ;:*****  
3737 ;:INCREMENT THE PASS NUMBER ($PASS)

```

```

3738                                     ;*IF THERES A MONITOR GO TO IT
3739                                     ;*IF THERE ISN'T JUMP TO LOOP
3740
3741 013110                               SEOP:
3742 013110 000240                       NOP
3743 013112 005037 001102                 CLR      $STSTNM      ;; ZERO THE TEST NUMBER
3744 013116 005037 001160                 CLR      $TIMES      ;; ZERO THE NUMBER OF ITERATIONS
3745 013122 005237 001202                 INC      $PASS      ;; INCREMENT THE PASS NUMBER
3746 013126 042737 100000 001202        BIC      #100000,$PASS ;; DON'T ALLOW A NEG. NUMBER
3747 013134 005327                       DEC      (PC)+      ;; LOOP?
3748 013136 000001                       SEOPCT: .WORD      1
3749 013140 003064                       BGT      $DOAGN      ;; YES
3750 013142 012737                       MOV      (PC)+,2(PC)+ ;; RESTORE COUNTER
3751 013144 000001                       SENDCT: .WORD      1
3752 013146 013136                       SEOPCT
3753 013150 104400 013156                 TYPE     65$      ;; TYPE ASCIZ STRING
3754 013154 000406                       BR       64$      ;; GET OVER THE ASCIZ
3755                                     ;;65$: .ASCIZ <15><12>#ENDPASS #
3756 013172                               64$:
3757 013172 013746 001202                 MOV      $PASS,-(SP) ;; SAVE $PASS FOR TYPEOUT
3758 013176 104401                       TYPC    ;; GO TYPE--OCTAL ASCII(ALL DIGITS)
3759 013200 104400 013206                 TYPE     67$      ;; TYPE ASCIZ STRING
3760 013204 000411                       BR       66$      ;; GET OVER THE ASCIZ
3761                                     ;;67$: .ASCIZ # TOTAL ERRORS #
3762 013230                               66$:
3763 013230 013746 001432                 MOV      ERCNT,-(SP) ;; SAVE ERCNT FOR TYPEOUT
3764 013234 104401                       TYPC    ;; GO TYPE--OCTAL ASCII(ALL DIGITS)
3765 013236 104400 013244                 TYPE     69$      ;; TYPE ASCIZ STRING
3766 013242 000410                       BR       68$      ;; GET OVER THE ASCIZ
3767                                     ;;69$: .ASCIZ # GOOD UNITS #
3768 013264                               68$:
3769 013264 013746 001430                 MOV      UTEST,-(SP) ;; SAVE UTEST FOR TYPEOUT
3770 013270 104404                       TYPBN   ;; GO TYPE--BINARY ASCII
3771 013272 013700 000042                 $GET42: MOV      2#42,R0 ;; GET MONITOR ADDRESS
3772 013276 001405                       BEQ     $DOAGN      ;; BRANCH IF NO MONITOR
3773 013300 000005                       RESET   ;; CLEAR THE WORLD
3774 013302 004710                       SENDAD: JSR     PC,(R0) ;; GO TO MONITOR
3775 013304 000240                       NOP     ;; SAVE ROOM
3776 013306 000240                       NOP     ;; FOR
3777 013310 000240                       NOP     ;; ACT11
3778 013312
3779 013312 000137                       $DOAGN: JMP     2(PC)+      ;; RETURN
3780 013314 002260                       $RTNAD: .WORD    LOOP
3781 013316 377 377 000 $ENULL: .BYTE  -1,-1,0 ;; NULL CHARACTER STRING
3782 013322
3783
3784
3785
3786                                     .SBTTL      ;; I/O SIGNAL TEST #1 ST1 IN AND ST2 OUT IN AND OUT
3787
3788
3789
3790                                     ;; SWITCH PACK S2 MUST BE SET UP AS FOLLOWS:
3791
3792                                     SWITCH 1 - OFF
3793                                     SWITCH 2 - ON

```

3794  
3795  
3796  
3797  
3798  
3799  
3800  
3801  
3802  
3803  
3804  
3805  
3806  
3807  
3808  
3809  
3810  
3811  
3812  
3813  
3814  
3815  
3816  
3817  
3818  
3819  
3820  
3821  
3822  
3823  
3824  
3825  
3826  
3827  
3828  
3829  
3830  
3831  
3832  
3833  
3834  
3835  
3836  
3837  
3838  
3839  
3840  
3841  
3842  
3843  
3844  
3845  
3846  
3847  
3848  
3849

013322 104406  
013324 005077 166046  
013330 005077 166044  
013334 012777 000061 166034  
013342 052777 001000 166026  
013350 012777 000005 166020  
013356 052777 001000 166012  
013364 027727 166010 000001  
013372 007753  
013374 104000  
013376 000751

IOTST1: CKSWR  
IS: CLR 2ASR  
CLR 2ABR  
MOV #61,2ASR  
BIS #BIT9,2ASR  
MOV #5,2ASR  
BIS #BIT9,2ASR  
CMP 2ABR,#1  
BEQ IOTST1  
ERROR  
BR IOTST1

- 3 - OFF
- 4 - OFF
- 5 - ON
- 6 - ON
- 7 - NOT USED

: THIS SELECTS TTL THRESHOLDS AND POSITIVE SLOPE FOR  
: SCHMITT TRIGGER 1.

: PLEASE REMOVE ANY PREVIOUS JUMPER.

: JUMPER THE FOLLOWING PINS TOGETHER:

    J1 - SS (ST2 OUT)            TO J1 - VW (ST1-IN)

: LOAD AND START AT LOCATION 210  
: END PASSES OCCUR IMMEDIATELY AND ARE NOT REPORTED  
: ERRORS ARE REPORTED AS IN THE REGULAR LOGIC TEST AND  
: THEIR PRINTOUT MAY BE INHIBITED

: CHECK THE SWR  
: CLEAR THE CSR  
: CLEAR THE BUFFER REG.  
: RATE ST1, MODE 0, GO.  
: GENERATE A MAINTENANCE ST2.  
: NOW SET TO READ COUNT REG  
: FORCE COUNT -> BUFFER REG.  
: DID COUNT REG ADVANCE ONCE?  
: YES - LOOP.  
: ST2 OUT TO ST1 IN FAILED.

.SBTTL

: I/O SIGNAL TEST #2 CLOCK OVFLOW OUT TEST.

: SWITCH PACK S2 MUST BE SET UP AS FOLLOWS:

- SWITCH 1 - OFF
- 2 - OFF
- 3 - OFF
- 4 - ON
- 5 - ON
- 6 - OFF
- 7 - NOT USED

: THIS SELECTS TTL THRESHOLDS AND POSITIVE SLOPE FOR  
: SCHMITT TRIGGER 2.

: PLEASE REMOVE ANY PREVIOUS JUMPER.

: JUMPER THE FOLLOWING PINS TOGETHER:

    J1 - RR (CLK OV) TO J1 - TT (ST2-IN)

```

3850                                     ;LOAD AND START AT LOCATION 214.
3851                                     ;END PASSES OCCUR IMMEDIATELY AND ARE NOT REPORTED.
3852                                     ;ERRORS ARE REPORTED AS IN TH REGULAR LOGIC TEST AND
3853                                     ;THEIR PRINTOUT MAY BE INHIBITED.
3854
3855 013400 104405          IOTST2: CKSWR          ;CHECK THE SWR.
3856 013402 005077      165770          CLR          2ASR          ;CLEAR THE CSR.
3857 013406 012777      177777  165764  MOV          #-1,2ASR      ;PRELOAD PRESET BUFFER.
3858 013414 012777      000063  165754  MOV          #63,2ASR      ;RATE ST1, MODE 1, GO.
3859 013422 052777      000400  165746  BIS          #BIT8,2ASR    ;GENERATE A MAIN. ST1.
3860 013430 000240
3861 013432 000240
3862 013434 005777      165736          NOP
3863 013440 100757          TST          2ASR          ;DID OVERFLOW SET ST2 FLAG?
3864 013442 104000          BMI          IOTST2      ;YES - LOOP
3865 013444 000755          ERROR
3866                                     BR          IOTST2      ;CLK OV OUT TO ST2 IN FAILED.
3867                                     ;LOOP
3868
3869                                     .SBTTL
3870                                     ;I/O SIGNAL TEST #3 ST1 OUT AND ST2 IN
3871
3872                                     ;SWITCH PACK S2 MUST BE SET UP AS FOLLOWS:
3873                                     SWITCH 1 - OFF
3874                                     2 - OFF
3875                                     3 - OFF
3876                                     4 - ON
3877                                     5 - ON
3878                                     6 - ON
3879                                     7 - NOT USED
3880                                     ;THIS SELECTS TTL THRESHOLD AND POSITIVE SLOPE FOR
3881                                     ;SCHMITT TRIGGER 2.
3882
3883                                     ;PLEASE REMOVE ANY PREVIOUS JUMPERS.
3884
3885                                     ;JUMPER THE FOLLOWING PINS TOGETHER:
3886                                     J1 - UU (ST1 OUT)      TO J1 - TT (ST2-IN)
3887
3888                                     ;LOAD AND START AT LOCATION 220
3889                                     ;END PASSES OCCUR IMMEDIATELY AND ARE NOT REPORTED
3890                                     ;ERRORS ARE REPORTED AS IN THE REGULAR LOGIC TEST AND
3891                                     ;THEIR PRINTOUT MAY BE INHIBITED
3892
3893 013446 104406          IOTST3: CKSWR          ;CHECK THE SWR
3894 013450 012777      000001  165720  MOV          #1,2ASR      ;SET GO BIT.
3895 013456 052777      000400  165712  BIS          #BIT8,2ASR    ;GENERATE A MAIN. ST1.
3896 013464 005777      165706          TST          2ASR          ;DID ST2 FLAG SET?
3897 013470 100401          BMI          IS
3898 013472 104000          ERROR
3899                                     ;ST1 OUT TO ST2 IN FAILED
3900
3901 013474 032777      010000  165674  IS:          BIT          #BIT12,2ASR ;DID "FOR" BIT SET?
3902 013502 001761          BEQ          IOTST3      ;NO - GOOD!
3903 013504 104000          ERROR
3904 013506 000757          BR          IOTST3      ;"FOR" BIT SET ON ONLY 1 ST2.
3905                                     ;LOOP
3906                                     .SBTTL

```

```

3906
3907
3908
3909
3910
3911
3912
3913
3914
3915
3916
3917
3918
3919
3920
3921
3922
3923
3924
3925
3926
3927
3928
3929
3930
3931
3932
3933
3934 013510 017646 000000
3935 013514 116637 000001 013733
3936 013522 112637 013735
3937 013526 062716 000002
3938 013532 000406
3939 013534 112737 000001 013733
3940 013542 112737 000006 013735
3941 013550 112737 000005 013732
3942 013556 010346
3943 013560 010446
3944 013562 010546
3945 013564 113704 013735
3946 013570 005404
3947 013572 062704 000006
3948 013576 110437 013734
3949 013602 113704 013733
3950 013606 016605 000012
3951 013612 005003
3952 013614 006105 1S:
3953 013616 000404 2S:
3954 0 320 006105
3955 013622 006105
3956 013624 006105
3957 013626 010503
3958 013630 006103 3S:
3959 013632 105337 013734
3960 013636 100016
3961 013640 042703 177770

```

```

.SBTTL *SYSMAC ROUTINES
.SBTTL

.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.
*STYPOS---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
*STYPON---ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
*STYPOS OR STYPOC
*CALL:
*   MOV     NUM,-(SP)      ;; NUMBER TO BE TYPED
*   TYPON   ;; CALL FOR TYPEOUT
*STYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
*CALL:
*   MOV     NUM,-(SP)      ;; NUMBER TO BE TYPED
*   TYPOC   ;; CALL FOR TYPEOUT
*STYPOS: MOV     2(SP),-(SP) ;; PICKUP THE MODE
*         MOVB  1(SP),SOFILL ;; LOAD ZERO FILL SWITCH
*         MOVB  (SP)+,SOMODE+1 ;; NUMBER OF DIGITS TO TYPE
*         ADD   #2,(SP)      ;; ADJUST RETURN ADDRESS
*         BR    STYPON
*STYPOC: MOVB  #1,SOFILL    ;; SET THE ZERO FILL SWITCH
*         MOVB  #6,SOMODE+1 ;; SET FOR SIX(6) DIGITS
*STYPON: MOVB  #5,SOCNT     ;; SET THE ITERATION COUNT
*         MOV   R3,-(SP)    ;; SAVE R3
*         MOV   R4,-(SP)    ;; SAVE R4
*         MOV   R5,-(SP)    ;; SAVE R5
*         MOVB  SOMODE+1,R4 ;; GET THE NUMBER OF DIGITS TO TYPE
*         NEG   R4
*         ADD   #6,R4      ;; SUBTRACT IT FOR MAX. ALLOWED
*         MOVB  R4,SOMCDE   ;; SAVE IT FOR USE
*         MOVB  SOFILL,R4   ;; GET THE ZERO FILL SWITCH
*         MOV   12(SP),R5   ;; PICKUP THE INPUT NUMBER
*         CLR   R3         ;; CLEAR THE OUTPUT WORD
*         ROL  R5         ;; ROTATE MSB INTO "C"
*         BR   3S         ;; GO DO MSB
*         ROL  R5         ;; FORM THIS DIGIT
*         ROL  R5
*         ROL  R5
*         MOV  R5,R3
*         ROL  R3         ;; GET LSB OF THIS DIGIT
*         DECB SOMODE     ;; TYPE THIS DIGIT?
*         BPL  7S         ;; BR IF NO
*         BIC  #177770,R3 ;; GET RID OF JUNK

```

```

3962 013644 001002           BNE      4$          ;; TEST FOR 0
3963 013646 005704           TST     R4          ;; SUPPRESS THIS 0?
3964 013650 001403           BEQ     5$          ;; BR IF YES
3965 013652 005204           4$: INC     R4          ;; DON'T SUPPRESS ANYMORE 0'S
3966 013654 052703 000060       BIS     #'0,R3     ;; MAKE THIS DIGIT ASCII
3967 013660 052703 000040       5$: BIS     #' ,R3     ;; MAKE ASCII IF NOT ALREADY
3968 013664 110337 013730       MOVB   R3,B$       ;; SAVE FOR TYPING
3969 013670 104400 013730       TYPE   B$          ;; GO TYPE THIS DIGIT
3970 013674 105337 013732       7$: DECB  $OCNT     ;; COUNT BY 1
3971 013700 003347           BGT     2$          ;; BR IF MORE TO DO
3972 013702 002402           BLT     6$          ;; BR IF DONE
3973 013704 005204           INC     R4          ;; INSURE LAST DIGIT ISN'T A BLANK
3974 013706 003744           BR      2$          ;; GO DO THE LAST DIGIT
3975 013710 012605           6$: MOV     (SP)+,R5  ;; RESTORE R5
3976 013712 012604           MOV     (SP)+,R4     ;; RESTORE R4
3977 013714 012603           MOV     (SP)+,R3     ;; RESTORE R3
3978 013716 016666 000002 000004       MOV     2(SP),4(SP) ;; SET THE STACK FOR RETURNING
3979 013724 012616           MOV     (SP)+,(SP)
3980 013726 000002           RTI
3981 013730           8$: .BYTE 0          ;; RETURN
3982 013731           .BYTE 0          ;; STORAGE FOR ASCII DIGIT
3983 013732           .BYTE 0          ;; TERMINATOR FOR TYPE ROUTINE
3984 013733           .BYTE 0          ;; OCTAL DIGIT COUNTER
3985 013734 000000       .WORD 0          ;; ZERO FILL SWITCH
3986           .SBTTL BINARY TO ASCII AND TYPE ROUTINE
3987
3988           ;; *****
3989           ;; *THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 16-BIT
3990           ;; *BINARY-ASCII NUMBER AND TYPE IT.
3991           ;; *CALL:
3992           ;; *   MOV     NUMBER,-(SP)   ;; NUMBER TO BE TYPED
3993           ;; *   TYPBN          ;; TYPE IT
3994
3995 013736 010146           STYPBN: MOV     R1,-(SP)  ;; SAVE R1 ON THE STACK
3996 013740 016601 000006       MOV     6(SP),R1     ;; GET THE INPUT NUMBER
3997 013744 000261           SEC
3998 013746 112737 000060 014010       1$: MOVB  #'0,$BIN    ;; SET "C" SO CAN KEEP TRACK OF THE NUMBER OF BITS
3999 013754 006101           ROL     R1          ;; SET CHARACTER TO AN ASCII "0".
4000 013756 001406           BEQ     2$          ;; GET THIS BIT
4001 013760 105537 014010       ADCB   $BIN         ;; DONE?
4002 013764 104400 014010       TYPE   , $BIN       ;; NO--SET THE CHARACTER EQUAL TO THIS BIT
4003 013770 000241           CLC
4004 013772 000765           BR      1$          ;; GO TYPE THIS BIT
4005 013774 012601           2$: MOV     (SP)+,R1     ;; CLEAR "C" SO CAN KEEP TRACK OF BITS
4006 013776 016666 000002 000004       MOV     2(SP),4(SP)  ;; GO DO THE NEXT BIT
4007 014004 012616           MOV     (SP)+,(SP)  ;; POP THE STACK INTO R1
4008 014006 000002           RTI          ;; ADJUST THE STACK
4009 014010           8$: .BYTE 0,0       ;; RETURN TO USER
4010           ;; STORAGE FOR ASCII CHAR. AND TERMINATOR
4011
4012           .SBTTL ERROR HANDLER ROUTINE
4013
4014           ;; *****
4015           ;; *THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
4016           ;; *SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
4017           ;; *AND GO TO SERRTYP ON ERROR

```

```

4018
4019
4020
4021
4022
4023
4024
4025
4026 014012
4027 014012 104406
4028 014014 105237 001103
4029 014020 001775
4030 014022 013777 001102 165112
4031 014030 032777 002000 165102
4032 014036 001402
4033 014040 104400 001164
4034 014044 005237 001112
4035 014050 011637 001116
4036 014054 162737 000002 001116
4037 014062 117737 165030 001114
4038 014070 032777 020000 165042
4039 014076 001004
4040 014100 004737 014220
4041 014104 104400 001171
4042 014110
4043 014110 122737 000001 001214
4044 014116 001007
4045 014120 113737 001114 014132
4046 014126 004737 015712
4047 014132 000
4048 014133 000
4049 014134 000777
4050 014136 005777 164776
4051 014142 100002
4052 014144 000000
4053 014146 104406
4054 014150 032777 001000 164762
4055 014156 001402
4056 014160 013716 001110
4057 014164 005737 001162
4058 014170 001402
4059 014172 013716 001162
4060 014176
4061
4062 014176 005237 001432
4063 014202 001002
4064 014204 005337 001432
4065 014210
4066 014210 043737 001426 001430
4067 014216 000002
4068
4069
4070
4071
4072
4073

; *THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
; *SW15=1 HALT ON ERROR
; *SW13=1 INHIBIT ERROR TYPEOUTS
; *SW10=1 BELL ON ERROR
; *SW09=1 LOOP ON ERROR
; *CALL
; * ERROR N ;;ERROR=EMT AND N=ERROR ITEM NUMBER

SERROR:
7S: CKSWR ;: TEST FOR CHANGE IN SOFT-SWR
INCB $ERFLG ;: SET THE ERROR FLAG
BEQ 7S ;: DON'T LET THE FLAG GO TO ZERO
MOV $STNM, $DISPLAY ;: DISPLAY TEST NUMBER AND ERROR FLAG
BIT #BIT10, $SWR ;: BELL ON ERROR?
BEQ 1S ;: NO - SKIP
TYPE $BELL ;: RING BELL
1S: INC $ERTTL ;: COUNT THE NUMBER OF ERRORS
MOV (SP), $ERRPC ;: GET ADDRESS OF ERROR INSTRUCTION
SUB #2, $ERRPC
MOVB $ERRPC, $ITEMB ;: STRIP AND SAVE THE ERROR ITEM CODE
BIT #BIT13, $SWR ;: SKIP TYPEOUT IF SET
BNE 20S ;: SKIP TYPEOUTS
JSR PC, $ERRTYP ;: GO TO USER ERROR ROUTINE
TYPE , $CRLF

20S: CMPB #APTENV, $ENV ;: RUNNING IN APT MODE
BNE 2S ;: NO SKIP APT ERROR REPORT
MOVB $ITEMB, 21S ;: SET ITEM NUMBER AS ERROR NUMBER
JSR PC, $ATY4 ;: REPORT FATAL ERROR TO APT

21S: .BYTE 0
.BYTE 0

22S: BR 2S ;: APT ERROR LOOP
2S: TST $SWR ;: HALT ON ERROR
BPL 3S ;: SKIP IF CONTINUE
HALT ;: HALT ON ERROR!
CKSWR ;: TEST FOR CHANGE IN SOFT-SWR
3S: BIT #BIT09, $SWR ;: LOOP ON ERROR SWITCH SET?
BEQ 4S ;: BR IF NO
MOV $LPERR, (SP) ;: FUDGE RETURN FOR LOOPING
4S: TST $ESCAPE ;: CHECK FOR AN ESCAPE ADDRESS
BEQ 5S ;: BR IF NONE
MOV $ESCAPE, (SP) ;: FUDGE RETURN ADDRESS FOR ESCAPE

5S:
10S: INC ERCNT ;: /UPDATE ERROR COUNT.
BNE 10S ;: /BUT DON'T LET IT OVERFLOW.
DEC ERCNT ;: /KEEP AT 177777 IF OVERFLOW.

BIC ROTATE, UTEST ;: /REMOVE UNIT FROM LIST OF GOOD ONES.
RTI ;: /EXIT.

.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),
    
```

```

4074
4075
4076 014220
4077 014220 104400 001171
4078 014224 010046
4079 014226 005000
4080 014230 153700 001114
4081 014234 001004
4082
4083 014236 013746 001116
4084
4085 014242 104401
4086 014244 000426
4087 014246 005300
4088 014250 006300
4089 014252 006300
4090 014254 006300
4091 014256 062700 001256
4092 014252 012037 014272
4093 014256 001404
4094 014270 104400
4095 014272 000000
4096 014274 104400 001171
4097 014300 012037 014310
4098 014304 001404
4099 014306 104400
4100 014310 000000
4101 014312 104400 001171
4102 014316 011000
4103 014320 001004
4104 014322 012600
4105 014324 104400 001171
4106 014330 000207
4107 014332
4108 014332 013046
4109 014334 104401
4110 014336 005710
4111 014340 001770
4112 014342 104400 014350
4113 014346 000771
4114 014350 020040 000
4115 014354

```

;;\*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.

```

SERRTYP:
        TYPE      $CRLF          ;; "CARRIAGE RETURN" & "LINE FEED"
        MOV       R0,-(SP)       ;; SAVE R0
        CLR       R0             ;; PICKUP THE ITEM INDEX
        BISB     @($ITEMB,R0
        BNE      1$             ;; IF ITEM NUMBER IS ZERO, JUST
                                ;; TYPE THE PC OF THE ERROR
        MOV       $ERRPC,-(SP)   ;; SAVE $ERRPC FOR TYPEOUT
                                ;; ERROR ADDRESS
        TYPOC     GO TYPE--OCTAL ASCII(ALL DIGITS)
        BR       6$             ;; GET OUT
1$:     DEC       R0             ;; ADJUST THE INDEX SO THAT IT WILL
        ASL      R0             ;; WORK FOR THE ERROR TABLE
        ASL      R0
        ASL      R0
        ADD      @($ERRTB,R0    ;; FORM TABLE POINTER
        MOV      (R0)+,2$      ;; PICKUP "ERROR MESSAGE" POINTER
        BEQ     3$             ;; SKIP TYPEOUT IF NO POINTER
        TYPE     "ERROR MESSAGE" ;; TYPE THE "ERROR MESSAGE"
        WORD    0              ;; "ERROR MESSAGE" POINTER GOES HERE
        TYPE     $CRLF          ;; "CARRIAGE RETURN" & "LINE FEED"
        MOV     (R0)+,4$      ;; PICKUP "DATA HEADER" POINTER
        BEQ     5$             ;; SKIP TYPEOUT IF 0
        TYPE     "DATA HEADER" ;; TYPE THE "DATA HEADER"
        WORD    0              ;; "DATA HEADER" POINTER GOES HERE
        TYPE     $CRLF          ;; "CARRIAGE RETURN" & "LINE FEED"
        MOV     (R0),R0       ;; PICKUP "DATA TABLE" POINTER
        BNE     7$             ;; GO TYPE THE DATA
        MOV     (SP)+,R0      ;; RESTORE R0
        TYPE     $CRLF          ;; "CARRIAGE RETURN" & "LINE FEED"
        RTS     PC            ;; RETURN
        MOV     @ (R0)+,-(SP)  ;; SAVE @ (R0)+ FOR TYPEOUT
        TYPOC     GO TYPE--OCTAL ASCII(ALL DIGITS)
        TST     (R0)          ;; IS THERE ANOTHER NUMBER?
        BEQ     6$             ;; BR IF NO
        TYPE     "  "         ;; TYPE TWO(2) SPACES
        BR      7$             ;; LOC
        .ASCIZ  "/ /"        ;; TWO(2) SPACES
        .EVEN

```

.SBTTL SCOPE HANDLER ROUTINE

```

4116
4117
4118
4119
4120
4121
4122
4123
4124
4125
4126
4127
4128
4129

```

```

*****
; THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
; *AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REGISTER. (DISPLAY<7:0>)
; *AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
; *THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
; *SW14=1      LOOP ON TEST
; *SW11=1      INHIBIT ITERATIONS
; *SW09=1      LOOP ON ERROR
; *SW08=1      LOOP ON TEST IN SWR<7:0>
; *CALL
; * SCOPE          ;; SCOPE=IOT

```

```

4130 014354
4131 014354 104406
4132 014356 104406
4133 014360 032777 040000 164552
4134 014366 001114
4135
4136 014370 000416
4137
4138 014372 013746 000004
4139 014376 012737 014416 000004
4140 014404 005737 177060
4141 014410 012637 000004
4142 014414 000463
4143 014416 022626
4144 014420 012637 000004
4145 014424 000423
4146 014426
4147 014426 032777 000400 164504
4148 014434 001404
4149 014436 127737 164476 001102
4150 014444 001465
4151 014446 105737 001103
4152 014452 001421
4153 014454 123737 001115 001103
4154 014462 101015
4155 014464 032777 001000 164446
4156 014472 001404
4157 014474 013737 001110 001106
4158 014502 000446
4159 014504 105037 001103
4160 014510 005037 001160
4161 014514 000415
4162 014516 032777 004000 164414
4163 014524 001011
4164 014526 005737 001202
4165 014532 001406
4166 014534 005237 001104
4167 014540 023737 001160 001104
4168 014546 002024
4169 014550 012737 000001 001104
4170 014556 013737 014634 001160
4171 014564 105237 001102
4172 014570 113737 001102 001200
4173 014576 011637 001106
4174 014602 011637 001110
4175 014606 005037 001162
4176 014612 112737 000001 001115
4177 014620 013777 001102 164314
4178 014626 013716 001106
4179 014632 000002
4180 014634 003720
4181
4182
4183
4184
4185

```

```

$SCOPE:
      CKSWR
      CKSWR
1$:   BIT    #BIT14,2SWR
      BNE   $OVER
:****START OF CODE FOR THE XOR TESTER****
$XTSTR: BR    6$
      MOV   2#ERRVEC, -(SP)
      MOV   2$,$2#ERRVEC
      TST   2#177060
      MOV   (SP)+,2#ERRVEC
      BR    $SVLAD
5$:   CMP   (SP)+, (SP)+
      MOV   (SP)+,2#ERRVEC
      BR    7$
6$: ;****END OF CODE FOR THE XOR TESTER****
      BIT   #BIT08,2SWR
      BEQ   2$
      CMPB  2SWR,$STSTNM
      BEQ   $OVER
2$:   TSTB  $ERFLG
      BEQ   3$
      CMPB  $ERMAX,$ERFLG
      BHI   3$
      BIT   #BIT09,2SWR
      BEQ   4$
7$:   MOV   $LPERR,$LPADR
      BR    $OVER
4$:   CLRB  $ERFLG
      CLR  $TIMES
      BR    1$
3$:   BIT   #BIT11,2SWR
      BNE   1$
      TST  $PASS
      BEQ  1$
      INC  $ICNT
      CMP  $TIMES,$ICNT
      BGE  $OVER
1$:   MOV   #1,$ICNT
      MOV  $MXCNT,$TIMES
$SVLAD: INCB  $STSTNM
      MOVB $STSTNM,$STSTNM
      MOV  (SP),$LPADR
      MOV  (SP),$LPERR
      CLR  $ESCAPE
      MOVB #1,$ERMAX
$OVER: MOV   $STSTNM,$DISPLAY
      MOV  $LPADR,(SP)
      RTI
$MXCNT: 2000
.SBTTL  TTY INPUT ROUTINE
:*****
:ENABL  LSB

```

```

;;TEST FOR CHANGE IN SOFT-SWR
;;LOOP ON PRESENT TEST?
;;YES IF SW14=1
TESTER****
;;IF RUNNING ON THE "XOR" TESTER CHANGE
THIS INSTRUCTION TO A "NOP" (NOP=240)
SAVE THE CONTENTS OF THE ERROR VECTOR
SET FOR TIMEOUT
TIME OUT ON XOR?
RESTORE THE ERROR VECTOR
GO TO THE NEXT TEST
CLEAR THE STACK AFTER A TIME OUT
RESTORE THE ERROR VECTOR
LOOP ON THE PRESENT TEST
TESTER****
;;LOOP ON SPEC. TEST?
BR IF NO
ON THE RIGHT TEST? SWR<7:0>
BR IF YES
HAS AN ERROR OCCURRED?
BR IF NO
MAX. ERRORS FOR THIS TEST OCCURRED?
BR IF NO
LOOP ON ERROR?
BR IF NO
SET LOOP ADDRESS TO LAST SCOPE
ZERO THE ERROR FLAG
CLEAR THE NUMBER OF ITERATIONS TO MAKE
ESCAPE TO THE NEXT TEST
INHIBIT ITERATIONS?
BR IF YES
IF FIRST PASS OF PROGRAM
INHIBIT ITERATIONS
INCREMENT ITERATION COUNT
CHECK THE NUMBER OF ITERATIONS MADE
BR IF MORE ITERATION REQUIRED
REINITIALIZE THE ITERATION COUNTER
SET NUMBER OF ITERATIONS TO DO
COUNT TEST NUMBERS
SET TEST NUMBER IN APT MAILBOX
SAVE SCOPE LOOP ADDRESS
SAVE ERROR LOOP ADDRESS
CLEAR THE ESCAPE FROM ERPR ADDRESS
ONLY ALLOW ONE(1) ERROR ON NEXT TEST
DISPLAY TEST NUMBER
FUDGE RETURN ADDRESS
FIXES PS
MAX. NUMBER OF ITERATIONS

```

```

4186
4187
4188
4189
4190
4191 014636 022737 000176 001140 $CKSWR: CMP      #SWREG,SWR      ;; IS THE SOFT-SWR SELECTED?
4192 014644 001074          BNE      15$      ;; BRANCH IF NO
4193 014646 105777 164272          TSTB     2$TKS     ;; CHAR THERE?
4194 014652 100071          BPL      15$      ;; IF NO, DON'T WAIT AROUND
4195 014654 117746 164266          MOVB     2$TKB,-(SP) ;; SAVE THE CHAR
4196 014660 042716 177600          BIC      #1C177,(SP) ;; STRIP-OFF THE ASCII
4197 014664 022726 000007          CMP      #7,(SP)+  ;; IS IT A CONTROL G?
4198 014670 001062          BNE      15$      ;; NO, RETURN TO USER
4199 014672 123727 001134 000001          CMPB     $AUTOB,#1 ;; ARE WE RUNNING IN AUTO-MODE?
4200 014700 001456          BEQ      15$      ;; BRANCH IF YES
4201
4202 014702 104400 015363          TYPE     , $CNTLG  ;; ECHO THE CONTROL-G (↑G)
4203 014706 104400 015370          SGT$WR: TYPE     $MSWR  ;; TYPE CURRENT CONTENTS
4204 014712 013746 000176          MOV      SWREG,-(SP) ;; SAVE SWREG FOR TYPEOUT
4205 014716 104401          TYPOC   ;; GO TYPE--OCTAL ASCII(ALL DIGITS)
4206 014720 104400 015401          TYPE     , $MNEW  ;; PROMPT FOR NEW SWR
4207 014724 005046          19$: CLR      -(SP)  ;; CLEAR COUNTER
4208 014726 005046          CLR      -(SP)  ;; THE NEW SWR
4209 014730 105777 164210          7$: TSTB     2$TKS  ;; CHAR THERE?
4210 014734 100375          BPL      7$      ;; IF NOT TRY AGAIN
4211
4212 014736 117746 164204          MOVB     2$TKB,-(SP) ;; PICK UP CHAR
4213 014742 042716 177600          BIC      #1C177,(SP) ;; MAKE IT 7-BIT ASCII
4214
4215
4216
4217 014746 021627 000025          9$: CMP      (SP),#25 ;; IS IT A CONTROL-U?
4218 014752 001005          BNE      10$     ;; BRANCH IF NOT
4219 014754 104400 015356          TYPE     , $CNTLU ;; YES, ECHO CONTROL-U (↑U)
4220 014760 062706 000006          20$: ADD     #6,SP  ;; IGNORE PREVIOUS INPUT
4221 014764 000757          BR       19$     ;; LET'S TRY IT AGAIN
4222
4223
4224 014766 021627 000015          10$: CMP     (SP),#15 ;; IS IT A <CR>?
4225 014772 001022          BNE      16$     ;; BRANCH IF NO
4226 014774 005766 000004          TST      4(SP)   ;; YES, IS IT THE FIRST CHAR?
4227 015000 001403          BEQ      11$     ;; BRANCH IF YES
4228 015002 016677 000002 164130          MOV      2(SP),2$SWR ;; SAVE NEW SWR
4229 015010 062706 000006          11$: ADD     #6,SP  ;; CLEAR UP STACK
4230 015014 104400 001171          14$: TYPE     $CRLF  ;; ECHO <CR> AND <LF>
4231 015020 123727 001135 000001          CMPB     $INTAG,#1 ;; RE-ENABLE TTY KBD INTERRUPTS?
4232 015026 001003          BNE      15$     ;; BRANCH IF NOT
4233 015030 012777 000100 164106          MOV      #100,2$TKS ;; RE-ENABLE TTY KBD INTERRUPTS
4234 015036 000002          15$: RTI      ;; RETURN
4235 015040 004737 015624          16$: JSR     PC,$TYPEC ;; ECHO CHAR
4236 015044 021627 000060          CMP      (SP),#60 ;; CHAR < 0?
4237 015050 002420          BLT      18$     ;; BRANCH IF YES
4238 015052 021627 000067          CMP      (SP),#67 ;; CHAR > 7?
4239 015056 003015          BGT      18$     ;; BRANCH IF YES
4240 015060 042726 000060          BIC      #60,(SP)+ ;; STRIP-OFF ASCII
4241 015064 005766 000002          TST      2(SP)   ;; IS THIS THE FIRST CHAR

```

```

4242 015070 001403          BEQ      17$          ;; BRANCH IF YES
4243 015072 006316          ASL      (SP)        ;; NO, SHIFT PRESENT
4244 015074 006316          ASL      (SP)        ;; CHAR OVER TO MAKE
4245 015076 006316          ASL      (SP)        ;; ROOM FOR NEW ONE.
4246 015100 005266 000002 17$: INC      2(SP)      ;; KEEP COUNT OF CHAR
4247 015104 056616 177776  BIS      -2(SP), (SP) ;; SET IN NEW CHAR
4248 015110 000707          BR       7$          ;; GET THE NEXT ONE
4249 015112 104400 001170 18$: TYPE   $QUES     ;; TYPE ?<CR><LF>
4250 015116 000720          BR       20$        ;; SIMULATE CONTROL-U
4251
4252
4253
4254
4255 *****
4256 *THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
4257 *CALL:
4258 *      ROCHR          ;; INPUT A SINGLE CHARACTER FROM THE TTY
4259 *      RETURN HERE    ;; CHARACTER IS ON THE STACK
4260 *
4261
4262 SRCCHR: MOV      (SP), -(SP) ;; PUSH DOWN THE PC
4263        MOV      4(SP), 2(SP) ;; SAVE THE PS
4264 1$:   TSTB     2$TKS        ;; WAIT FOR
4265        BPL     1$          ;; A CHARACTER
4266        MOVB    2$TKB, 4(SP) ;; READ THE TTY
4267        BIC     #'C(177), 4(SP) ;; GET RID OF JUNK IF ANY
4268        CMP     4(SP), #23   ;; IS IT A CONTROL-S?
4269        BNE     3$          ;; BRANCH IF NO
4270 2$:   TSTB     2$TKS        ;; WAIT FOR A CHARACTER
4271        BPL     2$          ;; LOOP UNTIL ITS THERE
4272        MOVB    2$TKB, -(SP) ;; GET CHARACTER
4273        BIC     #'C(177), (SP) ;; MAKE IT 7-BIT ASCII
4274        CMP     (SP)+, #21   ;; IS IT A CONTROL-Q?
4275        BNE     2$          ;; IF NOT DISCARD IT
4276        BR      1$          ;; YES, RESUME
4277 3$:   CMP     4(SP), #140    ;; IS IT UPPER CASE?
4278        BLT     4$          ;; BRANCH IF YES
4279        CMP     4(SP), #175  ;; IS IT A SPECIAL CHAR?
4280        BGT     4$          ;; BRANCH IF YES
4281        BIC     #40, 4(SP)   ;; MAKE IT UPPER CASE
4282 4$:   RTI
4283 *****
4284 *THIS ROUTINE WILL INPUT A STRING FROM THE TTY
4285 *CALL:
4286 *      RDLIN          ;; INPUT A STRING FROM THE TTY
4287 *      RETURN HERE    ;; ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
4288 *
4289 *      TERMINATOR WILL BE A BYTE OF ALL 0'S
4290 SRCRLIN: MOV     R3, -(SP)   ;; SAVE R3
4291 1$:   MOV     #'TTYIN, R3    ;; GET ADDRESS
4292 2$:   CMP     #'TTYIN+8., R3 ;; BUFFER FULL?
4293        BLOS    4$          ;; BR IF YES
4294        ROCHR   ;; GO READ ONE CHARACTER FROM THE TTY
4295        MOVB    (SP)+, (R3)  ;; GET CHARACTER
4296 10$:  CMPB    #'177, (R3)    ;; IS IT A RUBOUT
4297        BNE     3$          ;; SKIP IF NOT

```

```

4298 015266 104400 001170 45: TYPE $QUES ;; TYPE A '?'
4299 015272 000763 BR 15 ;; CLEAR THE BUFFER AND LOOP
4300 015274 111337 015344 35: MOV (R3),95 ;; ECHO THE CHARACTER
4301 015300 104400 015344 TYPE 95
4302 015304 122723 000015 CMPB 15,(R3)+ ;; CHECK FOR RETURN
4303 015310 001356 BNE 25 ;; LOOP IF NOT RETURN
4304 015312 105063 177777 CLRB -1(R3) ;; CLEAR RETURN (THE 15)
4305 015316 104400 001172 TYPE $LF ;; TYPE A LINE FEED
4306 015322 012603 MOV (SP)+,R3 ;; RESTORE R3
4307 015324 011646 MOV (SP)-,(SP) ;; ADJUST THE STACK AND PUT ADDRESS OF THE
4308 015326 016666 000004 000002 MOV 4(SP),2(SP) ;; FIRST ASCII CHARACTER ON IT
4309 015334 012766 015346 000004 MOV $TTYIN,4(SP)
4310 015342 000002 RTI ;; RETURN
4311 015344 000 95: .BYTE 0 ;; STORAGE FOR ASCII CHAR. TO TYPE
4312 015345 000 .BYTE 0 ;; TERMINATOR
4313 015346 000010 $TTYIN: .BLKB 8. ;; RESERVE 8 BYTES FOR TTY INPUT
4314 015356 052536 005015 000 $CNTLU: .ASCIZ /↑U/<15><12> ;; CONTROL "U"
4315 015363 136 006507 000012 $CNTLG: .ASCIZ /↑G/<15><12> ;; CONTROL "G"
4316 015370 005015 053523 020122 $MSWR: .ASCIZ <15><12>/SWR = /
4317 015376 020075 000
4318 015401 040 047040 053505 $MNEW: .ASCIZ / NEW = /
4319 015406 036440 000040
4320 .SBTTL TYPE ROUTINE
4321
4322 *****
4323 *ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
4324 *THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
4325 *NOTE1: $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
4326 *NOTE2: $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
4327 *NOTE3: $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
4328 *
4329 *CALL:
4330 *1) USING A TRAP INSTRUCTION
4331 * TYPE ,MESADR ;; MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
4332 *OR
4333 * TYPE
4334 * MESADR
4335 *
4336
4337 015412 105737 001157 $TYPE: TSTB $TFPLG ;; IS THERE A TERMINAL?
4338 015416 100002 BPL 15 ;; BR IF YES
4339 015420 000000 HALT ;; HALT HERE IF NO TERMINAL
4340 015422 000430 BR 35 ;; LEAVE
4341 015424 010046 15: MOV RO,-(SP) ;; SAVE RO
4342 015426 017600 000002 MOV 22(SP),RO ;; GET ADDRESS OF ASCIZ STRING
4343 015432 122737 000001 001214 CMPB $APTENV,$ENV ;; RUNNING IN APT MODE
4344 015440 001011 BNE 625 ;; NO GO CHECK FOR APT CONSOLE
4345 015442 132737 000100 001215 BITB $APTSPOOL,$ENVM ;; SPOOL MESSAGE TO APT
4346 015450 001405 BEQ 625 ;; NO GO CHECK FOR CONSOLE
4347 015452 010037 015462 MOV RO,615 ;; SETUP MESSAGE ADDRESS FOR APT
4348 015456 004737 015702 JSR PC,$ATY3 ;; SPOOL MESSAGE TO APT
4349 015462 000000 615: .WORD 0 ;; MESSAGE ADDRESS
4350 015464 132737 000040 001215 625: BITB $APTCSUP,$ENVM ;; APT CONSOLE SUPPRESSED
4351 015472 001013 BNE 605 ;; YES, SKIP TYPE OUT
4352 015474 112046 25: MOV (RO)+,-(SP) ;; PUSH CHARACTER TO BE TYPED ONTO STACK
4353 015476 001005 BNE 45 ;; BR IF IT ISN'T THE TERMINATOR

```

DVKWA.P11

TYPE ROUTINE

```

4354 015500 005726          TST      (SP)+          ;; IF TERMINATOR POP IT OFF THE STACK
4355 015502 012600          60S:    MOV      (SP)+,R0      ;; RESTORE R0
4356 015504 062716 000002    3S:     ADD      #2,(SP)      ;; ADJUST RETURN PC
4357 015510 000002          RTI                      ;; RETURN
4358 015512 122716 000011    4S:     CMPB     #HT,(SF)      ;; BRANCH IF <HT>
4359 015516 001430          BEQ      B$              ;;
4360 015520 122716 000200          CMPB     #CRLF,(SP)      ;; BRANCH IF NOT <CRLF>
4361 015524 001006          BNE      B$              ;;
4362 015526 005726          TST      (SP)+          ;; POP <CR><LF> EQUIV
4363 015530 104400          TYPE                     ;; TYPE A CR AND LF
4364 015532 001171          SCRLF                    ;;
4365 015534 105037 015670          CLRB     $CHARCNT      ;; CLEAR CHARACTER COUNT
4366 015540 000755          BR       2$              ;; GET NEXT CHARACTER
4367 015542 004737 015624    5S:     JSR      PC,$TYPEC     ;; GO TYPE THIS CHARACTER
4368 015546 123726 001156    6S:     CMPB     $FILLC,(SP)+  ;; IS IT TIME FOR FILLER CHARS.?
4369 015552 001350          BNE      2$              ;; IF NO GO GET NEXT CHAR.
4370 015554 013746 001154          MOV      $NULL,-(SP)    ;; GET # OF FILLER CHARS. NEEDED
4371                                ;; AND THE NULL CHAR.
4372 015560 105366 000001    7S:     DECB     1(SP)        ;; DOES A NULL NEED TO BE TYPED?
4373 015564 002770          BLT      B$              ;; BR IF NO--GO POP THE NULL OFF OF STACK
4374 015566 004737 015624          JSR      PC,$TYPEC     ;; GO TYPE A NULL
4375 015572 105337 015670          DECB     $CHARCNT      ;; DO NOT COUNT AS A COUNT
4376 015576 000770          BR       7$              ;; LOOP

```

;HORIZONTAL TAB PROCESSOR

```

4377
4378
4379
4390 015600 112716 000040    8S:     MOVVB   #' (SP)        ;; REPLACE TAB WITH SPACE
4391 015604 004737 015624    9S:     JSR      PC,$TYPEC     ;; TYPE A SPACE
4392 015610 132737 000007 015670          BITB     #7,$CHARCNT    ;; BRANCH IF NOT AT
4393 015616 001372          BNE      9$              ;; TAB STOP
4394 015620 005726          TST      (SP)+          ;; POP SPACE OFF STACK
4395 015622 000724          BR       2$              ;; GET NEXT CHARACTER
4396 015624 105777 163320    STYPEC: TSTB     2$TPS     ;; WAIT UNTIL PRINTER IS READY
4397 015630 100375          BPL      STYPEC         ;;
4398 015632 116677 000002 163312          MOVVB   2(SP),2$TPB     ;; LOAD CHAR TO BE TYPED INTO DATA REG.
4399 015640 122766 000015 000002          CMPB     #CR,2(SP)      ;; IS CHARACTER A CARRIAGE RETURN?
4400 015646 001003          BNE      1$              ;; BRANCH IF NO
4401 015650 105037 015670          CLRB     $CHARCNT      ;; YES--CLEAR CHARACTER COUNT
4402 015654 000406          BR       STYPEX         ;; EXIT
4403 015656 122766 000012 000002    1S:     CMPB     #LF,2(SP)    ;; IS CHARACTER A LINE FEED?
4404 015664 001402          BEQ      STYPEX         ;; BRANCH IF YES
4405 015666 105227          INCB     (PC)+          ;; COUNT THE CHARACTER
4406 015670 000000          $CHARCNT: .WORD 0      ;; CHARACTER COUNT STORAGE
4407 015672 000207          STYPEX: RTS            PC

```

.SBTTL APT COMMUNICATIONS ROUTINE

```

4408
4409
4401 *****
4402 015674 112737 000001 016140 $ATY1: MOVVB   #1,$FFLG      ;; TO REPORT FATAL ERROR
4403 015702 112737 000001 016136 $ATY3: MOVVB   #1,$MFLG      ;; TO TYPE A MESSAGE
4404 015710 000403          BR       $ATYC         ;;
4405 015712 112737 000001 016140 $ATY4: MOVVB   #1,$FFLG      ;; TO ONLY REPORT FATAL ERROR
4406 015720          $ATYC:
4407 015720 010046          MOV      R0,-(SP)      ;; PUSH R0 ON STACK
4408 015722 010146          MOV      R1,-(SP)      ;; PUSH R1 ON STACK
4409 015724 105737 016136          TSTB     $MFLG         ;; SHOULD TYPE A MESSAGE?

```

```

4410 015730 001450      BEQ      55      ;; IF NOT: BR
4411 015732 122737 000001 001214    CMPB    #APTENV,SENV ;; OPERATING UNDER APT?
4412 015740 001031      BNE     35      ;; IF NOT: BR
4413 015742 132737 000100 001215    BITB    #APTSPool,SENVM ;; SHOULD SPOOL MESSAGES?
4414 015750 001425      BEQ     35      ;; IF NOT: BR
4415 015752 017600 000004      MOV     24(SP),RO ;; GET MESSAGE ADDR.
4416 015756 062766 000002 000004      ADD     #2,4(SP) ;; BUMP RETURN ADDR.
4417 015764 005737 001174      TST    SMSGTYPE ;; SEE IF DONE W/ LAST XMISSION?
4418 015770 001375      BNE     15      ;; IF NOT: WAIT
4419 015772 012037 001210      MOV     RO,SMSGAD ;; PUT ADDR IN MAILBOX
4420 015776 105720      TSTB   (RO)+    ;; FIND END OF MESSAGE
4421 016000 001376      BNE     25      ;;
4422 016002 163700 001210      SUB     SMSGAD,RO ;; SUB START OF MESSAGE
4423 016006 006200      ASR     RO      ;; GET MESSAGE LNGTH IN WORDS
4424 016010 010037 001212      MOV     RO,SMSGLGT ;; PUT LENGTH IN MAILBOX
4425 016014 012737 000004 001174      MOV     #4,SMSGTYPE ;; TELL APT TO TAKE MSG.
4426 016022 000413      BR      55      ;;
4427 016024 017637 000004 016050 35:      MOV     24(SP),45 ;; PUT MSG ADDR IN JSR LINKAGE
4428 016032 062766 000002 000004      ADD     #2,4(SP) ;; BUMP RETURN ADDRESS
4429 016040 013746 177776      MOV     177776-(SP) ;; PUSH 177776 ON STACK
4430 016044 004737 015412      JSR    PC,STYPE ;; CALL TYPE MACRO
4431 016050 000000      JSR    .WORD    0
4432 016052      JSR    45:
4433 016052 105737 016140      JSR    55:
4434 016056 001416      TSTB   $FFLG    ;; SHOULD REPORT FATAL ERROR?
4435 016060 005737 001214      BEQ     125     ;; IF NOT: BR
4436 016064 001413      TST    SENV     ;; RUNNING UNDER APT?
4437 016066 005737 001174      BEQ     125     ;; IF NOT: BR
4438 016072 001375      TST    SMSGTYPE ;; FINISHED LAST MESSAGE?
4439 016074 017637 000004 001176      BNE     115     ;; IF NOT: WAIT
4440 016102 062766 000002 000004      MOV     24(SP),SFATAL ;; GET ERROR #
4441 016110 005237 001174      ADD     #2,4(SP) ;; BUMP RETURN ADDR.
4442 016114 105037 016140      INC     SMSGTYPE ;; TELL APT TO TAKE ERROR
4443 016120 105037 016137      CLRB   $FFLG    ;; CLEAR FATAL FLAG
4444 016124 105037 016136      CLRB   $LFLG    ;; CLEAR LOG FLAG
4445 016130 012601      CLRB   $MFLG    ;; CLEAR MESSAGE FLAG
4446 016132 012600      MOV     (SP)+,R1 ;; POP STACK INTO R1
4447 016134 000207      MOV     (SP)+,RO ;; POP STACK INTO RO
4448 016136 000      RTS     PC      ;; RETURN
4449 016137 000      SMFLG: .BYTE 0 ;; MESSG. FLAG
4450 016140 000      SLFLG: .BYTE 0 ;; LOG FLAG
4451      SFFLG: .BYTE 0 ;; FATAL FLAG
4451      .EVEN
4452      APTSIZE=200
4453      APTENV=001
4454      APTSPool=100
4455      APTCSUP=040
4456      .SBTTL POWER DOWN AND JP ROUTINES
4457
4458      ;*****
4459      ;POWER DOWN ROUTINE
4460 016142 012737 016302 000024 $PWDRN: MOV #SILLUP,#PWVVEC ;; SET FOR FAST UP
4461 016150 012737 000340 000026      MOV     #340,#PWVVEC+2 ;; PRIO:7
4462 016156 010046      MOV     RO,-(SP) ;; PUSH RO ON STACK
4463 016160 010146      MOV     R1,-(SP) ;; PUSH R1 ON STACK
4464 016162 010246      MOV     R2,-(SP) ;; PUSH R2 ON STACK
4465 016164 010346      MOV     R3,-(SP) ;; PUSH R3 ON STACK

```

```

4466 016166 010446      MOV      R4,-(SP)      ;; PUSH R4 ON STACK
4467 016170 010546      MOV      R5,-(SP)      ;; PUSH R5 ON STACK
4468 016172 017746 162742  MOV      @SWR,-(SP)    ;; PUSH @SWR ON STACK
4469 016176 010637 016306  MOV      SP,$$SAVR6    ;; SAVE SP
4470 016202 012737 016214 000024  MOV      @SPWRUP,@@PWVVEC ;; SET UP VECTOR
4471 016210 000000      HALT
4472 016212 000776      BR      -2            ;; HANG UP
4473
4474
4475
4476 016214 012737 016302 000024  $PWUP: MOV      @SILLUP,@@PWVVEC ;; SET FOR FAST DOWN
4477 016222 013706 016306      MOV      $$SAVR6,SP    ;; GET SP
4478 016226 005037 016306      CLR      $$SAVR6      ;; WAIT LOOP FOR THE TTY
4479 016232 005237 016306 1$: INC      $$SAVR6    ;; WAIT FOR THE INC
4480 016236 001375      BNE     1$            ;; OF WORD
4481 016240 012677 162674      MOV      (SP)+,@SWR    ;; POP STACK INTO @SWR
4482 016244 012605      MOV      (SP)+,R5     ;; POP STACK INTO R5
4483 016246 012604      MOV      (SP)+,R4     ;; POP STACK INTO R4
4484 016250 012603      MOV      (SP)+,R3     ;; POP STACK INTO R3
4485 016252 012602      MOV      (SP)+,R2     ;; POP STACK INTO R2
4486 016254 012601      MOV      (SP)+,R1     ;; POP STACK INTO R1
4487 016256 012600      MOV      (SP)+,R0     ;; POP STACK INTO R0
4488 016260 012737 016142 000024  MOV      @SPWRDN,@@PWVVEC ;; SET UP THE POWER DOWN VECTOR
4489 016266 012737 000340 000026  MOV      @340,@@PWVVEC+2 ;; @PWVVEC+2
4490 016274 104400      TYPE     $POWER      ;; REPORT THE POWER FAILURE
4491 016276 016310  $PWMSG: .WORD    $POWER ;; POWER FAIL MESSAGE POINTER
4492 016300 000002      RTI
4493 016302 000000  $ILLUP: HALT        ;; THE POWER UP SEQUENCE WAS STARTED
4494 016304 000776      BR      -2            ;; BEFORE THE POWER DOWN WAS COMPLETE
4495 016306 000000  $$SAVR6: 0           ;; PUT THE SP HERE
4496 016310 005015 047520 042527  $POWER: .ASCIZ <15><12>"POWER"
4497 016316 000122
4498
4499
4500
4501
4502
4503
4504
4505
4506
4507 016320 010046 000002  $TRAP: MOV      R0,-(SP)    ;; SAVE R0
4508 016322 016600      MOV      2(SP),R0     ;; GET TRAP ADDRESS
4509 016326 005740      TST     -(R0)        ;; BACKUP BY 2
4510 016330 111000      MOV     B(R0),R0     ;; GET RIGHT BYTE OF TRAP
4511 016332 006300      ASL    R0            ;; POSITION FOR INDEXING
4512 016334 016000 016342  MOV      $TRAPAD(R0),R0 ;; INDEX TO TABLE
4513 016340 000200      RTS     R0          ;; GO TO ROUTINE
4514
4515
4516
4517
4518
4519
4520
4521

```

\*\*\*\*\*

POWER UP ROUTINE

SPWRMG: .WORD \$POWER

;; THE POWER UP SEQUENCE WAS STARTED  
;; BEFORE THE POWER DOWN WAS COMPLETE  
;; PUT THE SP HERE

.SBTTL EVEN  
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.

.SBTTL TRAP TABLE

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

ROUTINE  
-----

4522	016342				
4523	016342	015412			
4524	016344	013534			
4525	016346	013510			
4526	016350	013550			
4527	016352	013736			
4528					
4529	016354	014706			
4530					
4531	016356	014636			
4532	016360	015120			
4533	016362	015240			
4534	016364	005015	046103	041517	EM1:
4535	016372	020113	051123	043040	
4536	016400	047125	052103	047511	
4537	016406	020116	051105	047522	
4538	016414	000122			
4539	016416	005015	046103	041517	EM2:
4540	016424	020113	051123	042040	
4541	016432	052101	020101	051105	
4542	016440	047522	000122		
4543	016444	005015	046103	041517	EM3:
4544	016452	020113	051102	042040	
4545	016460	052101	020101	051105	
4546	016466	047522	000122		
4547	016472	005015	047503	047125	EM5:
4548	016500	020124	042522	027107	
4549	016506	042440	051122	051117	
4550	016514	000			
4551	016515	015	041412	047514	EM11:
4552	016522	045503	041440	052517	
4553	016530	052116	042440	051122	
4554	016536	051117	000040		
4555	016542	005015	046103	041517	EM12:
4556	016550	020113	047503	047125	
4557	016556	020124	052506	041516	
4558	016564	044524	047117	042440	
4559	016572	051122	051117	000	
4560	016577	015	041412	047514	EM16:
4561	016604	045503	044440	052116	
4562	016612	051105	052522	052120	
4563	016620	042440	051122	051117	
4564	016626	000040			
4565	016630	005015	046103	041517	EM20:
4566	016636	020113	042522	042520	
4567	016644	052101	041101	046111	
4568	016652	052111	020131	051105	
4569	016660	047522	020122	000	
4570	016665	015	041412	047514	EM26:
4571	016672	045503	040440	042104	
4572	016700	042522	051523	047111	
4573	016706	020107	051105	047522	
4574	016714	000122			
4575					
4576	016716	005015	051105	050122	DM1:
4577	016724	004503	051501	004522	

\*TRPAD:

\$TYPE	::CALL=TYPE	TRAP+0(104400)	TTY TYPEOUT ROUTINE
\$TYPOC	::CALL=TYPOC	TRAP+1(104401)	TYPE OCTAL NUMBER (WITH LEADING ZEROS)
\$TYPOS	::CALL=TYPOS	TRAP+2(104402)	TYPE OCTAL NUMBER (NO LEADING ZEROS)
\$TYPON	::CALL=TYPON	TRAP+3(104403)	TYPE OCTAL NUMBER (AS PER LAST CALL)
\$TYPBN	::CALL=TYPBN	TRAP+4(104404)	TYPE BINARY (ASCII) NUMBER
\$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
.ASCIZ	<15><12>/CLOCK	SR FUNCTION ERROR/	
.ASCIZ	<15><12>/CLOCK	SR DATA ERROR/	
.ASCIZ	<15><12>/CLOCK	BR DATA ERROR/	
.ASCIZ	<15><12>/COUNT	REG. ERROR/	
.ASCIZ	<15><12>#CLOCK	COUNT ERROR #	
.ASCIZ	<15><12>#CLOCK	COUNT FUNCTION ERROR#	
.ASCIZ	<15><12>#CLOCK	INTERRUPT ERROR #	
.ASCIZ	<15><12>#CLOCK	REPEATABILITY ERROR #	
.ASCIZ	<15><12>#CLOCK	ADDRESSING ERROR#	
.ASCIZ	<15><12>#ERRPC	ASR WAS S/Bx	

4578	016732	040527	004523	027523						
4579	016740	000102								
4580	016742	005015	051105	050122	DH3:	.ASCIZ	<15><12>	#ERRPC	ABR	WAS S/B#
4581	016750	004503	041101	004522						
4582	016756	040527	004523	027523						
4583	016764	000102								
4584	016766	005015	051105	050122	DH4:	.ASCIZ	<15><12>	#ERRPC	ASR	WAS S/B#
4585	016774	004503	051501	004522						
4586	017002	040527	004523	027523						
4587	017010	000102								
4588	017012	005015	051105	050122	DH12:	.ASCIZ	<15><12>	#ERRPC	ASR	#
4589	017020	004503	051501	004522						
4590	017026	000								
4591	017027	015	042412	051122	DH20:	.ASCIZ	<15><12>	#ERRPC	ASR	2ND CNT 1STNCT 3RD CNT#
4592	017034	041520	040411	051123						
4593	017042	031011	042116	047103						
4594	017050	004524	051461	047124						
4595	017056	052103	031411	042122						
4596	017064	047103	000124							
4597	017070	005015	051105	050122	DH26:	.ASCIZ	<15><12>	#ERRPC	CLOCK ADDR.#	
4598	017076	004503	046103	041517						
4599	017104	020113	042101	051104						
4600	017112	000056								
4601										
4602										
4603										.EVEN
4604	017114	001116	001376	001126	DT1:	.WORD	SERRPC, ASR, \$BODAT, \$GDDAT, 0			
4605	017122	001124	000000							
4606	017126	001116	001400	001126	DT3:	.WORD	SERRPC, ABR, \$BODAT, \$GDDAT, 0			
4607	017134	001124	000000							
4608	017140	001116	001376	000000	DT12:	.WORD	SERRPC, ASR, 0			
4609	017146	001116	001376	001126	DT20:	.WORD	SERRPC, ASR, \$BODAT, \$GDDAT, \$TMPD, 0			
4610	017154	001124	001420	000000						
4611	017162	001116	001376	001126	DT22:	.WORD	SERRPC, ASR, \$BODAT, \$TMPD, 0			
4612	017170	001420	000000							
4613	017174	001116	001420	000000	DT26:	.WORD	SERRPC, \$TMPD, 0			
4614										
4615	017202	000000	000000		DF0:	.WORD	0, 0			
4616										
4617		000001								.END



1584	1611*	1612*	1614	1627*	1629	1656*	1657*	1659	1672*	1674	1701*	1702*
1704	1717*	1719	1332*	1833*	1838	1871*	1873*	1877*	1883*	1885	1908*	1913*
1917	1921*	1925*	1928*	1953*	1958*	1962	1966*	1970*	1973*	2004*	2008	2077*
2078*	2080	2112*	2113*	2114	2127*	2138*	2140*	2141*	2143	2147*	2151*	2154*
2179*	2184*	2186*	2202	2206*	2210*	2213*	2227*	2245*	2248*	2250*	2252	2281*
2285*	2287*	2291	2307*	2309*	2311*	2313	2324*	2341*	2343*	2349	2353*	2357*
2360*	2376*	2392*	2394*	2400	2404*	2408*	2411*	2427*	2443*	2445*	2451	2455*
2459*	2462*	2478*	2494*	2496*	2502	2506*	2510*	2513*	2529*	2545*	2547*	2553
2557*	2561*	2564*	2580*	2600*	2602*	2608	2612*	2616*	2619*	2635*	2651*	2653*
2654*	2655	2659*	2663*	2666*	2682*	2692*	2695*	2697*	2703	2707*	2711*	2714*
2717	2729*	2737*	2739*	2744	2775*	2778*	2779*	2803*	2816*	2818*	2819*	2842*
2853*	2854*	2855*	2856*	2858	2861	2879*	2881*	2887	2890	2906*	2907*	2908*
2910*	2912*	2915	2927*	2935*	2939*	2940*	2944	2948*	2952*	2955*	2966*	2973*
2975*	2976*	2979*	2985	2989*	2993*	2996*	3008*	3018*	3020*	3021*	3027*	3033
3037*	3041*	3044*	3055*	3070	3074*	3078*	3081*	3095*	3104*	3106*	3107*	3113*
3119	3123*	3127*	3130*	3151*	3156	3160*	3164*	3167*	3181*	3190*	3192*	3193*
3199*	3205	3209*	3213*	3216*	3237*	3242	3246*	3250*	3253*	3267*	3277*	3279*
3280*	3286*	3292	3296*	3300*	3303*	3324*	3329	3333*	3337*	3340*	3354*	3363*
3365*	3366*	3373*	3381	3385*	3389*	3392*	3414*	3421	3425*	3429*	3432*	3446*
3471*	3473*	3476*	3480*	3496	3500*	3504*	3507*	3534*	3536*	3539*	3543*	3559
3563*	3567*	3570*	3584*	3585	3605*	3607*	3621	3625*	3629*	3632*	3647	3676*
3680	3726*	3815*	3817*	3818*	3819*	3820*	3856*	3858*	3859*	3862	3894*	3895*
3836	3900	4604	4608	4609	4611							

ASWREG= 000000  
ATESTM= 000000  
AUNIT = 000000  
ALSWR = 000000  
AVECT1= 000440  
AVECT2= 000000  
BIT0 = 000001

926	939											
926	930											
926	933											
926	940											
840*	926	965	1071	1072	1073	1074						
926	966											
823*	1702	1703	1717	1913	1920	1958	1965	2140	2146	2184	2205	2248
2285	2291	2313	2343	2352	2394	2403	2445	2454	2496	2505	2547	2556
2602	2611	2653	2658	2695	2706	2818	2862	2906	2910	2912	2940	2947
2976	2988	3036	3073	3122	3159	3208	3245	3295	3332	3384	3424	3499
3562	3624											

BIT00 = 000001  
BIT01 = 000002  
BIT02 = 000004  
BIT03 = 000010  
BIT04 = 000020  
BIT05 = 000040  
BIT06 = 000100  
BIT07 = 000200  
BIT08 = 000400  
BIT09 = 001000  
BIT1 = 000002  
BIT10 = 002000  
BIT11 = 004000  
BIT12 = 010000  
BIT13 = 020000  
BIT14 = 040000  
BIT15 = 100000  
BIT2 = 000004

813*	823	2114										
812*	822											
811*	821											
810*	820											
809*	819											
808*	818											
807*	817											
806*	816											
805*	815	4147										
804*	814	4054	4155									
822*	1657	1658	1672	2190								
803*	2026	2070	2979	3027	3065	3113	3151	3199	3237	3286	3324	3373
3414	3476	3539	4031									
802*	1387	1388	1402	1920	1965	2026	2070	2146	2205	2352	2403	2454
2505	2556	2611	2658	2706	2939	2947	2975	2988	3020	3036	3073	3106
3122	3159	3192	3208	3245	3279	3295	3332	3365	3384	3424	3499	3562
3624	4162											
801*	2026	2070	2595	2858	2862	2887	2917	3900				
800*	1342	1343	1357	2112	4038							
799*	1297	1298	1312	2818	4133							
798*	2862											
821*	1612	1613	1627	1920	1965	2095	2146	2205	2352	2403	2454	2505



IOTVEC=	000020	831#	1108#	1109#										
LF =	000012	737#	4393	4399										
LOOP	002260	1188#	3706	3780										
MDEVCT	001434	1084#	1183#	3671	3674#	3730#								
PC =	x000007	757#	3747#	3750#	3774#	3779	4040#	4046#	4106#	4235#	4348#	4367#	4374#	4381#
		4395#	4397#	4430#	4447#									
PIRQ =	177772	743#												
PIRQVE=	000240	837#												
PRIOR	001412	1075#												
PRO =	000000	760#												
PR1 =	000040	761#												
PR2 =	000100	762#												
PR3 =	000140	763#												
PR4 =	000200	764#												
PR5 =	000240	765#												
PR6 =	000300	766#												
PR7 =	000340	767#												
PS =	177776	740#	741											
PSW =	177776	741#												
PWRVEC=	000024	832#	1114#	1115#	4460#	4461#	4470#	4476#	4488#	4489#				
ROCHR =	104407	4294	4532#											
ROLIN =	104410	4533#												
RESVEC=	000010	827#												
ROTATE	001426	1081#	1186#	1187	3673#	3685	3731#	4066						
RSTART	002070	715	1171#											
RO =	x000000	748#	2344#	2346#	2395#	2397#	2446#	2448#	2497#	2499#	2548#	2550#	2603#	2605#
		2698#	2700#	2740#	2741#	2882#	2883#	2941#	2942#	2977#	2982#	3025#	3028#	3062#
		3066#	3111#	3114#	3148#	3152#	3197#	3200#	3234#	3238#	3284#	3287#	3321#	3325#
		3371#	3374#	3411#	3415#	3475#	3477#	3538#	3540#	3771#	3774	4078	4079#	4080#
		4087#	4089#	4089#	4090#	4091#	4092	4097	4102#	4104#	4108	4110	4341	4342#
		4347	4352	4355#	4407	4415#	4419	4420	4422#	4423#	4424	4446#	4462	4487#
		4507	4508#	4509	4510#	4511#	4512#	4513#						
R1 =	x000001	749#	3369#	3377#	3410#	3417#	3995	3996#	3999#	4005#	4408	4445#	4463	4486#
R2 =	x000002	750#	4464	4485#										
R3 =	x000003	751#	3942	3951#	3957#	3958#	3961#	3966#	3967#	3968	3977#	4290	4291#	4292
		4295#	4296	4300	4302	4304#	4306#	4465	4484#					
R4 =	x000004	752#	3943	3945#	3946#	3947#	3948	3949#	3963	3965#	3973#	3976#	4466	4483#
R5 =	x000005	753#	3944	3950#	3952#	3954#	3955#	3956#	3957	3975#	4467	4482#		
R6 =	x000006	754#	756	1102#	1103#	1104								
R7 =	x000007	755#	757											
SP =	x000006	756#	1106#	1123#	1131#	1135	1144#	1145#	1215#	1226#	1242	1251#	1262#	1278
		2770#	2771#	2782#	2783#	2789#	2790#	2802#	2811#	2812#	2821#	2822#	2828#	2829#
		2841#	3691#	3702#	3709#	3719#	3757#	3763#	3769#	3934#	3935	3936	3937#	3942#
		3743#	3944#	3950	3975	3976	3977	3978#	3979#	3995#	3996	4005	4006#	4007#
		4035	4056#	4059#	4078#	4083#	4104	4108#	4138#	4141	4143	4144	4173	4174
		4178#	4195#	4196#	4197	4204#	4207#	4208#	4212#	4213#	4217	4220#	4224	4226
		4228	4229#	4236	4238	4240#	4241	4243#	4244#	4245#	4246#	4247#	4262#	4263#
		4266#	4267#	4268	4272#	4273#	4274	4277	4279	4281#	4290#	4295	4306	4307#
		4308#	4309#	4341#	4342	4352#	4354	4355	4356#	4358	4360	4362	4368	4370#
		4372#	4380#	4384	4388	4389	4393	4407#	4408#	4415	4416#	4427	4428#	4429#
		4439	4440#	4445	4446	4462#	4463#	4464#	4465#	4466#	4467#	4468#	4469	4477#
		4481	4482	4483	4484	4485	4486	4487	4507#	4508				
STACK =	001100	731#	1106											
STAPT =	001472	714	1096#											
ST*LMT=	177774	742#												
SWR	001140	906#	1104	1125#	1127	1133#	1140#	1162	2595	4031	4038	4050	4054	4133

		4147	4149	4155	4162	4191	4228*	4468	4481*
SWREG	000176	708#	1133	1162	4191	4204			
SW0	= 000001	795#							
SW00	= 000001	785#	795						
SW01	= 000002	784#	794						
SW02	= 000004	783#	793						
SW03	= 000010	782#	792						
SW04	= 000020	781#	791						
SW05	= 000040	780#	790						
SW06	= 000100	779#	789						
SW07	= 000200	778#	788						
SW08	= 000400	777#	787						
SW09	= 001000	776#	786						
SW1	= 000002	794#							
SW10	= 002000	775#							
SW11	= 004000	774#							
SW12	= 010000	773#							
SW13	= 020000	772#							
SW14	= 040000	771#							
SW15	= 100000	770#							
SW2	= 000004	793#							
SW3	= 000010	792#							
SW4	= 000020	791#							
SW5	= 000040	790#							
SW6	= 000100	789#							
SW7	= 000200	788#							
SW8	= 000400	787#							
SW9	= 001000	786#							
TBITVE	= 000014	828#							
TKVEC	= 000060	835#							
TPVEC	= 000064	836#							
TRAPVE	= 000034	834#	1112*	1113*					
TRIVEC	= 000014	829#							
TSTCNT	001436	1085#	1090*	1093*	1097*	3671			
TSTSTR	001442	723	1089#						
TST1	002346	1207#							
TST10	003170	1518#							
TST11	003266	1563#							
TST12	003364	1608#							
TST13	003462	1653#							
TST14	003560	1698#							
TST15	003656	1739#							
TST16	003746	1778#							
TST17	004036	1829#							
TST2	002440	1248#							
TST20	004110	1868#							
TST21	004172	1905#							
TST22	004320	1950#							
TST23	004446	2000#							
TST24	004534	2022	2025	2028	2046#				
TST25	004606	2069	2075#						
TST26	004656	2092	2094	2096	2110#				
TST27	004712	2136#							
TST3	002502	1293#							
TST30	005022	2176#							
TST31	005232	2200	2236#						

TST32	005336	2260	2262	2265	2279*													
TST33	005400	2305*																
TST34	005446	2337*																
TST35	005576	2388*																
TST36	005726	2439*																
TST37	006056	2490*																
TST4	002600	1338*																
TST40	006206	2541*																
TST41	006336	2552*																
TST42	006476	2598	2647*															
TST43	006626	2699*																
TST44	006770	2735*																
TST45	007030	2766*																
TST46	007150	2808*																
TST47	007254	2851*																
TST5	002676	1383*																
TST50	007330	2876*																
TST51	007414	2904*																
TST52	007506	2932*																
TST53	007642	2971*																
TST54	007774	3015*																
TST55	010266	3101*																
TST56	010560	3187*																
TST57	011052	3274*																
TST6	002774	1428*																
TST60	011344	3360*																
TST61	011656	3468*																
TST62	012050	3531*																
TST63	012264	3556	3582	3586	3599*													
TST7	003072	1473*																
TYPBN =	104404	3770	4527*															
TYPE =	104400	1156	1174	1179	3609	3616	3687	3693	3698	3715	3721	3753	3759	3765				
		3969	4002	4033	4041	4077	4094	4096	4099	4101	4105	4112	4202	4203				
		4206	4219	4230	4249	4298	4301	4305	4363	4490	4523*							
		3692	3703	3720	3758	3764	4085	4109	4205	4524*								
TYPOC =	104401	4526*																
TYPON =	104403	4525*																
TYPOS =	104402	4525*																
UTEST	001430	1082*	1187*	3685*	3769	4066*												
VECTP	001404	1072*	1191*	1192*														
VECT1	001402	1071*	1149*	1190*	1191	1193	2780*	3705*	3727*									
VECT2	001406	1073*	1193*	1194*	1195	2817*												
VECT2P	001410	1074*	1195*	1196*														
WSTART=	001456	721	1092*															
\$APTHO	001000	856	872*															
\$ASTAT=	***** U	443*	4448															
\$ATYC	015720	440.	4406*															
\$ATY1	015674	4402*																
\$ATY3	015702	4348	4403*															
\$ATY4	015712	4046	4405*															
\$AUT08	001134	903*	1166*	4199	4320													
\$BASE	001250	967*	1150	3726														
\$BOR	001122	898*																
\$BOAT	001126	900*	1299*	1300	1314*	1344*	1345	1359*	1389*	1390	1404*	1434*	1435	1449*				
		1479*	1480	1494*	1524*	1525	1539*	1569*	1570	1584*	1614*	1615	1629*	1659*				
		1660	1674*	1704*	1705	1719*	1744*	1746	1761*	1783*	1785	1800*	1838*	1840				
		1843	1885*	1886	1888	1926*	1929	1931	1971*	1974	1976	2008*	2054*	2152*				









ACTST	1201#	1202	1244												
BUFLO	1734#	1735	1774												
COMMEM	1#	838#													
COUNTM	2326#	2328	2379	2430	2481	2532	2583	2638							
CSROTA	1282#	1283	1329	1373	1418	1463	1508	1553	1598	1643	1688				
DFC	988#	994	1002	1010	1018	1026	1034	1042	1050	1058	1066				
DIVCM	3010#	3011	3097	3183	3270	3356									
ECB	682#	1229	1239	1265	1275	1302	1307	1317	1323	1347	1352	1362	1368	1392	1397
	1407	1413	1437	1442	1452	1458	1482	1487	1497	1503	1527	1532	1542	1548	1572
	1577	1587	1593	1617	1622	1632	1638	1662	1667	1677	1693	1707	1712	1722	1738
	1749	1754	1764	1769	1788	1793	1803	1808	1847	1854	1891	1897	1935	1942	1980
	1987	2012	2019	2029	2034	2058	2064	2093	2089	2097	2102	2117	2124	2157	2162
	2192	2197	2218	2222	2255	2256	2271	2294	2315	2320	2367	2372	2418	2423	2469
	2474	2520	2525	2571	2576	2626	2631	2673	2678	2721	2726	2747	2752	2793	2797
	2832	2836	2863	2868	2892	2897	2919	2924	2958	2963	3000	3005	3052	3058	3095
	3032	3138	3144	3171	3178	3224	3230	3257	3264	3311	3317	3344	3351	3400	3406
	3436	3443	3485	3490	3511	3517	3548	3553	3573	3579	3588	3592	3636	3641	3649
	3654														
ENDCOM	1#	838#													
ENDPAS	3732#	3753													
ERROR	732#	1232	1268	1305	1320	1350	1365	1395	1410	1440	1455	1485	1500	1530	1545
	1575	1590	1620	1635	1665	1680	1710	1725	1752	1767	1791	1806	1850	1894	1938
	1983	2016	2032	2061	2086	2100	2120	2160	2195	2221	2258	2269	2297	2318	2370
	2421	2472	2523	2574	2629	2676	2724	2750	2796	2835	2866	2895	2922	2961	3003
	3055	3098	3141	3174	3227	3260	3314	3347	3403	3439	3488	3514	3551	3576	3591
	3639	3652	3823	3864	3898	3902									
ESCAPE	1#	838#													
GETPRI	1#	838#													
GETSWR	1#	838#	1157#												
INSTR2	4011#	4061													
LOCKM	2762#	2769	2781	2788	2810	2820	2827								
MULT	1#	838#													
NEWTST	1#	838#	1204	1245	1285	1330	1375	1420	1465	1510	1555	1600	1645	1690	1736
	1775	1820	1859	1902	1947	1993	2038	2072	2107	2133	2167	2233	2276	2302	2329
	2380	2431	2482	2533	2584	2639	2686	2732	2763	2805	2848	2873	2901	2929	2968
	3012	3098	3184	3271	3357	3450	3523	3596							
POP	1#	838#	4445	4446	4481	4482									
POPSP2	683#	1225	1261	2801	2840	3708									
PR	684#	1143													
PUSH	1#	838#	4406	4408	4429	4462	4468								
ROCLK	685#	1917	1962	2143	2202	2349	2400	2451	2502	2553	2608	2655	2703	2944	2985
	3033	3070	3119	3156	3205	3242	3292	3329	3381	3421	3495	3559	3621		
REPORT	1#	838#													
SCOPE	733#	1248	1293	1338	1383	1428	1473	1518	1563	1608	1653	1698	1739	1778	1829
	1868	1905	1950	2000	2046	2075	2110	2136	2176	2236	2279	2305	2337	2388	2439
	2490	2541	2592	2647	2689	2735	2766	2808	2851	2876	2904	2932	2971	3015	3101
	3187	3274	3360	3468	3531	3599	3664								
SETPRI	1#	838#													
SETTRA	4515#	4524	4525	4526	4527	4529	4531	4532	4533						
SETUP	1#	838#	1099												
SKIP	1#	838#	2022	2025	2028	2069	2092	2094	2096	2200	2260	2262	2265	2598	3556
	3502	3586													
SLASH	1#	838#													
SPACE	838#														
STARS	1#	838#	848	859	861	868	881	922	925	1204	1206	1245	1247	1285	1292
	1330	1337	1375	1382	1420	1427	1465	1472	1510	1517	1555	1562	1600	1607	1645



.SRAND	18		
.SRDE	18		
.SRDOC	18	675#	
.SREAO	18	677#	4181
.SRZAZ	18		
.SSAVE	18		
.SSS20	18		
.SSS20	18		
.SSCOP	18	677#	4116
.SSITE	18		
.SSUPR	18		
.STRAP	18	675#	4499
.STYPB	18	675#	3986
.STYPO	18	677#	
.STYFE	18	677#	4320
.STYPO	18	676#	3909
.S400A	18		
.1170	18		

ROCB	4001														
ROO	1192	1194	1196	1200	1226	1262	2802	2841	3676	3705	3709	3937	3947	4091	4220
ASL	4229	4256	4416	4428	4440										
XSR	3673	4088	4089	4090	4243	4244	4245	4511							
BEQ	4423														
	1139	1161	1173	1301	1315	1346	1360	1391	1405	1436	1450	1481	1495	1526	1540
	1571	1585	1516	1630	1661	1675	1706	1720	1747	1762	1786	1801	1845	1889	1933
	1978	2010	2025	2028	2056	2069	2094	2189	2217	2239	2262	2292	2598	2918	2957
	3051	3084	3137	3170	3223	3256	3310	3343	3398	3435	3484	3510	3547	3572	3586
	3603	3635	3672	3712	3772	3822	3901	3964	4000	4029	4032	4055	4058	4093	4098
	4111	4148	4150	4152	4156	4165	4200	4227	4242	4346	4359	4394	4410	4414	4434
	4436														
BGE	4168														
BGT	3749	3971	4239	4280											
BHI	4154														
BIC	1190	1312	1357	1402	1447	1492	1537	1582	1627	1672	1717	1759	1798	1919	1964
	2145	2204	2351	2402	2453	2504	2555	2610	2657	2705	2910	2946	2987	3035	3072
	3121	3158	3207	3244	3294	3331	3383	3423	3498	3561	3623	3746	3961	4066	4196
	4213	4240	4267	4273	4281										
BIS	1297	1342	1387	1432	1477	1522	1567	1612	1657	1702	1913	1920	1925	1958	1965
	1970	2026	2070	2078	2113	2140	2141	2146	2151	2184	2186	2205	2210	2243	2248
	2250	2285	2287	2309	2311	2352	2357	2403	2408	2454	2459	2505	2510	2556	2561
	2611	2616	2654	2658	2663	2695	2697	2706	2711	2739	2779	2819	2855	2856	2881
	2907	2908	2912	2940	2947	2952	2975	2976	2979	2988	2993	3020	3021	3027	3036
	3041	3065	3073	3078	3106	3107	3113	3122	3127	3151	3159	3164	3192	3193	3199
	3208	3213	3237	3245	3250	3279	3280	3286	3295	3300	3324	3322	3337	3365	3366
	3373	3384	3389	3414	3424	3429	3476	3480	3499	3504	3539	3543	3562	3567	3624
	3629	3685	3818	3820	3859	3835	3966	3967	4247						
BISB	4230														
BIT	2027	2095	2114	2190	2291	2313	2595	2858	2887	2917	3900	4031	4038	4054	4133
	4147	4155	4162												
BITB	1138	4345	4350	4382	4413										
BLOS	4293														
BLT	3972	4237	4278	4373											
BMI	2081	2253	2265	2365	2416	2467	2518	2569	2624	2671	2745	3648	3668	3863	3897
BME	1105	1128	1155	1159	1163	2096	2115	2156	2191	2230	2314	2347	2362	2398	2413
	2449	2464	2500	2515	2551	2566	2606	2621	2668	2701	2716	2742	2859	2884	2888
	2943	2983	2998	3029	3067	3115	3153	3201	3239	3288	3326	3375	3378	3416	3418
	3478	3541	3684	3962	4039	4044	4063	4081	4103	4134	4163	4192	4198	4218	4225
	4232	4269	4275	4297	4303	4344	4351	4353	4361	4369	4383	4390	4412	4418	4421
	4438	4480													
BPL	2718	3614	3960	4051	4194	4210	4265	4271	4338	4387					
BR	1091	1095	1130	1165	1168	1175	1180	1223	1259	1310	1355	1400	1445	1490	1535
	1580	1625	1670	1715	1757	1796	2022	2092	2200	2225	2260	2800	2839	3061	3147
	3233	3320	3409	3493	3556	3582	3610	3617	3644	3688	3694	3699	3716	3722	3754
	3760	3766	3824	3855	3903	3938	3953	3974	4004	4049	4096	4113	4136	4142	4145
	4158	4161	4221	4248	4250	4276	4299	4340	4366	4376	4385	4392	4404	4426	4472
	4494														
CLC	4003														
CLR	1094	1098	1103	1116	1117	1137	1183	1184	1185	1296	1313	1341	1358	1386	1403
	1431	1448	1476	1493	1521	1538	1566	1583	1611	1628	1656	1673	1701	1718	1741
	1760	1780	1799	1832	1871	1908	1953	2003	2049	2127	2138	2139	2179	2180	2227
	2241	2245	2281	2307	2324	2341	2342	2344	2376	2392	2393	2395	2427	2443	2444
	2446	2478	2494	2495	2497	2529	2545	2546	2548	2580	2600	2601	2603	2635	2651
	2652	2682	2692	2693	2698	2720	2729	2737	2740	2775	2803	2816	2842	2853	2879
	2862	2927	2935	2936	2937	2941	2966	2973	2974	3008	3018	3019	3095	3104	3105

	3181	3190	3191	3267	3277	3278	3354	3363	3364	3446	3471	3472	3534	3535	3558
	3584	3587	3605	3620	3730	3743	3744	3815	3816	3856	3951	4079	4160	4175	4207
	4208	4478													
CLRB	1841	1887	4159	4304	4365	4391	4442	4443	4444						
CMP	1104	1127	1162	1300	1345	1390	1435	1480	1525	1570	1615	1660	1705	1746	1795
	1931	1976	2216	3048	3083	3134	3169	3220	3255	3307	3342	3396	3434	3483	3509
	3546	3671	3711	3821	4143	4167	4191	4197	4217	4224	4236	4238	4268	4274	4277
	4279	4292													
CMPB	1160	4043	4149	4153	4199	4231	4296	4302	4343	4358	4360	4368	4389	4393	4411
DEC	1823	3066	3152	3238	3325	3377	3415	3417	3747	4064	4087				
DECB	3959	3970	4372	4375											
EMT	732														
HALT	706	4052	4339	4471	4493										
INC	1089	1154	1873	2215	2346	2397	2448	2499	2550	2605	2700	2854	2982	3028	3114
	3200	3287	3374	3477	3540	3674	3697	3745	3965	3973	4034	4062	4166	4246	4441
	4479														
INCB	2741	2883	2942	4028	4171	4395									
IOT	733														
JMP	714	715	716	717	718	721	723	3706	3779						
JSR	3774	4040	4046	4235	4348	4367	4374	4381	4430						
MOV	1090	1093	1097	1102	1106	1108	1109	1110	1111	1112	1113	1114	1115	1119	1120
	1123	1124	1125	1126	1131	1133	1134	1135	1140	1144	1145	1149	1150	1186	1187
	1191	1193	1195	1199	1208	1209	1211	1215	1216	1227	1242	1251	1252	1263	1278
	1294	1296	1299	1314	1339	1343	1344	1359	1384	1388	1389	1404	1429	1433	1434
	1443	1474	1478	1479	1494	1519	1523	1524	1539	1564	1568	1569	1584	1609	1613
	1614	1629	1654	1658	1659	1674	1699	1703	1704	1719	1742	1743	1744	1761	1781
	1782	1783	1800	1830	1838	1840	1869	1885	1886	1906	1909	1915	1917	1918	1921
	1926	1928	1951	1954	1960	1962	1963	1966	1971	1973	2001	2004	2008	2047	2050
	2054	2077	2112	2143	2144	2147	2152	2154	2177	2191	2182	2202	2203	2206	2211
	2213	2228	2246	2283	2308	2338	2343	2349	2350	2353	2358	2360	2389	2394	2400
	2401	2404	2409	2411	2440	2445	2451	2452	2455	2460	2462	2491	2496	2502	2503
	2506	2511	2513	2542	2547	2553	2554	2557	2562	2564	2593	2602	2608	2609	2612
	2617	2619	2648	2653	2655	2656	2659	2664	2666	2690	2703	2704	2707	2712	2714
	2738	2767	2770	2771	2776	2778	2780	2782	2783	2789	2790	2811	2812	2817	2818
	2821	2822	2828	2829	2861	2862	2877	2880	2890	2891	2906	2915	2916	2933	2939
	2944	2945	2948	2953	2955	2977	2985	2986	2989	2994	2996	3016	3025	3032	3033
	3034	3037	3042	3044	3046	3062	3070	3071	3074	3079	3081	3102	3111	3118	3119
	3120	3123	3128	3130	3132	3148	3156	3157	3160	3165	3167	3198	3197	3204	3205
	3206	3209	3214	3216	3218	3234	3242	3243	3246	3251	3253	3275	3294	3291	3292
	3293	3296	3301	3303	3305	3321	3329	3330	3333	3338	3340	3361	3369	3371	3380
	3381	3382	3385	3390	3392	3394	3410	3411	3421	3422	3425	3430	3432	3469	3473
	3475	3481	3482	3496	3497	3500	3505	3507	3532	3536	3538	3544	3545	3559	3560
	3563	3568	3570	3585	3600	3606	3607	3621	3622	3625	3630	3632	3677	3678	3691
	3702	3704	3710	3719	3726	3727	3728	3731	3750	3757	3763	3769	3771	3817	3819
	3857	3858	3894	3934	3942	3943	3944	3950	3957	3975	3976	3977	3978	3979	3995
	3996	4005	4006	4007	4030	4035	4056	4059	4078	4083	4092	4097	4102	4104	4108
	4138	4139	4141	4144	4157	4169	4170	4173	4174	4177	4178	4204	4228	4233	4262
	4263	4290	4291	4306	4307	4308	4309	4341	4342	4347	4355	4370	4407	4408	4415
	4419	4424	4425	4427	4429	4439	4445	4446	4460	4461	4462	4463	4464	4465	4466
	4467	4468	4469	4470	4476	4477	4481	4482	4483	4484	4485	4486	4487	4488	4489
	4507	4508	4512												
MOV8	1118	1166	1210	1833	1877	3935	3936	3939	3940	3941	3945	3948	3949	3968	3998
	4037	4045	4172	4176	4195	4212	4266	4272	4295	4300	4352	4380	4388	4402	4403
	4405	4510													
NEG	3946														
NOP	1207	2787	2826	3742	3775	3776	3777	3860	3861						

RESET	2006	2052	3773												
RXL	3952	3954	3955	3956	3358	3999									
RTI	1132	1146	2772	2784	2791	2813	2823	2830	3980	4008	4067	4179	4234	4282	4310
	4357	4492													
RTS	4106	4397	4447	4513											
SEC	3997														
SUB	4036	4422													
TRAP	4515	4524	4525	4526	4527	4529	4531	4532	4533						
TST	1158	1172	1219	1255	1929	1974	2024	2068	2080	2093	2155	2188	2214	2229	2238
	2242	2261	2361	2412	2463	2514	2565	2620	2667	2715	2956	2997	3045	3082	3131
	3168	3217	3254	3304	3341	3393	3433	3508	3571	3602	3608	3615	3633	3647	3680
	3683	3862	3896	3963	4050	4057	4110	4140	4164	4226	4241	4354	4362	4384	4417
	4435	4437	4509												
TSTB	1843	1888	2252	2263	2363	2414	2465	2516	2567	2622	2669	2717	2744	3613	3667
	4151	4193	4209	4264	4270	4337	4386	4409	4420	4433					
.ASCII	919	920													
.ASCIZ	918	921	1170	1177	1182	3612	3619	3690	3696	3701	3718	3724	3756	3762	3768
	4114	4314	4315	4316	4318	4496	4534	4539	4543	4547	4551	4555	4560	4565	4570
	4576	4580	4584	4588	4591	4597									
.BLKB	4313														
.BYTE	888	889	894	895	903	904	912	913	914	915	937	938	948	949	956
	957	959	960	962	963	3781	3981	3982	3983	3984	4009	4047	4048	4311	4312
	4448	4449	4450												
.DSABL	4251														
.ENABL	1	673	674	4184											
.END	4617														
.ENDC	692	709	732	824	838	849	853	855	860	862	869	882	886	888	916
	917	918	919	923	926	948	956	959	962	965	966	967	968	969	972
	1089	1106	1107	1110	1112	1114	1116	1117	1119	1121	1142	1156	1162	1168	1170
	1177	1182	1205	1206	1207	1208	1209	1210	1213	1246	1247	1248	1249	1296	1287
	1292	1293	1294	1295	1331	1332	1337	1338	1339	1340	1376	1377	1382	1383	1384
	1395	1421	1422	1427	1428	1429	1430	1466	1467	1472	1473	1474	1475	1511	1512
	1517	1518	1519	1520	1556	1557	1562	1563	1564	1565	1601	1602	1607	1608	1609
	1610	1646	1647	1652	1653	1654	1655	1691	1692	1697	1698	1699	1700	1737	1738
	1739	1740	1776	1777	1778	1779	1821	1822	1828	1829	1830	1831	1860	1861	1867
	1868	1959	1870	1903	1904	1905	1906	1907	1948	1949	1950	1951	1952	1994	1995
	1999	2000	2001	2002	2023	2026	2029	2039	2040	2045	2046	2047	2048	2070	2073
	2074	2075	2076	2093	2095	2097	2108	2109	2110	2111	2134	2135	2136	2137	2168
	2169	2175	2176	2177	2178	2201	2234	2235	2236	2237	2261	2263	2266	2277	2278
	2279	2290	2303	2304	2305	2306	2330	2331	2336	2337	2338	2339	2340	2349	2381
	2382	2387	2388	2389	2390	2391	2400	2432	2433	2438	2439	2440	2441	2442	2451
	2483	2484	2489	2490	2491	2492	2493	2502	2534	2535	2540	2541	2542	2543	2544
	2553	2585	2586	2591	2592	2593	2594	2599	2608	2640	2641	2646	2647	2648	2649
	2650	2655	2687	2688	2689	2690	2691	2733	2734	2735	2736	2764	2765	2766	2767
	2768	2806	2807	2808	2809	2849	2850	2851	2852	2874	2875	2876	2877	2878	2902
	2903	2904	2905	2930	2931	2932	2933	2934	2969	2970	2971	2972	3013	3014	3015
	3016	3017	3024	3031	3052	3062	3068	3099	3100	3101	3102	3103	3110	3117	3138
	3148	3154	3185	3186	3187	3188	3189	3196	3203	3224	3234	3240	3272	3273	3274
	3275	3276	3283	3290	3311	3321	3327	3358	3359	3360	3361	3362	3370	3379	3400
	3411	3419	3451	3452	3467	3468	3469	3470	3524	3525	3530	3531	3532	3533	3557
	3583	3587	3597	3598	3599	3600	3601	3612	3619	3630	3696	3701	3718	3724	3737
	3738	3740	3743	3749	3752	3753	3756	3762	3768	3771	3773	3779	3781	3782	3912
	3989	4015	4018	4028	4035	4040	4041	4042	4050	4061	4069	4072	4087	4116	4119
	4122	4127	4133	4135	4146	4149	4150	4151	4153	4155	4162	4166	4171	4173	4177
	4180	4181	4184	4185	4187	4215	4251	4255	4283	4284	4291	4293	4296	4298	4314
	4320	4323	4352	4402	4403	4406	4433	4448	4459	4468	4469	4475	4481	4482	4492

	4499	4502	4508	4511	4523	4524	4525	4526	4527	4528	4529	4530	4531	4532	4533
.EQUIV	4534 732 814	733 815	741 816	756 817	757 818	786 819	787 820	788 821	789 822	790 823	791	792	793	794	795
.EVEN	926	1170	1177	1182	3612	3619	3690	3696	3701	3718	3724	3756	3762	3768	3782
.IF	4115 688 917 372 1160 1292 1427 1562 1697 1861 2001 2094 2235 2339 2442 2548 2654 2848 2970 3110 3273 3410 3596 3740 4014 4118 4174 4297 4481 4532	4451 709 918 1089 1169 1294 1429 1564 1699 1867 2002 2096 2237 2340 2446 2584 2696 2850 2972 3117 3275 3417 3598 3742 4017 4121 4179 4313 4482 4533	4498 730 922 1101 1176 1295 1430 1565 1700 1869 2022 2107 2260 2344 2482 2586 2698 2852 3012 3136 3276 3450 3600 3748 4028 4126 4180 4314 4490 4534	4602 796 923 1106 1181 1330 1465 1600 1736 1870 2025 2109 2262 2330 2484 2591 2690 2873 3014 3148 3283 3452 3601 3751 4031 4132 4181 4320 4492	824 925 1108 1203 1332 1467 1602 1738 1902 2028 2111 2265 2382 2439 2593 2691 2875 3016 3154 3290 3467 3611 3753 4038 4133 4183 4322 4496	848 948 1110 1204 1337 1472 1607 1740 1904 2038 2132 2276 2387 2491 2594 2732 2877 3017 3184 3309 3469 3618 3755 4040 4145 4186 4343 4501	851 956 1112 1206 1339 1474 1609 1775 1906 2040 2137 2278 2389 2492 2595 2734 2878 3024 3186 3321 3470 3619 3761 4041 4147 4187 4401 4507	853 959 1114 1208 1340 1475 1610 1777 1907 2045 2137 2280 2390 2493 2598 2736 2880 3031 3188 3327 3470 3619 3767 4043 4148 4187 4403 4511	859 962 1116 1209 1375 1510 1645 1779 1947 2047 2167 2302 2391 2497 2598 2603 2763 2903 3050 3189 3357 3525 3700 3771 4050 4149 4215 4406 4515	861 965 1117 1210 1377 1512 1647 1820 1949 2048 2169 2304 2395 2497 2533 2639 2765 2905 3062 3196 3359 3525 3717 3773 4054 4151 4254 4433 4524	868 966 1119 1245 1382 1517 1652 1822 1951 2069 2175 2306 2431 2535 2641 2767 2929 3068 3198 3361 3532 3723 3779 4061 4152 4255 4448 4525	891 967 1137 1247 1384 1519 1654 1828 1952 2072 2177 2329 2433 2540 2646 2768 2931 3098 3222 3362 3533 3736 3781 4069 4153 4293 4458 4526	895 968 1155 1249 1385 1520 1655 1830 1993 2074 2178 2331 2438 2542 2648 2805 2933 3100 3234 3369 3556 3737 3782 4071 4162 4291 4468 4527	887 969 1156 1285 1385 1420 1555 1690 1831 1995 2076 2178 2336 2440 2543 2649 2807 2934 3102 3240 3377 3582 3738 3911 4086 4164 4292 4469 4529	916 970 1157 1287 1422 1557 1692 1859 1999 2092 2233 2338 2441 2544 2650 2809 2968 3103 3271 3398 3586 3739 3988 4102 4172 4296 4474 4531
.IFF	730 1156 1294 1473 1647 1821 1951 2075 2176 2304 2439 2592 2735 2875 3013 3272 3525 3781 4180 4475	849 1204 1331 1474 1653 1822 1994 2076 2177 2305 2440 2593 2736 2876 3014 3273 3531 3912 4181 4492	853 1205 1332 1511 1654 1829 1995 2093 2201 2306 2446 2599 2764 2877 3015 3274 3532 3989 4184 4502	855 1206 1338 1512 1691 1830 2000 2095 2234 2330 2483 2603 2765 2902 3016 3275 3557 4015 4187 4508	860 1207 1339 1518 1692 1850 2001 2097 2235 2331 2484 2640 2766 2904 3050 3309 3583 4017 4255 3612	862 1208 1376 1519 1698 1861 2023 2108 2236 2337 2490 2641 2767 2905 3099 3358 3587 4031 4257 3690	869 1213 1377 1556 1699 1868 2026 2109 2237 2338 2491 2647 2806 2905 3100 3359 3597 4061 4262 3696	882 1245 1383 1557 1737 1869 2029 2110 2261 2344 2497 2648 2807 2930 3101 3360 3598 4072 4283 3701	885 1246 1384 1563 1738 1869 2039 2111 2263 2381 2534 2655 2808 2931 3102 3361 3599 4087 4284 3718	888 1247 1385 1564 1739 1903 2040 2134 2266 2382 2535 2687 2809 2932 3136 3400 3599 4087 4293 3724	916 1248 1422 1601 1740 1905 2046 2135 2277 2388 2541 2688 2849 2933 3185 3451 3600 4116 4297 3756	923 1249 1428 1602 1776 1906 2047 2136 2278 2389 2542 2689 2850 2969 3186 3452 3600 4119 4314 3762	926 1286 1429 1608 1777 1948 2070 2137 2279 2389 2548 2690 2851 2970 3187 3468 3600 4146 4323 3768	1106 1287 1466 1609 1778 1949 2073 2168 2280 2432 2585 2733 2852 2971 3188 3469 3600 4149 4402 4041	1155 1293 1467 1646 1779 1950 2074 2169 2303 2433 2586 2734 2974 2972 3222 3524 3752 4150 4459 4161
.IFT	1170	1177	1182	3612	3619	3690	3696	3701	3718	3724	3756	3762	3768	4041	4161

.IFTF	4257	4262													
	1170	1177	1182	3612	3619	3690	3696	3701	3718	3724	3756	3762	3768	4040	4159
.IIF	4202	4255	4258												
	687	692	697	698	706	922	926	1107	1110	1116	1117	1119	1120	1156	3692
	3703	3720	3738	3743	3744	3758	3764	3770	3781	3792	4018	4019	4020	4021	4022
	4027	4053	4061	4069	4084	4109	4122	4123	4124	4125	4126	4127	4131	4160	4161
	4177	4180	4181	4184	4205	4306	4314	4320	4399	4523	4524	4525	4526	4527	4529
	4531	4532	4533												
.IRP	1089	1204	1245	1285	1330	1375	1420	1465	1510	1555	1600	1645	1690	1736	1775
	1820	1859	1902	1947	1993	2038	2072	2107	2133	2167	2233	2276	2302	2329	2380
	2431	2482	2533	2584	2639	2686	2732	2763	2805	2848	2873	2901	2929	2968	3012
	3098	3184	3271	3357	3450	3523	3596	3742	4061	4132	4407	4408	4429	4445	4446
	4462	4468	4481	4482											
.LIST	1	672	706	838	916	923	926	1089	1121	1156	1157	1170	1177	1182	1204
	1208	1230	1232	1240	1242	1245	1249	1266	1268	1276	1278	1284	1285	1294	1296
	1303	1305	1308	1310	1318	1320	1324	1326	1329	1330	1339	1341	1348	1350	1353
	1355	1363	1365	1369	1371	1374	1375	1384	1386	1393	1395	1398	1400	1408	1410
	1414	1416	1419	1420	1429	1431	1438	1440	1443	1445	1453	1455	1459	1461	1464
	1465	1474	1476	1483	1485	1488	1490	1498	1500	1504	1506	1509	1510	1519	1521
	1528	1530	1533	1535	1543	1545	1549	1551	1554	1555	1564	1566	1573	1575	1578
	1580	1588	1590	1594	1596	1599	1600	1609	1611	1618	1620	1623	1625	1633	1635
	1639	1641	1644	1645	1654	1656	1663	1665	1668	1670	1678	1680	1684	1686	1689
	1690	1699	1701	1708	1710	1713	1715	1723	1725	1729	1731	1736	1737	1750	1752
	1755	1757	1765	1767	1770	1772	1775	1779	1789	1791	1794	1796	1804	1806	1809
	1811	1820	1830	1848	1850	1855	1857	1859	1869	1872	1894	1898	1900	1902	1906
	1936	1938	1943	1945	1947	1951	1981	1983	1988	1990	1993	2001	2013	2015	2020
	2022	2030	2032	2035	2037	2038	2047	2059	2061	2065	2067	2072	2076	2084	2086
	2090	2092	2098	2100	2103	2105	2107	2111	2118	2120	2125	2127	2133	2137	2158
	2160	2163	2165	2167	2177	2193	2195	2198	2200	2209	2221	2223	2225	2233	2237
	2256	2258	2267	2269	2272	2274	2276	2280	2295	2297	2302	2306	2316	2318	2321
	2323	2329	2338	2368	2370	2373	2375	2380	2389	2419	2421	2424	2426	2431	2440
	2470	2472	2475	2477	2482	2491	2521	2523	2526	2528	2533	2542	2572	2574	2577
	2579	2584	2593	2627	2629	2632	2634	2639	2648	2674	2676	2679	2681	2686	2690
	2722	2724	2727	2729	2732	2736	2748	2750	2753	2755	2763	2767	2794	2796	2798
	2800	2805	2809	2833	2835	2837	2839	2848	2852	2864	2866	2869	2871	2873	2877
	2893	2895	2898	2900	2901	2905	2920	2922	2925	2927	2929	2933	2959	2961	2964
	2966	2968	2972	3001	3003	3006	3008	3012	3016	3053	3055	3059	3061	3086	3088
	3093	3095	3098	3102	3139	3141	3145	3147	3172	3174	3179	3181	3184	3188	3225
	3227	3231	3233	3258	3260	3265	3267	3271	3275	3312	3314	3318	3320	3345	3347
	3352	3354	3357	3361	3401	3403	3407	3409	3437	3439	3444	3446	3450	3469	3486
	3488	3491	3493	3512	3514	3518	3520	3523	3532	3549	3551	3554	3556	3574	3576
	3580	3582	3589	3591	3593	3595	3596	3600	3612	3619	3637	3639	3642	3644	3650
	3652	3655	3657	3690	3696	3701	3718	3724	3743	3756	3762	3768	3773	4061	4126
	4283	4515	4523	4524	4525	4526	4527	4528	4529	4530	4531	4532	4533	4534	
.MACRO	1	682	683	684	685	879	988	1137	1201	1281	1282	1734	1819	1992	2037
	2167	2326	2328	2379	2430	2481	2532	2583	2638	2762	3010	3449	3522	3732	3784
	4011	4515													
.MCALL	675	676	677	678	838	923	1121	1157							
.MEYIT	971														
.MLIST	1	671	706	838	916	923	926	1089	1121	1156	1157	1170	1177	1182	1204
	1208	1230	1232	1240	1242	1245	1249	1266	1268	1276	1278	1284	1285	1294	1296
	1303	1305	1308	1310	1318	1320	1324	1326	1329	1330	1339	1341	1348	1350	1353
	1355	1363	1365	1369	1371	1374	1375	1384	1386	1393	1395	1398	1400	1408	1410
	1414	1416	1419	1420	1429	1431	1438	1440	1443	1445	1453	1455	1459	1461	1464
	1465	1474	1476	1483	1485	1488	1490	1498	1500	1504	1506	1509	1510	1519	1521
	1528	1530	1533	1535	1543	1545	1549	1551	1554	1555	1564	1566	1573	1575	1578

	1580	1588	1590	1594	1596	1593	1600	1609	1611	1618	1620	1623	1625	1633	1635
	1639	1641	1644	1645	1654	1656	1663	1668	1670	1678	1680	1680	1694	1696	1699
	1690	1699	1701	1708	1710	1713	1715	1723	1725	1729	1731	1736	1740	1750	1752
	1755	1757	1765	1767	1770	1772	1775	1779	1789	1791	1794	1796	1804	1806	1809
	1811	1820	1830	1848	1850	1855	1857	1859	1869	1892	1894	1898	1900	1902	1906
	1936	1938	1943	1945	1947	1951	1981	1983	1988	1990	1993	2001	2013	2015	2020
	2022	2030	2032	2035	2037	2038	2047	2059	2061	2065	2067	2072	2076	2084	2086
	2090	2092	2098	2100	2103	2105	2107	2111	2118	2120	2125	2127	2133	2137	2158
	2160	2163	2165	2167	2177	2193	2195	2198	2200	2219	2221	2223	2225	2233	2237
	2256	2258	2267	2269	2272	2274	2276	2280	2295	2297	2302	2306	2316	2318	2321
	2323	2329	2338	2368	2370	2373	2375	2380	2389	2419	2421	2424	2426	2431	2440
	2470	2472	2475	2477	2482	2491	2521	2523	2526	2529	2533	2542	2572	2574	2577
	2579	2584	2593	2627	2629	2632	2634	2639	2648	2674	2676	2679	2681	2686	2690
	2722	2724	2727	2729	2732	2736	2748	2750	2753	2755	2763	2767	2794	2796	2798
	2800	2805	2809	2833	2835	2837	2839	2848	2852	2864	2866	2869	2871	2873	2877
	2893	2895	2899	2900	2901	2905	2920	2922	2925	2927	2929	2933	2959	2961	2964
	2966	2968	2972	3001	3003	3006	3008	3012	3016	3053	3055	3059	3061	3086	3088
	3093	3095	3098	3102	3139	3141	3145	3147	3172	3174	3179	3181	3184	3188	3225
	3227	3231	3233	3258	3260	3265	3267	3271	3275	3312	3314	3318	3320	3345	3347
	3352	3354	3357	3361	3401	3403	3407	3409	3437	3439	3444	3446	3450	3469	3486
	3488	3491	3493	3512	3514	3518	3520	3523	3532	3549	3551	3554	3556	3574	3576
	3580	3582	3589	3591	3593	3595	3596	3600	3612	3619	3637	3639	3642	3644	3650
	3652	3655	3657	3690	3696	3701	3718	3724	3743	3756	3762	3768	3773	4061	4126
	4283	4515	4523	4524	4525	4526	4527	4528	4529	4530	4531	4532	4533	4534	
.PAGE	879	972	1244	1230	1284	1328	1329	1373	1374	1418	1419	1463	1464	1508	1509
.RADIX	1553	1554	1598	1599	1643	1644	1688	1689	1733						
	1284	1296	1329	1341	1374	1386	1419	1431	1464	1476	1509	1521	1554	1566	1599
	1611	1644	1656	1689	1701	1733									
.REM	1														
.REPT	706	2848													
.SBTTL	700	728	846	857	879	923	972	1100	1152	1157	1204	1245	1285	1330	1375
	1420	1465	1510	1555	1600	1645	1690	1736	1775	1815	1816	1817	1820	1859	1902
	1947	1993	2038	2072	2107	2129	2130	2131	2133	2167	2233	2276	2302	2329	2380
	2431	2482	2533	2584	2639	2686	2732	2758	2759	2760	2763	2905	2844	2845	2846
	2848	2873	2901	2929	2968	3012	3098	3184	3271	3357	3450	3523	3596	3734	3786
	3827	3868	3905	3906	3907	3909	3986	4012	4069	4116	4181	4320	4399	4456	4499
	4515														
.TITLE	687														
.WORD	706	707	708	710	854	873	874	875	876	877	878	887	890	891	892
	893	896	897	898	899	900	901	902	905	906	907	928	929	930	931
	932	933	934	935	939	940	941	954	958	961	964	965	966	967	968
	969	1069	1070	1071	1072	1073	1074	1075	1076	1077	1078	1079	1080	1081	1082
	1083	1084	1085	1086	3748	3751	3780	3985	4095	4100	4349	4396	4431	4491	4604
	4606	4608	4609	4611	4613	4615									

ERRORS DETECTED: 0  
DEFAULT GLOBALS GENERATED: 0

\* DVKWA.SEQ/SOL/CRF/PAGNUM/NL: TOC=DVKWA.SML, DVKWA.P11  
RUN-TIME: 50 65 8 SECONDS  
RUN-TIME RATIO: 285/125=2.2  
CORE USED: 35K (69 PAGES)

