KWV11A

DIAGNOSTIC

MD-11-DVKWA-A

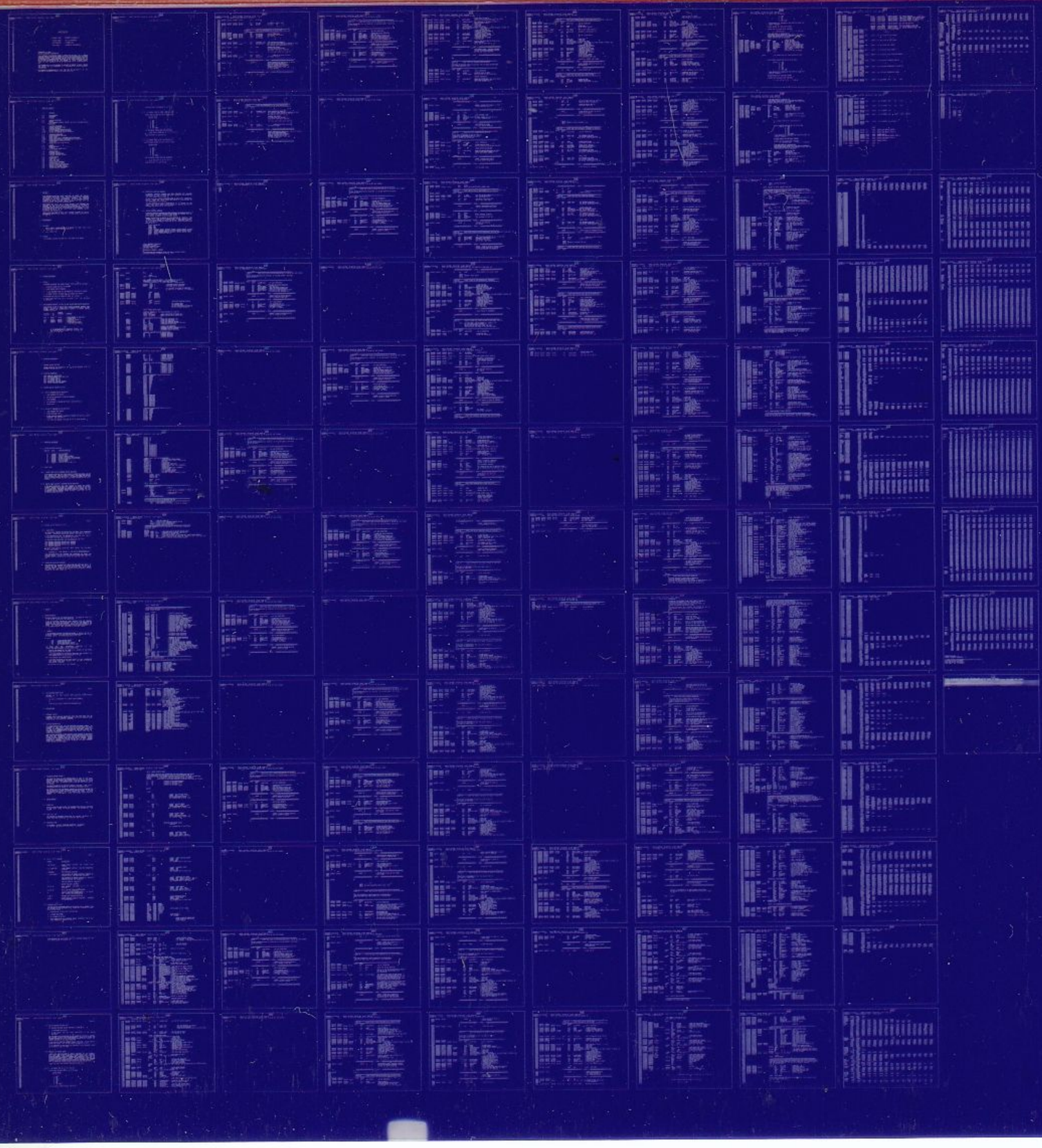
EP-DVKWA-A-DL-A

FICHE 1 OF 1

NOV 1976

digital

MADE IN U.S.A.



```
                        .REM    x
```

IDENTIFICATION

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

TABLE OF CONTENTS

1.0 ABSTRACT

THIS PROGRAM ALLOWS THE USER CHECK-OUT OR DEBUG THE KWV11A, 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 KWV11 COULD INJECT SIGNALS ON ST2, ST1, OR SLAVE IN INPUTS, IT MUST BE DISCONNECTED.

EVEN THOUGH THE KWV11 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. KWV11 UNDER TEST.

2.2 STORAGE

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

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 KWV11 WITH A STANDARD ADDRESS AND VECTOR. IF ANY OF THESE ARE DIFFERENT ON THE KW11K 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.

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.

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.

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 KWV11'S TO EXERCISE, ONE PASS WILL EXERCISE ALL KWV11'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 0000000000001011"

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 KWV11 IT DETECTS ON THE SYSTEM. TO INHIBIT THIS FEATURE, SET BIT 15 OF LOCATION "SENVM". ALSO, TO TEST AN INDIVIDUAL KWV11 IN A GROUP, SET THIS BIT AND REFER TO SECTION 3.2 FOR CHANGING THE BASE ADDRESS OF THE KWV11 UNDER TEST.

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(10) WE GO TO LOCATION "SERRTB" AND LOOK THROUGH THE ERROR POINTER TABLE UNTIL WE FOUND ITEM 5. THE INFORMATION WOULD LOOK LIKE:

```
;ITEM 5

        EMS     ;CLOCK SR DATA ERROR
        DMS     ;ERRPC ASR WAS 5/B
        DTS     ;$ERRPC,ASR,$BDDAT,$GDDAT
        DFO     ;ALL NUMBERS ARE IN OCTAL FORM
```

TO FIND OUT THE INFORMATION SPECIFIED BY DTS ($ERRPC,BSR,$BDADR,$BDADR) FOLLOW THESE STEPS:

1. LOOK UP THE ADDRESS OF THE LABEL (I.E., $ERRPC) 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.

6.2 NON-STANDARD ERROR HALTS

ANY HALT IN THE TRAP CATCHER AREA LOCATIONS 000000-001000, INDICATES:

1. THE KWV11 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.

7.3 POSSIBLE PROGRAM "BOMBS"

THE FIRST TWO TESTS OF THIS PROGRAM CHECK TO SEE IF THE KWV11 RESPONDS TO THE ADDRESS THE PROGRAM THINKS ITS AT. IF THE KWV11 DOES NOT RESPOND, A BUS ERROR OCCURS. ALSO BUS ERRORS CAN OCCUR DURING THE TIME THE PROGRAM SIZES TO SEE HOW MANY KWV11 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 REWITE 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

8.4 LSI-11 "ODT" COMMANDS

FORMAT	DESCRIPTION
<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 KD11F (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).

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

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 ↑G (CONTROL AND LETTER G KEYS TYPED SIMULTANEOUSLY). WHEN ↑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 ↑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 TYPEOUT OPTIONS MAY BE USED.

LOGIC TEST MUST HAVE ALREADY BEEN RUN ON THE KWV11.

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.

589 J1-SS (ST2 OUT) TO J1-VV (ST1 IN)

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	ON
5	ON
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.

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 ALLOTED 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.

```
                        %
                        .NLIST  MC,MD,CMD
                        .LIST   ME
                        .ENABL  ABS
                        .ENABL  AMA
                        .MCALL  .HEADER,.SETUP,.SETTRAP,.TRMTRP,.$TRAP,.$RDOCT,.$TYPBIN
                        .MCALL  TYPOCS,.$POWER,.$CATCH,.$TYPOCT,.EQUAT,.$CMTAG,.SWRHI
                        .MCALL  .SEOP,.SERROR,.SERRTYP,.$TYPDEC,.$SCOPE,.$READ,.$TYPE
                        .MCALL  .SACT11,.SAPTHDR,.SAPTYPE
167400                  $SWR=   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 PDP-11 MAINDEC SYSMAC
                ;*PACKAGE (MAINDEC-11-DZQAC-CO),MAR 21, 1976.
```

```
                                ;*
000001                          $TN=1

                                        .SWRH:
000000                          .SBTTL  TRAP CATCHER

000000                                  .=0
                                ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
                                ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
                                ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
000174                                  .=174
000174  000000                 DISPREG: .WORD  0               ;;SOFTWARE DISPLAY REGISTER
000176  000000                 SWREG:   .WORD  0               ;;SOFTWARE SWITCH REGISTER
000100                                  .=100
000100  000102  000002                  .WORD   102,2           ;IF "B EVENT" ON Q-BUS IS CONNECTED
                                                                ;WE NEED WAY OF IGNORING ITS INTERRUPTS.

000200                                  .=200
000200  000137  001472                  JMP     @#START
000204  000137  002070                  JMP     @#RSTART
000210  000137  013322                  JMP     @#IOTST1
000214  000137  013400                  JMP     @#IOTST2
000220  000137  013446                  JMP     @#IOTST3

000230                                  .=230
000230  000137  001456                  JMP     @#WSTART                ;WESTFIELD STARTING ADDRESS
000240                                  .=240
000240  000137  001442                  JMP     @#TSTSTR                ;ALL TESTER TESTS
                                                                        ;IF STARTED HERE.
                                                                        ;ALLOWS PRODUCTION TO EXERCISE
                                                                        ;UP TO 16 CLOCKS.NORMAL=4.

                                .SBTTL  BASIC DEFINITIONS

                                ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
001100                          STACK=  1100
                                .EQUIV  EMT,ERROR               ;;BASIC DEFINITION OF ERROR CALL
                                .EQUIV  IOT,SCOPE               ;;BASIC DEFINITION OF SCOPE CALL

                                ;*MISCELLANEOUS DEFINITIONS
000011                          HT=     11                      ;;CODE FOR HORIZONTAL TAB
000012                          LF=     12                      ;;CODE FOR LINE FEED
000015                          CR=     15                      ;;CODE FOR CARRIAGE RETURN
000200                          CRLF=   200                     ;;CODE FOR CARRIAGE RETURN-LINE FEED
177776                          PS=     177776                  ;;PROCESSOR STATUS WORD
                                .EQUIV  PS,PSW
177774                          STKLMT= 177774                  ;;STACK LIMIT REGISTER
177772                          PIRQ=   177772                  ;;PROGRAM INTERRUPT REQUEST REGISTER
177570                          DSWR=   177570                  ;;HARDWARE SWITCH REGISTER
177570                          DDISP=  177570                  ;;HARDWARE DISPLAY REGISTER

                                ;*GENERAL PURPOSE REGISTER DEFINITIONS
000000                          R0=     %0                      ;;GENERAL REGISTER
000001                          R1=     %1                      ;;GENERAL REGISTER
000002                          R2=     %2                      ;;GENERAL REGISTER
000003                          R3=     %3                      ;;GENERAL REGISTER
```

```
                        R4=     %4              ;;GENERAL REGISTER
                        R5=     %5              ;;GENERAL REGISTER
                        R6=     %6              ;;GENERAL REGISTER
                        R7=     %7              ;;GENERAL REGISTER
                        .EQUIV  R6,SP           ;;STACK POINTER
                        .EQUIV  R7,PC           ;;PROGRAM COUNTER

                        ;*PRIORITY LEVEL DEFINITIONS
                        PR0=    0               ;;PRIORITY LEVEL 0
                        PR1=    40              ;;PRIORITY LEVEL 1
                        PR2=    100             ;;PRIORITY LEVEL 2
                        PR3=    140             ;;PRIORITY LEVEL 3
                        PR4=    200             ;;PRIORITY LEVEL 4
                        PR5=    240             ;;PRIORITY LEVEL 5
                        PR6=    300             ;;PRIORITY LEVEL 6
                        PR7=    340             ;;PRIORITY LEVEL 7

                        ;*"SWITCH REGISTER" SWITCH DEFINITIONS
                        SW15=   100000
                        SW14=   40000
                        SW13=   20000
                        SW12=   10000
                        SW11=   4000
                        SW10=   2000
                        SW09=   1000
                        SW08=   400
                        SW07=   200
                        SW06=   100
                        SW05=   40
                        SW04=   20
                        SW03=   10
                        SW02=   4
                        SW01=   2
                        SW00=   1
                        .EQUIV  SW09,SW9
                        .EQUIV  SW08,SW8
                        .EQUIV  SW07,SW7
                        .EQUIV  SW06,SW6
                        .EQUIV  SW05,SW5
                        .EQUIV  SW04,SW4
                        .EQUIV  SW03,SW3
                        .EQUIV  SW02,SW2
                        .EQUIV  SW01,SW1
                        .EQUIV  SW00,SW0

                        ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
                        BIT15=  100000
                        BIT14=  40000
                        BIT13=  20000
                        BIT12=  10000
                        BIT11=  4000
                        BIT10=  2000
                        BIT09=  1000
                        BIT08=  400
                        BIT07=  200
                        BIT06=  100
```

```
                        000040          BIT05=  40
                        000020          BIT04=  20
                        000010          BIT03=  10
                        000004          BIT02=  4
                        000002          BIT01=  2
                        000001          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
                        000004          ERRVEC= 4               ;;TIME OUT AND OT ER ERRORS
                        000010          RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
                        000014          TBITVEC=14              ;;"T" BIT
                        000014          TRTVEC= 14              ;;TRACE TRAP
                        000014          BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
                        000020          IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
                        000024          PWRVEC= 24              ;;POWER FAIL
                        000030          EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
                        000034          TRAPVEC=34              ;;"TRAP" TRAP
                        000060          TKVEC=  60              ;;TTY KEYBOARD VECTOR
                        000064          TPVEC=  64              ;;TTY PRINTER VECTOR
                        000240          PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR

                        170420                  ABASE=  170420
                        000440                  AVECTI= 440
                        000200                  APRIOR= 200

                        167400                  $SWR=   167400
                        000001                  $TN=    1

                                        .SBTTL  ACT11 HOOKS

                                        ;;**************************************************************
                                        ;HOOKS REQUIRED BY ACT11
                        000244                  $SVPC=.                 ;SAVE PC
                        000046                  .=46
000046  013302                          $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
                        000052                  .=52
000052  000000                          .WORD   0               ;;2)SET LOC.52 TO ZERO
                        000244                  .=$SVPC                 ;; RESTORE PC
                        001000                  .=1000
                                        .SBTTL  APT PARAMETER BLOCK

                                        ;;**************************************************************
                                        ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
                                        ;;**************************************************************
                        001000                  .$X=.   ;;SAVE CURRENT LOCATION
                        000024                  .=24    ;;SET POWER FAIL TO POINT TO START OF PROGRAM
```

```
                                                200     ;;FOR APT START UP
                                                .=44    ;;POINT TO APT INDIRECT ADDRESS PNTR.
                                                $APTHDR ;;POINT TO APT HEADER BLOCK
                                                .=.$X   ;;RESET LOCATION COUNTER
                                        ;;**************************************************************
                                        ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
                                        ;INTERFACE SPEC.

001000                                  $APTHD:
001000  000000                          $HIBTS: .WORD   0       ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
001002  001174                          $MBADR: .WORD   $MAIL   ;;ADDRESS OF APT MAILBOX (BITS 0-15)
001004  000002                          $TSTM:  .WORD   2       ;;RUN TIM OF LONGEST TEST
001006  000170                          $PASTM: .WORD   120.    ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
001010  000170                          $UNITM: .WORD   120.    ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
001012  000031                                  .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
```

```
                                .SBTTL  COMMON TAGS

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

                        001100          .=1100
001100                  $CMTAG:                         ;;START OF COMMON TAGS
001100  000000                  .WORD   0
001102  000                     $TSTNM: .BYTE   0       ;;CONTAINS THE TEST NUMBER
001103  000                     $ERFLG: .BYTE   0       ;;CONTAINS ERROR FLAG
001104  000000                  $ICNT:  .WORD   0       ;;CONTAINS SUBTEST ITERATION COUNT
001106  000000                  $LPADR: .WORD   0       ;;CONTAINS SCOPE LOOP ADDRESS
001110  000000                  $LPERR: .WORD   0       ;;CONTAINS SCOPE RETURN FOR ERRORS
001112  000000                  $ERTTL: .WORD   0       ;;CONTAINS TOTAL ERRORS DETECTED
001114  000                     $ITEMB: .BYTE   0       ;;CONTAINS ITEM CONTROL BYTE
001115  001                     $ERMAX: .BYTE   1       ;;CONTAINS MAX. ERRORS PER TEST
001116  000000                  $ERRPC: .WORD   0       ;;CONTAINS PC OF LAST ERROR INSTRUCTION
001120  000000                  $GDADR: .WORD   0       ;;CONTAINS ADDRESS OF 'GOOD' DATA
001122  000000                  $BDADR: .WORD   0       ;;CONTAINS ADDRESS OF 'BAD' DATA
001124  000000                  $GDDAT: .WORD   0       ;;CONTAINS 'GOOD' DATA
001126  000000                  $BDDAT: .WORD   0       ;;CONTAINS 'BAD' DATA
001130  000000                          .WORD   0       ;;RESERVED--NOT TO BE USED
001132  000000                          .WORD   0
001134  000                     $AUTOB: .BYTE   0       ;;AUTOMATIC MODE INDICATOR
001135  000                     $INTAG: .BYTE   0       ;;INTERRUPT MODE INDICATOR
001136  000000                          .WORD   0
001140  177570                  SWR:    .WORD   DSWR            ;;ADDRESS OF SWITCH REGISTER
001142  177570                  DISPLAY: .WORD  DDISP           ;;ADDRESS OF DISPLAY REGISTER
001144  177560                  $TKS:   177560                  ;;TTY KBD STATUS
001146  177562                  $TKB:   177562                  ;;TTY KBD BUFFER
001150  177564                  $TPS:   177564                  ;;TTY PRINTER STATUS REG. ADDRESS
001152  177566                  $TPB:   177566                  ;;TTY PRINTER BUFFER REG. ADDRESS
001154  000                     $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER FOR FILLS
001155  002                     $FILLS: .BYTE   2               ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
001156  012                     $FILLC: .BYTE   12              ;;INSERT FILL CHARS. AFTER A "LINE FEED"
001157  000                     $TPFLG: .BYTE   0               ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
001160  000000                  $TIMES: 0                       ;;MAX. NUMBER OF ITERATIONS
001162  000000                  $ESCAPE:0                       ;;ESCAPE ON ERROR ADDRESS
001164  177607  000377          $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
001170  077                     $QUES:  .ASCII  /?/             ;;QUESTION MARK
001171  015                     $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
001172  000012                  $LF:    .ASCIZ  <12>            ;;LINE FEED
                                ;;**********************************************************************
                                .SBTTL  APT MAILBOX-ETABLE

                                ;;**********************************************************************
                                .EVEN
001174                          $MAIL:                  ;;APT MAILBOX
001174  000000                  $MSGTY: .WORD   AMSGTY   ;;MESSAGE TYPE CODE
001176  000000                  $FATAL: .WORD   AFATAL   ;;FATAL ERROR NUMBER
001200  000000                  $TESTN: .WORD   ATESTN   ;;TEST NUMBER
001202  000000                  $PASS:  .WORD   APASS    ;;PASS COUNT
001204  000000                  $DEVCT: .WORD   ADEVCT   ;;DEVICE COUNT
001206  000000                  $UNIT:  .WORD   AUNIT    ;;I/O UNIT NUMBER
001210  000000                  $MSGAD: .WORD   AMSGAD   ;;MESSAGE ADDRESS
```

```
  935  001212  000000                $MSGLG: .WORD   AMSGLG  ;;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              $USWR:  .WORD   AUSWR   ;;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#2BUS PRIORITY#2
  967  001250  170420              $BASE:  .WORD   ABASE   ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
  968  001252  000000              $DEVM:  .WORD   ADEVM   ;;DEVICE MAP
  969  001254  000000              $CDW1:  .WORD   ACDW1   ;;CONTROLLER DESCRIPTION WORD#1
  970  001256                      $ETEND:
  971                              .MEXIT
```

```
                                .SBTTL  ERROR POINTER TABLE

                                ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
                                ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
                                ;*LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
                                ;*NOTE1:        IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
                                ;*NOTE2:        EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:

                                ;*      EM              ;;POINTS TO THE ERROR MESSAGE
                                ;*      DH              ;;POINTS TO THE DATA HEADER
                                ;*      DT              ;;POINTS TO THE DATA
                                ;*      DF              ;;POINTS TO THE DATA FORMAT

001256                          $ERRTB:

                                ;ITEM   1

001256  016364                          EM1                     ;CLOCK   SR FUNCTION ERROR
001260  016716                          DH1                     ;ERRPC   ASR     WAS     S/B
001262  017114                          DT1                     ;$ERRPC,ASR,$BDDAT,$GDDAT
001264  017202                          DF0                     ;ALL NUMBERS ARE IN OCTAL FORM

                                ;ITEM   2

001266  016416                          EM2                     ;CLOCK   SR DATA ERROR
001270  016716                          DH1                     ;ERRPC   ASR     WAS     S/B
001272  017114                          DT1                     ;$ERRPC,ASR,$BDDAT,$GDDAT
001274  017202                          DF0                     ;ALL NUMBERS ARE IN OCTAL FORM

                                ;ITEM   3

001276  016444                          EM3                     ;CLOCK  BR DATA ERROR
001300  016742                          DH3                     ;ERRPC   ABR     WAS
001302  017126                          DT3                     ;$ERRPC,ABR,$BDDAT,$GDDAT
001304  017202                          DF0                     ;ALL NUMBERS ARE IN OCTAL FORM

                                ;ITEM   4

001306  000000                          0               ;RESERVED FOR FUTURE ERROR.
001310  000000                          0                       ;RESERVED.
001312  000000                          0
001314  017202                          DF0                     ;ALL NUMBERS ARE IN OCTAL FORM

                                ;ITEM   5

001316  016472                          EM5                     ;CLOCK   COUNT REG ERROR
001320  016766                          DH4                     ;ERRPC   ACR   WAS   S/B
001322  017114                          DT1                     ;$ERRPC, ACR, $BDDAT, $GDDAT
001324  017202                          DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
```

```
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                            DF0                     ;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                    ;$ERRPC, ASR
1042  001344  017202                            DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
1043
1044
1045                                    ;ITEM   10
1046
1047  001346  016630                            EM20                    ;CLOCK   REPEATABILITY ERROR
1048  001350  017027                            DH20                    ;ERROR   ASR   2ND CNT 1ST CNT  3RD CNT
1049  001352  017146                            DT20                    ;$ERRPC, ASR, $BDDAT, $GDDAT, $TMP0
1050  001354  017202                            DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
1051
1052
1053                                    ;ITEM   11
1054
1055  001356  016515                            EM11                    ;CLOCK   COUNT ERROR
1056  001360  016766                            DH4                     ;ERRPC   ASR   WAS   S/B
1057  001362  017162                            DT22                    ;$ERRPC, ASR, $BDDAT, $TMP0
1058  001364  017202                            DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
1059
1060
1061                                    ;ITEM   12
1062
1063  001366  016665                            EM26                    ;CLOCK ADDRESSING ERROR
1064  001370  017070                            DH26                    ;ERRPC   CLOCK ADDR.
1065  001372  017174                            DT26                    ;$ERRPC,$TMP0
1066  001374  017202                            DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
1067
1068
1069  001376  170420            ASR:    .WORD   ABASE
1070  001400  170422            APR:    .WORD   ABASE+2
1071  001402  000440            VECT1:  .WORD   AVECT1
1072  001404  000442            VECTP:  .WORD   AVECT1+2
1073  001406  000444            VECT2:  .WORD   AVECT1+4        ;VECTOR ADDR. OF ST2 INTRS.
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            $TMP0:  .WORD   0               ;TEMP STORAGE.
1079  001422  000000            $TMP1:  .WORD   0               ;TMP STORAGE.
1080  001424  000000            $TMP3:  .WORD   0
1081  001426  000000            ROTATE: .WORD   0                       ;POINT TO DEVICE UNDER TEST.
1082  001430  000000            UTEST:  .WORD   0                       ;KEEPS TRACK OF GOOD UNITS.
1083  001432  000000            ERCNT:  .WORD   0                       ;COUNTS ERRORS.
```

```
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
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          001456                  WSTART=.
1093  001456  012737  000020  001436          MOV     #16.,TSTCNT     ;TEST UP TO 16 UNITS.
1094  001464  005037  001440                  CLR     EXS
1095  001470  000405                          BR      1$
1096          001472                  START=.
1097  001472  012737  000004  001436          MOV     #4,TSTCNT       ;TEST UP TO FOUR UNITS.
1098  001500  005037  001440                  CLR     EXS
1099  001504                          1$:
1100                                  .SBTTL  INITIALIZE THE COMMON TAGS
1101                                  ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
1102  001504  012706  001100                  MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
1103  001510  005026                          CLR     (R6)+           ;;CLEAR MEMORY LOCATION
1104  001512  022706  001140                  CMP     #SWR,R6 ;;DONE?
1105  001516  001374                          BNE     .-6             ;;LOOP BACK IF NO
1106  001520  012706  001100                  MOV     #STACK,SP       ;;SETUP THE STACK POINTER
1107                                  ;;INITIALIZE A FEW VECTORS
1108  001524  012737  014354  000020          MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
1109  001532  012737  000340  000022          MOV     #340,@#IOTVEC+2 ;;LEVEL 7
1110  001540  012737  014012  000030          MOV     #SERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
1111  001546  012737  000340  000032          MOV     #340,@#EMTVEC+2 ;;LEVEL 7
1112  001554  012737  016320  000034          MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
1113  001562  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
1114  001570  012737  016142  000024          MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
1115  001576  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;LEVEL 7
1116  001604  005037  001160                  CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
1117  001610  005037  001162                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
1118  001614  112737  000001  001115          MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
1119  001622  012737  001622  001106          MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
1120  001630  012737  001630  001110          MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
1121                                  ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
1122                                  ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
1123  001636  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
1124  001642  012737  001676  000004          MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
1125  001650  012737  177570  001140          MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
1126  001656  012737  177570  001142          MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
1127  001664  022777  177777  177246          CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
1128  001672  001012                          BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
1129                                                                  ;;AND  THE HARDWARE SWR IS NOT = -1
1130  001674  000403                          BR      65$             ;;BRANCH IF NO TIMEOUT
1131  001676  012716  001704          64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
1132  001702  000002                          RTI
1133  001704  012737  000176  001140  65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
1134  001712  012737  000174  001142          MOV     #DISPREG,DISPLAY
1135  001720  012637  000004          66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR
1136
1137  001724  005037  001202                  CLR     $PASS           ;;CLEAR PASS COUNT
1138  001730  132737  000200  001215          BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
1139  001736  001403                          BEQ     67$             ;;YES,USE NON-APT SWITCH
```

```
1140  001740  012737  001216  001140          MOV     #SSWREG,SWR     ;;NO,USE APT SWITCH REGISTER
1141  001746                          67$:
1142
1143
1144  001746  012746  000340                  MOV     #340,-(SP)              ;SET CPU PRIORITY ON RETERN.
1145  001752  012746  001760                  MOV     #68$,-(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:                           ;COME HERE FOR NEXT UNIT,OR END PASS.
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.
```

```
;196  002324  062737  000002  001410          ADD     #2,VECT2P
1197
1198
1199  002332  013737  001376  001400          MOV     ASR,ABR         ;FIX ADDR OF PRESET REG=
1200  002340  062737  000002  001402          ADD     #2,ABR          ;CSR + 2
1201
1202
1203
1204                                  ;;**********************************************************************
1205                                  ;*TEST 1         *TEST THE ADDRESSABILITY OF CLOCK CSR
1206                                  ;;**********************************************************************
1207  002346  000240                  TST1:   NOP
1208  002350  012737  000050  001160          MOV     #50,$TIMES      ;;DO 50 ITERATIONS
1209  002356  012737  002400  001106          MOV     #1$,$LPADR      ;;SET SCOPE LOOP ADDRESS
1210  002364  112737  000001  001102          MOVB    #1,$TSTNM       ;SET TEST #1.
1211  002372  012737  002400  001110          MOV     #1$,$LPERR
1212
1213
1214
1215  002400  013746  000004          1$:     MOV     @#ERRVEC,-(SP)  ;SAVE CONTENTS OF ADDRS 6.
1216  002404  012737  002420  000004          MOV     #2$,@#ERRVEC    ;SET TIME-OUT TRAP VECTOR TO HANDLER IN CASE.
1217                                                                  ;WE TIME-OUT WHEN ADDRESSING THE KW11.
1218
1219  002412  005777  176760                  TST     @ASR            ;ADDRESS THE CLOCK!
1220                                                                  ;IF CLOCK DOES NOT RETURN
1221                                                                  ;"BUS SSYN" THEN WE'LL TIME-OUT
1222
1223  002416  000406                          BR      3$              ;THE CLOCK WAS THERE! EXIT SUB-TEST.
1224
1225  002420                          2$:
1226  002420  062706  000004                  ADD     #4,SP                   ;/ADD #4 TO STACK POINTER.
1227  002424  013737  001376  001420          MOV     ASR,$TMP0       ;FOR ERROR TYPEOUT.
1228
1229

                                  ;;SSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSS

1232  002432  104012                          ERROR   12              ;REPORT ERROR=CLOCK CSR FAILED TO RETURN
1233                                                                  ;"BUS SSYN" WHEN ADDRESSED.
1234                                                                  ;NOTE: IF PROGRAM HAS INCORRECT
1235                                                                  ;ADDRESS THEN WE MIG NOT BE
1236                                                                  ;TALKING TO THE CLOCK. MAKE SURE
1237                                                                  ;OF CLOCK ADDRESS.
1238
1239

                                  ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSS

1242  002434  012637  000004          3$:     MOV     (SP)+,@#ERRVEC
1243
```

```
                                ;;*****************************************************************
                                ;*TEST 2        *TEST THE ADDRESSABILITY OF CLOCK BUFFER REG.
                                ;;*****************************************************************
002440  000004                  TST2:   SCOPE

002442  013746  000004          1$:     MOV     @#ERRVEC,-(SP)  ;SAVE CONTENTS OF ADDRS 6.
002446  012737  002462  000004          MOV     #2$,@#ERRVEC    ;SET TIME-OUT TRAP VECTOR TO HANDLER IN CASE.
                                                                ;WE TIME-OUT WHEN ADDRESSING THE KW11.
002454  005777  176720                  TST     @ABR            ;ADDRESS THE CLOCK!
                                                                ;IF CLOCK DOES NOT RETURN
                                                                ;"BUS SSYN" THEN WE'LL TIME-OUT
002460  000406                          BR      3$              ;THE CLOCK WAS THERE! EXIT SUB-TEST.

002462                          2$:
002462  062706  000004                  ADD     #4,SP                   ;/ADD #4 TO STACK POINTER.
002466  013737  001400  001420          MOV     ABR,$TMPO       ;FOR ERROR TYPEOUT.

                                ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSS

002474  104012                          ERROR   12              ;REPORT ERROR=CLOCK BUFFER REG. FAILED TO RETURN
                                                                ;"BUS SSYN" WHEN ADDRESSED.
                                                                ;NOTE: IF PROGRAM HAS INCORRECT
                                                                ;ADDRESS THEN WE MIG NOT BE
                                                                ;TALKING TO THE CLOCK. MAKE SURE
                                                                ;OF CLOCK ADDRESS.

                                ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSS

002476  012637  000004          3$:     MOV     (SP)+,@#ERRVEC
```

```
                        *TEST THE ADDRESSABILITY OF CLOCK BUFFER REG.

1280
1281
1282
1283
```

```
                                        ;/*
                                ;;**********************************************************************
                                ;*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
                                ;*

                                ;;**********************************************************************
002502  000004                  TST3:   SCOPE
002504  012737  000100  001160          MOV     #100,$TIMES     ;;DO 100 ITERATIONS

002512  005077  176660          CLR     @#SR            ;/CLEAR THE STATUS REGISTEF
002516  052777  040000  176652  BIS     #BIT14,@#SR     ;/SET BIT 14.
002524  012737  040000  001124  MOV     #BIT14,$GDDAT   ;/SET FOR ERROR TYPEOUT S/B.
002532  017737  176640  001126  MOV     @#SR,$BDDAT     ;/READ THE STATUS REGISTER.
002540  023737  001124  001126  CMP     $GDDAT,$BDDAT   ;/DID BIT 14 AND ONLY BIT 14 SET?
002546  001402                  BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.

                                ;;SSSSSSSSSSSSSSSSSSSSSSSS>>> ERROR <<<SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

002550  104002                  ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER
                                                        ;/BIT 14 FAILED TO BIT SET.

                                ;;SSSSSSSSSSSSSSSSSSSSSSSS>>> ERROR <<<SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

002552  000412                  BR      2$              ;/BR TO END SUBTEST.

002554  042777  040000  176614  1$:     BIC     #BIT14,@#SR     ;/TRY CLEARING BIT 14.
002562  005037  001124          CLR     $GDDAT          ;/CLEAR S/S FOR TYPEOUT IF ANY.
002566  017737  176604  001126  MOV     @#SR,$BDDAT     ;/NOW READ IT BACK.
002574  001401                  BEQ     2$              ;/IF ZERO - NO ERROR!

                                ;;SSSSSSSSSSSSSSSSSSSSSSSS>>> ERROR <<<SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

002576  104002                  ERROR   2               ;/ERROR - CLOCK A STATUS REGISTER.
                                                        ;/BIT 14 FAILED TO CLEAR.

                                ;;SSSSSSSSSSSSSSSSSSSSSSSS>>> ERROR <<<SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

002600                          2$:
```

```
MAINDEC-11-DVKWA-A      MACY11 27(732)  04-OCT-76  14:57  PAGE 32
DVKWA.P11         T3    *TEST THAT CLOCK A STATUS REGISTER BIT 14 CAN BE SET AND CLEARED

  1328
```

```
1329                                            ;/*
1330                                            ;;**********************************************************************
1331                                            ;*TEST 4        *TEST THAT CLOCK A STATUS REGISTER BIT 13 CAN BE SET AND CLEARED
1332                                            ;*
1333                                            ;*CLOCK STATUS REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
1334                                            ;*F/FS OR GATES
1335                                            ;*
1336
1337                                            ;;**********************************************************************
1338  002600  000004                           TST4:   SCOPE
1339  002602  012737  000100  001160                    MOV     #100,$TIMES     ;;DO 100 ITERATIONS
1340
1341  002610  005077  176562                    CLR     @ASR            ;/CLEAR THE STATUS REGISTER.
1342  002614  052777  020000  176554            BIS     #BIT13,@ASR     ;/SET BIT 13.
1343  002622  012737  020000  001124            MOV     #BIT13,$GDDAT   ;/SET FOR ERROR TYPEOUT S/B.
1344  002630  017737  176542  001126            MOV     @ASR,$BDDAT     ;/READ THE STATUS REGISTER.
1345  002636  023737  001124  001126            CMP     $GDDAT,$BDDAT   ;/DID BIT 13 AND ONLY BIT 13 SET?
1346  002644  001402                            BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
1347
                                                ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

1350  002646  104002                                    ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER
1351                                                                            ;/BIT 13 FAILED TO BIT SET.
1352
                                                ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

1355  002650  000412                                    BR      2$              ;/BR TO END SUBTEST.
1356
1357  002652  042777  020000  176516  1$:       BIC     #BIT13,@ASR     ;/TRY CLEARING BIT 13.
1358  002660  005037  001124                    CLR     $GDDAT          ;/CLEAR S/B FOR TYPEOUT IF ANY.
1359  002664  017737  176506  001126            MOV     @ASR,$BDDAT     ;/NOW READ IT BACK.
1360  002672  001401                            BEQ     2$              ;/IF ZERO - NO ERROR!
1361
1362
                                                ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

1365  002674  104002                                    ERROR   2               ;/ERROR - CLOCK A STATUS REGISTER.
1366                                                                            ;/BIT 13 FAILED TO CLEAR.
1367
1368
                                                ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

1371  002676                                    2$:
1372
```

```
                        *TEST THAT CLOCK A STATUS REGISTER BIT 13 CAN BE SET AND CLEARED

  1373
```

```
MAINDEC-11-DVKWA-A     MACY11 27(732)  04-OCT-76  14:57  PAGE 35
DVKWA.P11       T4     *TEST THAT CLOCK A STATUS REGISTER BIT 13 CAN BE SET AND CLEARED

                                                ;/*
                                        ;;**************************************************************
                                        ;*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                    TST5:   SCOPE
002700  012737  000100  001160            MOV     #100,$TIMES     ;;DO 100 ITERATIONS
002706  005077  176464                    CLR     @#SR            ;/CLEAR THE STATUS REGISTER.
002712  052777  004000  176456            BIS     #BIT11,@#SR     ;/SET BIT 11.
002720  012737  004000  001124            MOV     #BIT11,$GDDAT   ;/SET FOR ERROR TYPEOUT S/B.
002726  017737  176444  001126            MOV     @#SR,$BDDAT     ;/READ THE STATUS REGISTER.
002734  023737  001124  001126            CMP     $GDDAT,$BDDAT   ;/DID BIT 11 AND ONLY BIT 11 SET?
002742  001402                            BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.

                                        ;;SSSSSSSSSSSSSSSSSSSSSS>>> ERROR <<<SSSSSSSSSSSSSSSSSSSSSSSSSSSSS

002744  104002                            ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER
                                                                ;/BIT 11 FAILED TO BIT SET.

                                        ;;SSSS<<SSSSSSSSSSSSSSSSSS>>> ERROR <<<SSSSSSSSSSSSSSSSSSSSSSSSSSSSS

002746  000412                            BR      2$              ;/BR TO END SUBTEST.

002750  042777  004000  176410    1$:     BIC     #BIT11,@#SR     ;/TRY CLEARING BIT 11.
002756  005037  001124                    CLR     $GDDAT          ;/CLEAR S/B FOR TYPEOUT IF ANY.
002762  017737  176410  001126            MOV     @#SR,$BDDAT     ;/NOW READ IT BACK.
002770  001401                            BEQ     2$              ;/IF ZERO - NO ERROR!

                                        ;;SSSSSSSSSSSSSSSSSSSSSS>>> ERROR <<<SSSSSSSSSSSSSSSSSSSSSSSSSSSSS

002772  104002                            ERROR   2               ;/ERROR - CLOCK A STATUS REGISTER.
                                                                ;/BIT 11 FAILED TO CLEAR.

                                        ;;SSSSSSSSSSSSSSSSSSSSSS>>> ERROR <<<SSSSSSSSSSSSSSSSSSSSSSSSSSSSS

002774                          2$:
```

DVKWA.P11 T5 *TEST THAT CLOCK A STATUS REGISTER BIT 11 CAN BE SET AND CLEARED

```
1418
```

```
1419                                                    ;/*
1420                                                    ;;**********************************************************************
1421                                                    ;*TEST 6         *TEST THAT CLOCK A STATUS REGISTER BIT 6 CAN BE SET AND CLEARED
1422                                                    ;*
1423                                                    ;*CLOCK STATUS REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
1424                                                    ;*F/FS OR GATES
1425                                                    ;*
1426
1427                                                    ;;**********************************************************************
1428  002774  000004                                   TST6:   SCOPE
1429  002776  012737  000100  001160                           MOV     #100,$TIMES     ;;DO 100 ITERATIONS
1430
1431  003004  005077  176366                           1$:     CLR     @ASR            ;/CLEAR THE STATUS REGISTER.
1432  003010  052777  000100  176360                           BIS     #BIT6,@ASR      ;/SET BIT 6.
1433  003016  012737  000100  001124                           MOV     #BIT6,$GDDAT    ;/SET FOR ERROR TYPEOUT S/B.
1434  003024  017737  176346  001126                           MOV     @ASR,$BDDAT     ;/READ THE STATUS REGISTER.
1435  003032  023737  001124  001126                           CMP     $GDDAT,$BDDAT   ;/DID BIT 6 AND ONLY BIT 6 SET?
1436  003040  001402                                           BEQ     2$              ;/IF SO-LETS TRY CLEARING IT.
1437
                                                        ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS
1440  003042  104002                                           ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER
1441                                                                                    ;/BIT 6 FAILED TO BIT SET.
1442
                                                        ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS
1445  003044  000412                                           BR      2$              ;/BR TO END SUBTEST.
1446
1447  003046  042777  000100  176322  1$:     BIC     #BIT6,@ASR      ;/TRY CLEARING BIT 6.
1448  003054  005037  001124                           CLR     $GDDAT          ;/CLEAR S/B FOR TYPEOUT IF ANY.
1449  003060  017737  176312  001126                   MOV     @ASR,$BDDAT     ;/NOW READ IT BACK.
1450  003066  001401                                   BEQ     2$              ;/IF ZERO - NO ERROR!
1451
1452
                                                        ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS
1455  003070  104002                                           ERROR   2               ;/ERROR - CLOCK A STATUS REGISTER.
1456                                                                                    ;/BIT 6 FAILED TO CLEAR.
1457
1458
                                                        ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS
1461  003072                                            2$:
1462
```

1463

```
                                        ;/*
                                        ;;**********************************************************************
                                        ;*TEST 7        *TEST THAT CLOCK A STATUS REGISTER BIT 5 CAN BE SET AND CLEARED
                                        ;*
                                        ;*CLOCK STATUS REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
                                        ;*F/FS OR GATES
                                        ;*
                                        ;;**********************************************************************
003072  000004                        TST7:   SCOPE
003074  012737  000100  001160                MOV     #100,$TIMES     ;;DO 100 ITERATIONS

003102  005077  176270                        CLR     @ASR            ;/CLEAR THE STATUS REGISTER.
003106  052777  000040  176262                BIS     #BIT5,@ASR      ;/SET BIT 5.
003114  012737  000040  001124                MOV     #BIT5,$GDDAT    ;/SET FOR ERROR TYPEOUT S/B.
003122  017737  176250  001126                MOV     @ASR,$BDDAT     ;/READ THE STATUS REGISTER.
003130  023737  001124  001126                CMP     $GDDAT,$BDDAT   ;/DID BIT 5 AND ONLY BIT 5 SET?
003136  001402                                BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.

                                        ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

003140  104002                                ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER
                                                                        ;/BIT 5 FAILED TO BIT SET.

                                        ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

003142  000412                                BR      2$              ;/BR TO END SUBTEST.

003144  042777  000040  176224        1$:     BIC     #BIT5,@ASR      ;/TRY CLEARING BIT 5.
003152  005037  001124                        CLR     $GDDAT          ;/CLEAR S/B FOR TYPEOUT IF ANY.
003156  017737  176214  001126                MOV     @ASR,$BDDAT     ;/NOW READ IT BACK.
003164  001401                                BEQ     2$              ;/IF ZERO - NO ERROR!

                                        ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

003166  104002                                ERROR   2               ;/ERROR - CLOCK A STATUS REGISTER.
                                                                        ;/BIT 5 FAILED TO CLEAR.

                                        ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

003170                                  2$:
```

```
                        ;*TEST THAT CLOCK A STATUS REGISTER BIT 5 CAN BE SET AND CLEARED

  1508
```

```
                                        ;/*
                                ;;**********************************************************************
                                ;*TEST 10        *TEST THAT CLOCK A STATUS REGISTER BIT 4 CAN BE SET AND CLEARED
                                ;*
                                ;*CLOCK STATUS REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
                                ;*F/FS OR GATES
                                ;*

                                ;;**********************************************************************
003170  000004                  TST10:  SCOPE
003172  012737  000100  001160          MOV     #100,$TIMES     ;;DO 100 ITERATIONS

003200  005077  176172                  CLR     @#SR            ;/CLEAR THE STATUS REGISTER.
003204  052777  000020  176164          BIS     #BIT4,@#SR      ;/SET BIT 4.
003212  012737  000020  001124          MOV     #BIT4,$GOODAT   ;/SET FOR ERROR TYPEOUT S/B.
003220  017737  176152  001126          MOV     @#SR,$BODAT     ;/READ THE STATUS REGISTER.
003226  023737  001124  001126          CMP     $GOODAT,$BODAT  ;/DID BIT 4 AND ONLY BIT 4 SET?
003234  001402                  BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.

                                ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSS

003236  104002                  ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER
                                                        ;/BIT 4 FAILED TO BIT SET.

                                ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSS

003240  000412                  BR      2$              ;/BR TO END SUBTEST.

003242  042777  000020  176126  1$:     BIC     #BIT4,@#SR      ;/TRY CLEARING BIT 4.
003250  005037  001124                  CLR     $GOODAT         ;/CLEAR S/B FOR TYPEOUT IF ANY.
003254  017737  176116  001126          MOV     @#SR,$BODAT     ;/NOW READ IT BACK.
003262  001401                  BEQ     2$              ;/IF ZERO - NO ERROR!

                                ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSS›SSSSSSSSSSSSSSSSSSSSS

003264  104002                  ERROR   2               ;/ERROR - CLOCK A STATUS REGISTER.
                                                        ;/BIT 4 FAILED TO CLEAR.

                                ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSS

003266                          2$:
```

1553

```
1554                                                    ;/*
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                    1$T11:  SCOPE
1564  003270  012737  000100  001160            MOV     #100,$TIMES     ;;DO 100 ITERATIONS
1565
1566  003276  005077  176074            2$:     CLR     @#SR            ;/CLEAR THE STATUS REGISTER.
1567  003302  052777  000010  176066            BIS     #BIT3,@#SR      ;/SET BIT 3.
1568  003310  012737  000010  001124            MOV     #BIT3,$GDDAT    ;/SET FOR ERROR TYPEOUT S/B.
1569  003316  017737  176054  001126            MOV     @#SR,$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     1$              ;/IF SO-LETS TRY CLEARING IT.
1572
                                        ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

1575  003334  104002                            ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER
1576                                                                    ;/BIT 3 FAILED TO BIT SET.
1577
                                        ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

1580  003336  000412                            BR      2$              ;/BR TO END SUBTEST.
1581
1582  003340  042777  000010  176030  1$:       BIC     #BIT3,@#SR      ;/TRY CLEARING BIT 3.
1583  003346  005037  001124                    CLR     $GDDAT          ;/CLEAR S/B FOR TYPEOUT IF ANY.
1584  003352  017737  176020  001126            MOV     @#SR,$BDDAT     ;/NOW READ IT BACK.
1585  003360  001401                            BEQ     2$              ;/IF ZERO - NO ERROR!
1586
1587
                                        ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

1590  003362  104002                            ERROR   2               ;/ERROR - CLOCK A STATUS REGISTER.
1591                                                                    ;/BIT 3 FAILED TO CLEAR.
1592
1593
                                        ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

1596  003364                            2$:
1597
```

```
1598
```

```
                                        ;/*
                                ;;**********************************************************************
                                ;*TEST 12        *TEST THAT CLOCK A STATUS REGISTER BIT 2 CAN BE SET AND CLEARED
                                ;*
                                ;*CLOCK STATUS REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
                                ;*F/FS OR GATES
                                ;*

                                ;;**********************************************************************
003364  000004                  TST12:  SCOPE
003366  012737  000100  001160          MOV     #100,$TIMES     ;;DO 100 ITERATIONS

003374  005077  175776          CLR     @#ASR           ;/CLEAR THE STATUS REGISTER.
003400  052777  000004  175770  BIS     #BIT2,@#ASR     ;/SET BIT 2.
003406  012737  000004  001124  MOV     #BIT2,$GDDAT    ;/SET FOR ERROR TYPEOUT S/B.
003414  017737  175756  001126  MOV     @#ASR,$BDDAT    ;/READ THE STATUS REGISTER.
003422  023737  001124  001126  CMP     $GDDAT,$BDDAT   ;/DID BIT 2 AND ONLY BIT 2 SET?
003430  001402                  BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.

                                ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

003432  104002                          ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER
                                                                ;/BIT 2 FAILED TO BIT SET.

                                ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

003434  000412                          BR      2$              ;/BR TO END SUBTEST.

003436  042777  000004  175732  1$:     BIC     #BIT2,@#ASR     ;/TRY CLEARING BIT 2.
003444  005037  001124          CLR     $GDDAT          ;/CLEAR S/B FOR TYPEOUT IF ANY.
003450  017737  175722  001126  MOV     @#ASR,$BDDAT    ;/NOW READ IT BACK.
003456  001401                  BEQ     2$              ;/IF ZERO - NO ERROR!

                                ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

003460  104002                          ERROR   2               ;/ERROR - CLOCK A STATUS REGISTER.
                                                                ;/BIT 2 FAILED TO CLEAR.

                                ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

003462                          2$:
```

#TEST THAT CLOCK A STATUS REGISTER BIT 2 CAN BE SET AND CLEARED

1643

```
                                        ;/*
                                ;;**********************************************************************
                                ;*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
                                ;*

                                ;;**********************************************************************
003462  000004                  TST13:  SCOPE
003464  012737  000100  001160          MOV     #100,$TIMES     ;;DO 100 ITERATIONS

003472  005077  175700          CLR     @ASR            ;/CLEAR THE STATUS REGISTER.
003476  052777  000002  175672  BIS     #BIT1,@ASR      ;/SET BIT 1.
003504  012737  000002  001124  MOV     #BIT1,$GDDAT    ;/SET FOR ERROR TYPEOUT S/B.
003512  017737  175660  001126  MOV     @ASR,$BDDAT     ;/READ THE STATUS REGISTER.
003520  023737  001124  001126  CMP     $GDDAT,$BDDAT   ;/DID BIT 1 AND ONLY BIT 1 SET?
003526  001402                  BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.

                                ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

003530  104002                  ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER
                                                        ;/BIT 1 FAILED TO BIT SET.

                                ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

003532  000412                  BR      2$              ;/BR TO END SUBTEST.

003534  042777  000002  175634  1$:     BIC     #BIT1,@ASR      ;/TRY CLEARING BIT 1.
003542  005037  001124          CLR     $GDDAT          ;/CLEAR S/B FOR TYPEOUT IF ANY.
003546  017737  175624  001126  MOV     @ASR,$BDDAT     ;/NOW READ IT BACK.
003554  001401                  BEQ     2$              ;/IF ZERO - NO ERROR!

                                ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

003556  104002                  ERROR   2               ;/ERROR - CLOCK A STATUS REGISTER.
                                                        ;/BIT 1 FAILED TO CLEAR.

                                ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

003560                          2$:
```

```
1688
```

```
1689                                            ;/*
1690                                            ;;**********************************************************************
1691                                            ;*TEST 14       *TEST THAT CLOCK A STATUS REGISTER BIT 0 CAN BE SET AND CLEARED
1692                                            ;*
1693                                            ;*CLOCK STATUS REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
1694                                            ;*F/FS OR GATES
1695                                            ;*
1696
1697                                            ;;**********************************************************************
1698  003560  000004                           TST14:  SCOPE
1699  003562  012737  000100  001160                   MOV     #100,$TIMES     ;;DO 100 ITERATIONS
1700
1701  003570  005077  175602                           CLR     @#SR            ;/CLEAR THE STATUS REGISTER.
1702  003574  052777  000001  175574                   BIS     #BIT0,@#SR      ;/SET BIT 0.
1703  003602  012737  000001  001124                   MOV     #BIT0,$GDDAT    ;/SET FOR ERROR TYPEOUT S/B.
1704  003610  017737  175562  001126                   MOV     @#SR,$BDDAT     ;/READ THE STATUS REGISTER.
1705  003616  023737  001124  001126                   CMP     $GDDAT,$BDDAT   ;/DID BIT 0 AND ONLY BIT 0 SET?
1706  003624  001402                                   BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
1707
                                                ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

1710  003626  104002                                   ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER
1711                                                                            ;/BIT 0 FAILED TO BIT SET.
1712
                                                ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

1715  003630  000412                                   BR      2$              ;/BR TO END SUBTEST.
1716
1717  003632  042777  000001  175536  1$:      BIC     #BIT0,@#SR      ;/TRY CLEARING BIT 0.
1718  003640  005037  001124                   CLR     $GDDAT          ;/CLEAR S/B FOR TYPEOUT IF ANY.
1719  003644  017737  175526  001126           MOV     @#SR,$BDDAT     ;/NOW READ IT BACK.
1720  003652  001401                           BEQ     2$              ;/IF ZERO - NO ERROR!
1721
1722
                                                ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

1725  003654  104002                                   ERROR   2               ;/ERROR - CLOCK A STATUS REGISTER.
1726                                                                            ;/BIT 0 FAILED TO CLEAR.
1727
1728
                                                ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

1731  003656                                    2$:
1732
```

```
1733            000010                          .RADIX  8
1734
1735
1736                                    ;;**************************************************************
1737                                    ;*TEST 15       *TEST THAT PATERN 125252 WILL SET AND CLEAR IN BUFFER REG.
1738                                    ;;**************************************************************
1739  003656  000004                    TST15:  SCOPE
1740
1741  003660  005077  175514                    CLR     @#BR            ;/CLEAR THE BUFFER REG.
1742  003664  012737  125252  001124            MOV     #125252,$GDDAT  ;/RECORD PATTERN: 125252 .
1743  003672  013777  001124  175500            MOV     $GDDAT,@#BR     ;/SET PATTERN IN BUFFER REG.
1744  003700  017737  175474  001126            MOV     @#BR,$BDDAT     ;/READ THE BUFFER REG.
1745
1746  003706  023737  001124  001126            CMP     $GDDAT,$BDDAT   ;/DID THE PATTERN SET OK?
1747  003714  001402                            BEQ     1$              ;/YES-TRY CLEARING IT.
1748
1749
                                        ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

1752  003716  104003                            ERROR   3               ;/ERROR PATTERN 125252 FAILED TO
1753                                                                    ;/SET PROPERLY IN BUFFER REG.
1754
                                        ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

1757  003720  000412                            BR      2$              ;/GOTO SCOPE LOOP.
1758
1759  003722  042777  125252  175450  1$:       BIC     #125252,@#BR    ;/TRY CLEARING PATTERN.
1760  003730  005037  001124                    CLR     $GDDAT          ;/EXPECT ZERO BACK.
1761  003734  017737  175440  001126            MOV     @#BR,$BDDAT     ;/READ BUFFER REG.,WAS IT ZERO?
1762  003742  001401                            BEQ     2$              ;/YES-NEXT TEST.
1763
1764
                                        ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

1767  003744  104003                            ERROR   3               ;/BUFFER REG. COULD NOT BE LOADED
1768                                                                    ;/TO A ZERO.
1769
                                        ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

1772  003746                            2$:
1773
1774
1775                                    ;;**************************************************************
1776                                    ;*TEST 16       *TEST THAT PATERN 052525 WILL SET AND CLEAR IN BUFFER REG.
1777                                    ;;**************************************************************
1778  003746  000004                    TST16:  SCOPE
1779
1780  003750  005077  175424                    CLR     @#BR            ;/CLEAR THE BUFFER REG.
1781  003754  012737  052525  001124            MOV     #052525,$GDDAT  ;/RECORD PATTERN: 052525 .
1782  003762  013777  001124  175410            MOV     $GDDAT,@#BR     ;/SET PATTERN IN BUFFER REG.
1783  003770  017737  175404  001126            MOV     @#BR,$BDDAT     ;/READ THE BUFFER REG.
1784
1785  003776  023737  001124  001126            CMP     $GDDAT,$BDDAT   ;/DID THE PATTERN SET OK?
1786  004004  001402                            BEQ     1$              ;/YES-TRY CLEARING IT.
1787
1788
```

```
                                        ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

1791  004006  104003                    ERROR   3               ;/ERROR PATTERN 052525 FAILED TO
1792                                                            ;/SET PROPERLY IN BUFFER REG.
1793
                                        ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

1796  004010  000412                    BR      2$              ;/GOTO SCOPE LOOP.
1797
1798  004012  042777  052525  175360  1$:     BIC     #052525,@#BR    ;/TRY CLEARING PATTERN.
1799  004020  005037  001124            CLR     $GDDAT          ;/EXPECT ZERO BACK.
1800  004024  017737  175350  001126    MOV     @#BR,$BDDAT     ;/READ BUFFER REG.,WAS IT ZERO?
1801  004032  001401                    BEQ     2$              ;/YES-NEXT TEST.
1802
1803
                                        ;;SSSSSSSSSS"SSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

1806  004034  104003                    ERROR   3               ;/BUFFER REG. COULD NOT BE LOADED
1807                                                            ;/TO A ZERO.
1808
                                        ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

1811  004036                            2$:
1812
1813
1814
1815                                            .SBTTL  *
1816                                            .SBTTL  * PHASE 2 ADVANCED BASIC LOGIC TESTS
1817                                            .SBTTL  *
1818
1819
1820                                    ;;**********************************************************************
1821                                    ;*TEST 17        *TEST   ~ LOW BYTE OPERATION OF CLOCK'S STATUS REGISTER
1822
1823                                    ;*
1824                                    ;*WE CAN SUCCESSFULLY WRITE EVERY BIT IN STATUS REG A
1825                                    ;*NOW LETS CHECK THE BYTE OPERATION OF THIS REGISTER.
1826                                    ;*
1827
1828                                    ;;**********************************************************************
1829  004036  000004                    TST17:  SCOPE
1830  004040  012737  000050  001160            MOV     #50,$TIMES      ;;DO 50 ITERATIONS
1831
1832  004046  005077  175324                    CLR     @#SR            ;MAKE SURE THE STATUS REGISTER IS CLEAR.
1833  004052  112777  127677  175316            MOVB    #127677,@#SR    ;TRY WRITING ALL BITS IN THE
1834                                                                    ;STATUS REGISTER. LOGIC SHOULD PREVENT IT
1835                                                                    ;FROM BEING WRITTEN INTO BECAUSE
1836                                                                    ;WE ARE USING A DATOB INSTRUCTION.
1837
1838  004060  017777  175312  175040      MOV     @#SR,@#SBDDAT   ;NOW EXAMINE THE
1839                                                                    ;STATUS REGISTER.
1840  004066  013737  001126  001124      MOV     $BDDAT,$GDDAT   ;FIX $GDDAT FOR ERROR TYPEOUT IF
1841  004074  105037  001125            CLRB    $GDDAT+1        ;ANY RROR HAS OCCURRED, UPPER BYTE CLEARED.
1842
1843  004100  105737  001127            TSTB    $BDDAT+1        ;ARE ANY BITS IN THE UPPER BYTE
1844                                                                    ;OF THE STATUS REGISTER SET?
```

```
1845  004104  001401                          BEQ     1$              ;BRANCH NEXT TEST IF UPPER BYTE=0.
1846
1847
                                        ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSS

1850  004106  104001                          ERROR   1               ;ERROR - WROTE INTO UPPER BYTE OF
1851                                                                  ;CLOCK'S STATUS WHEN
1852                                                                  ;DOING A DATOB TO THE LOW BYTE.
1853
1854
                                        ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSS

1857  004110                            1$:
1858
1859                                    ;;**************************************************************
1860                                    ;*TEST 20        *TEST THE HIGH BYTE OPERATION OF A'S STATUS REGISTER
1861
1862                                    ;*
1863                                    ;*WE CAN SUCCESSFULLY WRITE EVERY BIT IN STATUS REG A
1864                                    ;*NOW LETS CHECK THE BYTE OPERATION OF THIS REGISTER.
1865                                    ;*
1866
1867                                    ;;**************************************************************
1868  004110  000004                    TST20:  SCOPE
1869  004112  012737  000050  001160            MOV     #50,$TIMES      ;;DO 50 ITERATIONS
1870
1871  004120  005077  175252                    CLR     @ASR            ;CLEAR THE STATUS REGISTER.
1872
1873  004124  005237  001376                    INC     ASR             ;ADD #1 TO THE STATUS REGISTER'S ADDRESS
1874                                                                  ;SO THAT WE WILL BE WRITING INTO
1875                                                                  ;THE HIGH BYTE.
1876
1877  004130  112777  177313  175240            MOVB    #177313,@ASR    ;TRY WRITING ALL BITS IN THE STATUS
1878                                                                  ;REGISTER. LOGIC SHOULD PREVENT THE LOW
1879                                                                  ;BYTE OF THE STATUS REGISTER FROM
1880                                                                  ;BEING WRITTEN INTO BECAUSE WE ARE USING
1881                                                                  ;A DATOB INSTRUCTION WITH ADD SET.
1882
1883  004136  005337  001376                    DEC     ASR             ;FIX ADDRESS OF THE STATUS REGISTER ADDR.
1884                                                                  ;SO WE CAN LOOK AT THE WHOLE WORD.
1885  004142  017737  175230  001126            MOV     @ASR,$BDDAT     ;READ BACK WHAT THE STATUS REG. CONTAINS
1886  004150  013737  001126  001124            MOV     $BDDAT,$GDDAT   ;FIX $GDDAT FOR ERROR TYPEOUT IF AN ERROR
1887  004156  105C37  001124                    CLRB    $GDDAT          ;OCCURRED, LOWER BYTE CLEARED.
1888  004162  105737  001126                    TSTB    $BDDAT          ;IS LOWER BYTE CLEAR?
1889  004166  001401                            BEQ     1$              ;BR IF YES TO NEXT SUBTEST.
1890
1891
                                        ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSS

1894  004170  104001                          ERROR   1               ;ERROR - WROTE INTO LOWER BYTE
1895                                                                  ;OF CLOCKS STATUS REGISTER WHEN
1896                                                                  ;DOING A DATOB TO THE HIGH BYTE.
1897
                                        ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSS

1900  004172                            1$:
```

```
1901
1902                                    ;;*********************************************************************
1903                                    ;*TEST 21       *TEST CLOCK'S COUNT REGISTER WITH 125252 PATTERN
1904                                    ;;*********************************************************************
1905  004172  000004                    TST21:  SCOPE
1906  004174  012737  000100  001160            MOV     #100,$TIMES     ;;DO 100 ITERATIONS
1907
1908  004202  005077  175170            CLR     @ASR            ;SELECT MODE 0.
1909  004206  012777  125252  175164    MOV     #125252,@ABR    ;LOAD THE BUFFER REGISTER WITH
1910                                                            ;PATTERN 125252. IT WILL BE
1911                                                            ;TRANSFERRED TO THE COUNT REGISTER
1912                                                            ;SINCE THIS IS MODE 0.
1913  004214  052777  000001  175154    BIS     #BIT0,@ASR      ;SET GO BIT(ALLOWS BUFFER-COUNT REG XFER).
1914
1915  004222  012737  125252  001124    MOV     #125252,$GODAT  ;SET EXPECTED TO PATTERN IN CASE OF
1916                                                            ;NEED OF ERROR TYPEOUT.
1917  004230  017746  175142            MOV     @ASR,-(6)       ;/SAVE CSR
1918  004234  011637  001424            MOV     (6),$TMP3       ;/GET CSR.
1919  004240  042737  177707  001424    BIC     #177707,$TMP3   ;/SAVE RATE BITS.
1920  004246  052737  004005  001424    BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
1921  004254  013777  001424  175114    MOV     $TMP3,@ASR              ;/LOAD CSR.
1922                                                            ;/THIS MUST BE DONE IN
1923                                                            ;/ORDER TO XFERR COUNTER
1924                                                            ;/TO BUFFER ON ST2.
1925  004262  052777  001000  175106    BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
1926  004270  017737  175104  001126    MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
1927                                                            ;/PREVIOUS COUNTER
1928  004276  012677  175074            MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
1929  004302  005737  001126            TST     $BDDAT          ;/RESTORE CSR
1930
1931  004306  023737  001124  001126    CMP     $GODAT,$BDDAT   ;DID ALL THE BITS AND NO OTHER BITS
1932                                                            ;COME THROUGH?
1933  004314  001401                    BEQ     1$              ;BR IF YES TO NEXT TEST.
1934
1935

                                        ;;SSSSSSSSSSSSSSSSSSSSS>>> ERROR <<<SSSSSSSSSSSSSSSSSSSSSSSSSSSS

1938  004316  104005                    ERROR   5               ;DATA ERROR CLOCK - PATTERN "125252"
1939                                                            ;FAILED TO TRANSFER PROPERLY BETWEEN
1940                                                            ;BUFFER AND COUNT REGISTERS.
1941
1942

                                        ;;SSSSSSSSSSSSSSSSSSSSS>>> ERROR <<<SSSSSSSSSSSSSSSSSSSSSSSSSSSS

1945  004320                            1$:
1946
1947                                    ;;*********************************************************************
1948                                    ;*TEST 22       *TEST CLOCKS COUNTER REGISTER WITH 052525 PATTERN
1949                                    ;;*********************************************************************
1950  004320  000004                    TST22:  SCOPE
1951  004322  012737  000050  0C1160            MOV     #50,$TIMES      ;;DO 50 ITERATIONS
1952
1953  004330  005077  175042            CLR     @ASR            ;SELECT MODE 0.
1954  004334  012777  052525  175036    MOV     #052525,@ABR    ;LOAD THE BUFFER REGISTER WITH
1955                                                            ;PATTERN 052525. IT WILL BE
1956                                                            ;TRANSFERRED TO THE COUNT REGISTER
```

```
                                                                  ;SINCE THIS IS MODE 0.
004342  052777  000001  175026          BIS     #BIT0,@ASR        ;SET B0 BIT(ALLOWS BUFFER-COUNT REG XFER).

004350  012737  052525  001124          MOV     #052525,$GDDAT    ;SET EXPECTED TO PATTERN IN CASE OF
                                                                  ;NEED OF ERROR TYPEOUT.
004356  017746  175014                  MOV     @ASR,-(6)         ;/SAVE CSR
004362  011637  001424                  MOV     (6),$TMP3         ;/GET CSR.
004366  042737  177707  001424          BIC     #177707,$TMP3     ;/SAVE RATE BITS.
004374  052737  004005  001424          BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE.DISABLE INTERNAL OSC
004402  013777  001424  174766          MOV     $TMP3,@ASR          ;/LOAD CSR.
                                                                  ;/THIS MUST BE DONE IN
                                                                  ;/ORDER TO XFERR COUNTER
                                                                  ;/TO BUFFER ON ST2.
004410  052777  001000  174760          BIS     #BIT9,@ASR        ;/GENERATE ON ST2 PULSE
004416  017737  174756  001126          MOV     @ABR,$BDDAT       ;/READ THE PRESET BUFFER,
                                                                  ;/PREVIOUS COUNTER
004424  012677  174746                  MOV     (6)+,@ASR         ;/CONTENTS ARE IN $BDDAT.
004430  005737  001126                  TST     $BDDAT            ;/RESTORE CSR

004434  023737  001124  001126          CMP     $GDDAT,$BDDAT     ;DID ALL THE BITS AND NO OTHER BITS
                                                                  ;COME THROUGH?
004442  001401                          BEQ     1$                ;BR IF YES TO NEXT TEST.


                                ;;$$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

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


                                ;;$$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

004446                          1$:



                                ;;*************************************************************************
                                ;*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.
                                ;*
                                ;;*************************************************************************
004446  000004                  TST23:  SCOPE
004450  012737  000005  001160          MOV     #5,$TIMES         ;;DO 5 ITERATIONS

004456  005037  001124                  CLR     $GDDAT            ;EXPECTED DATA IS ZERO.
004462  012777  177777  174706          MOV     #177777,@ASR      ;SET ALL BITS IN THE STATUS REG.

004470  000005                          RESET                     ;SYSTEM INITIALIZE.

004472  017737  174700  001126          MOV     @ASR,$BDDAT       ;READ THE STATUS REG., ALL BITS SHOULD
                                                                  ;HAVE BEEN CLEARED BY INIT.
004500  001402                          BEQ     1$                ;BR IF YES TO NEXT TEST.
```

```
                                        ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

2015
2016  004502  104002                          ERROR   2               ;ERROR - SYSTEM INIT FAILED TO CLEAR
2017                                                                  ;STATUS REGISTER CLOCK A.
2018
2019
                                        ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

2022  004504  000413                          BR      TST24           ;;
2023  004506                          1$:
2024  004506  005737  001440                  TST     EXS             ;TEST EXTERNAL SIGS?
2025  004512  001410                          BEQ     TST24           ;;
2026  004514  052777  016000  174674          BIS     #BIT11!BIT12!BIT10,@DR2 ;ENABLE ST1,ST2 TO LATCH.
2027  004522  032777  000006  174664          BIT     #6,@DR          ;ST1,ST2, OVERFLOW SET?
2028  004530  001401                          BEQ     TST24           ;;
2029
                                        ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

2032  004532  104006                          ERROR   6               ;INIT FAILED TO CLEAR
2033                                                                  ;ST1,ST2, AND/OR OVERFLOW
2034
                                        ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

2037
2038                                    ;;**************************************************************
2039                                    ;*TEST 24        *TEST THAT INIT CLEARS BUFFER REGISTER
2040                                    ;*
2041                                    ;*THIS TEST IS DESIGNED TO SEE IF "INIT H"
2042                                    ;*CLEARS THE BUFFER REGISTER. WE ALREADY
2043                                    ;*KNOW IT CLEARS THE STATUS REG.
2044                                    ;*
2045                                    ;**************************************************************
2046  004534  000004                    TST24:  SCOPE
2047  004536  012737  000005  001160            MOV     #5,$TIMES       ;;DO 5 ITERATIONS
2048
2049  004544  005037  001124                    CLR     $GDDAT          ;CLEAR EXPECTED DATA.
2050  004550  012777  177777  174622            MOV     #177777,@DBR    ;SET ALL BITS IN THE BUFFER REGISTER.
2051
2052  004556  000005                          RESET                   ;ISSUE SYSTEM INITIALIZE.
2053
2054  004560  017737  174614  001106          MOV     @DBR,$BDDAT     ;READ THE BUFFER REGISTER, ALL BITS
2055                                                                  ;SHOULD HAVE BEEN CLEARED BY INIT.
2056  004566  001401                          BEQ     1$              ;BR IF YES TO NEXT SUBTEST.
2057
2058
                                        ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

2061  004570  104003                          ERROR   3               ;ERROR - SYSTEM INIT FAILED
2062                                                                  ;TO CLEAR BUFFER REGISTER A.
2063
2064
                                        ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

2067  004572                            1$:
2068  004572  005737  001440                  TST     EXS             ;TEST EXTERNAL SIGS?
```

```
                                        BEQ     TST25                   ;;
004600  052777  016000  174610          BIS     #BIT11!BIT12!BIT10,@DR2 ;ENABLE THEM

                        ;;**************************************************************
                        ;*TEST 25       #TEST THE SETTING OF MAINTENANCE ST2 IN CLOCK BIT 15 TO SET
                        ;;**************************************************************
004606  000004          TST25:  SCOPE

004610  012777  000001  174560          MOV     #1,@ASR         ;SET THE GO BIT(ENABLES ST2 TO SET FLG).
004616  052777  001000  174552          BIS     #BIT9,@ASR      ;SET MAINTENANCE ST2.

004624  005777  174546                  TST     @ASR            ;DID BIT15 (ST2 FLAG) SET?
004630  100402                          BMI     1$              ;BR IF YES - NEXT TEST

                        ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

004632  104001                          ERROR   1               ;ERROR - MAINTENANCE ST2 (BIT9)
                                                                ;DID NOT SET BIT15 (ST2 FLAG).

                        ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

004634  000410                          BR      TST26           ;;
004636  005737  001440          1$:     TST     @XS             ;TEST EXTERNAL SIGNALS?
004642  001405                          BEQ     TST26           ;;
004644  032777  000004  174542          BIT     #BIT2,@DR       ;DID EXTERNAL ST2 GET SET?
004652  001001                          BNE     TST26           ;;

                        ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

004654  104006                          ERROR   6               ;ST2 OUT NOT DETECTED
                                                                ;BY TESTOR

                        ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS



                        ;;**************************************************************
                        ;*TEST 26       #TEST THAT BIT00 IN CLOCK STATUS REG. WILL SET WHEN BIT13 AND MAIN. ST2
                        ;;**************************************************************
004656  000004          TST26:  SCOPE

004660  012777  020000  174510          MOV     #BIT13,@ASR     ;SET "ST2 ENB COUNTER" IN CLK  STATUS REG.
004666  052777  001000  174502          BIS     #BIT9,@ASR      ;GENERATE A MAINTENANCE ST2.
004674  032777  000001  174474          BIT     #BIT00,@ASR     ;DID BIT00 (GO) SET?
004702  001001                          BNE     1$              ;BR IF YES - NEXT TEST.

                        ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

004704  104001                          ERROR   1               ;ERROR - BIT00 OF CLOCK'S STATUS REGISTER
                                                                ;FAILED TO SET WHEN BIT13 WAS SET
                                                                ;AND A MAINTENANCE ST2 GENERATED.
```

```
                                        ;;SSSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

2127  004706  005077  174464            1$:     CLR     @#ASR           ;LEAVE SUBTEST WITH CLOCK CLEAR.
2128
2129                                            .SBTTL  *
2130                                            .SBTTL  *PHASE 3 COUNT TESTS
2131                                            .SBTTL  *
2132
2133                                    ;;**************************************************************
2134                                    ;*TEST 27        *TEST TO SEE IF THE COUNTER WILL INCREMENT
2135                                    ;;**************************************************************
2136  004712  000004                    TST27:  SCOPE
2137
2138  004714  005077  174456                    CLR     @#ASR           ;CLEAR THE CSR.
2139  004720  005077  174454                    CLR     @#BR            ;CLEAR THE BUFFER
2140  004724  052777  000061  174444            BIS     #BIT5!BIT4!BIT0,@#ASR ;SET RATE:ST1, GO.
2141  004732  052777  000400  174436            BIS     #BIT8,@#ASR     ;GENERATE A MAINTENANCE ST1.
2142                                                                    ;DID THE CLOCK COUNT?
2143  004740  017746  174432                    MOV     @#ASR,-(6)      ;/SAVE CSR
2144  004744  011637  001424                    MOV     (6),$TMP3       ;/GET CSR.
2145  004750  042737  177707  001424            BIC     #177707,$TMP3   ;/SAVE RATE BITS.
2146  004756  052737  004005  001424            BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
2147  004764  013777  001424  174404            MOV     $TMP3,@#ASR             ;/LOAD CSR.
2148                                                                    ;/THIS MUST BE DONE IN
2149                                                                    ;/ORDER TO XFERR COUNTER
2150                                                                    ;/TO BUFFER ON ST2.
2151  004772  052777  001000  174376            BIS     #BIT9,@#ASR     ;/GENERATE ON ST2 PULSE
2152  005000  017737  174374  001126            MOV     @#BR,$BDDAT     ;/READ THE PRESET BUFFER,
2153                                                                    ;/PREVIOUS COUNTER
2154  005006  012677  174364                    MOV     (6)+,@#ASR      ;/CONTENTS ARE IN $BDDAT.
2155  005012  005737  001126                    TST     $BDDAT          ;/RESTORE CSR
2156  005016  001001                            BNE     1$              ;YES, NEXT TEST.
2157
                                        ;;SSSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

2160  005020  104006                            ERROR   6               ;CLOCK FAILED TO INCREMENT.
2161
2162
                                        ;;SSSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

2165  005022                            1$:
2166
2167                                    ;;**************************************************************
2168                                    ;*TEST 30        *SEE IF CLOCK WILL COUNT UP FROM A ZERO BASE, RATE:ST1
2169                                    ;*
2170                                    ;*      NOTE:   IN THIS TEST, LOOP ON ERROR WILL CAUSE A LOOP
2171                                    ;*              ON THE FAILING COUT PATTERN;
2172                                    ;*              WHILE LOOP ON TEST WILL START THE TEST
2173                                    ;*              AND THE CLOCK FROM ZERO TO THE FAILING COUNT.
2174                                    ;*
2175                                    ;;**************************************************************
2176  005022  000004                    TST30:  SCOPE
2177  005024  012737  000010  001160            MOV     #10,$TIMES      ;;DO 10 ITERATIONS
2178
2179  005032  005077  174340                    CLR     @#ASR           ;CLEAR THE CSR.
2180  005036  005077  174336                    CLR     @#BR            ;CLEAR THE BUFFER REG
```

```
2181  005042  012737  000000  001124          MOV     #0,$GDDAT       ;CLEAR EXPECTED.
2182  005050  012737  005212  001110          MOV     #2$,$LPERR
2183
2184  005056  052777  000061  174312  1$:     BIS     #BIT5!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

                                        ;;SSSSSSSSSSSSSSSSSSSSSS))) ERROR (((SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

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

                                        ;;SSSSSSSSSSSSSSSSSSSSSS))) ERROR (((SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

2200  005112  000447                          BR      TST31           ;;
2201  005114                          10$:
2202  005114  017746  174256                  MOV     @ASR,-(6)       ;/SAVE CSR
2203  005120  011637  001424                  MOV     (6),$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

                                        ;;SSSSSSSSSSSSSSSSSSSSSS))) ERROR (((SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

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

                                        ;;SSSSSSSSSSSSSSSSSSSSSS))) ERROR (((SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

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
```

```
2237
2238  005234  005737  001440          TST     EXS             ;TESTOR MODE ENABLED??
2239  005240  001407                  BEQ     2$              ;NO-THEN SKIP NEXT SECTION OF CODE.
2240
2241  005242  005037  170500          CLR     @#170500        ;CLEAR TESTOR A/D
2242  005246  005737  170502          TST     @#170502        ;DUMB READ OF A/D BUFFER.
2243  005252  052737  000040  170500  BIS     #BIT5,@#170500  ;ENABLE EXTRENAL START OF A/D.
2244  005260                  2$:
2245  005260  005077  174112          CLR     @ASR            ;CLEAR THE CSR
2246  005264  012777  177777  174106  MOV     #-1,@ABR        ;SET PRESET BUFFER TO ALL ONES.
2247
2248  005272  052777  000061  174076  BIS     #BIT5!BIT4!BIT0,@ASR ;START CLOCK, RATE ST1.
2249
2250  005300  052777  000400  174070  BIS     #BIT8,@ASR      ;COUNT CLOCK ONCE, OVERFLOW
2251                                          ;SHOULD OCCUR.
2252  005306  105777  174064          TSTB    @ASR            ;DID OVERFLOW SET?
2253  005312  100402                  BMI     1$              ;YES - THEN NEXT TEST
2254
2255

                        ;;SSSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

2258  005314  104006          ERROR   6               ;ERROR - OVERFLOW, CSR BIT0?
2259                                                  ;FAILED TO SET ON OVERFLOW
2260  005316  000407          BR      TST32           ;;
2261  005320  005737  001440  1$:     TST     EXS     ;TEST EXTERNAL SIGNALS?
2262  005324  001404          BEQ     TST32           ;;
2263  005326  105737  170500          TSTB    @#170500        ;IF  OVERFLOW GOT OUT,IT GAVE A/D START,
2264                                                  ;WE'RE LOOKING FOR A/D DONE-DID IT GET SET?
2265  005332  100401          BMI     TST32           ;;
2266

                        ;;SSSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

2269  005334  104006          ERROR   6               ;OVERFLOW OUT NOT DETECTED
2270                                                  ;BY TESTOR
2271

                        ;;SSSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

2274
2275
2276                    ;;**************************************************************
2277                    ;*TEST 32        *TEST THAT OVERFLOW WILL CLEAR THE GO BIT
2278                    ;;**************************************************************
2279  005336  000004    TST32:  SCOPE
2280
2281  005340  005077  174032          CLR     @ASR            ;CLEAR THE CSR.
2282
2283  005344  012777  177777  174026  MOV     #-1,@ABR        ;PRESET CLOCK TO -1.
2284
2285  005352  052777  000061  174016  BIS     #BIT5!BIT4!BIT0,@ASR ;START CLOCK, RATE:ST1
2286
2287  005360  052777  000400  174010  BIS     #BIT8,@ASR      ;COUNT ONCE, OVERFLOW
2288                                                  ;SHOULD OCCUR CLEARING
2289                                                  ;ENABLE (CSR BIT00)
2290
2291  005366  032777  000001  174002  BIT     #BIT0,@ASR      ;DID THE ENABLE CLEAR?
2292  005374  001401                  BEQ     1$              ;YES - NEXT TEST.
```

```
2293
2294
                                        ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

2297  005376  104006                            ERROR   6               ;ERROR - OVERFLOW FAILED
2298                                                                    ;TO CLEAR ENABLE (CSR BIT0C)
2299
2300  005400                                1$:
2301
2302                                        ;;**********************************************************************
2303                                        ;*TEST 33       *TEST THAT GO BIT DOES NOT CLEAR ON OVERFLOW, IF MODE 1
2304                                        ;;**********************************************************************
2305  005400  000004                        TST33:  SCOPE
2306
2307  005402  005077  173770                        CLR     @ASR            ;CLEAR THE CSR.
2308  005406  012777  177777  173764                MOV     #-1,@ABR        ;PRESET BUFFER=ONE COUNT FROM OVERFLOW.
2309  005414  052777  000063  173754                BIS     #63,@ASR        ;MODE 1, RATE:ST1, GO.
2310
2311  005422  052777  000400  173746                BIS     #BIT8,@ASR      ;GENERATE MAINTENANCE ST1.
2312
2313  005430  032777  000001  173740                BIT     #BIT0,@ASR      ;DID ENABLE (GO BIT) CLEAR?
2314  005436  001001                                BNE     1$              ;NO (GOOD) NEXT TEST.
2315
                                        ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

2318  005440  104006                            ERROR   6               ;GO BIT CLEARED ON OVERFLOW
2319                                                                    ;WHEN MODE 1 WAS SELECTED
2320
                                        ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

2323
2324  005442  005077  173730                1$:     CLR     @ASR            ;CLEAR THE CLOCK.
2325
2326
2327
2328
2329                                        ;;**********************************************************************
2330                                        ;*TEST 34       *TEST THE ABILITY OF CLOCK TO COUNT AT 1MHZ RATE
2331
2332                                        ;*
2333                                        ;*THIS TEST IS DESIGNED TO TEST THE CLOCK'S ABILITY
2334                                        ;*TO COUNT AT 1MHZ RATE.
2335                                        ;*
2336                                        ;;**********************************************************************
2337  005446  000004                        TST34:  SCOPE
2338  005450  012737  000005  001160                MOV     #5,$TIMES       ;;DO 5 ITERATIONS
2339
2340
2341  005456  005077  173714                        CLR     @ASR            ;/CLEAR CLOCK
2342  005462  005077  173712                        CLR     @ABR            ;/CLEAR PRESET BUFFER
2343  005466  012777  000011  173702                MOV     #BIT0!10,@ASR   ;/START CLOCK, MODE0, RATE:1MHZ
2344  005474  005000                                CLR     R0              ;/NOW WE'LL DO A LITTLE DELAY. THIS DELAY
2345
2346  005476  005200                        1$:     INC     R0              ;/WILL AMOUNT TO APPROXIMATELY
2347  005500  001376                                BNE     1$              ;/369 MS.
2348
```

```
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     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
2353  005526  013777  001424  173642  MOV     $TMP3,@ASR              ;/LOAD CSR.
2354                                                  ;/THIS MUST BE DONE IN
2355                                                  ;/ORDER TO XFERR COUNTER
2356                                                  ;/TO BUFFER ON ST2.
2357  005534  052777  001000  173634  BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
2358  005542  017737  173632  001126  MOV     @ABR,$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.
2366
2367

                                ;;SSSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

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

                                ;;SSSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

2375
2376  005572  005077  173600  2$:     CLR     @ASR            ;/CLEAR THE CLOCK.
2377
2378
2379
2380                            ;;**************************************************************
2381                            ;*TEST 35        *TEST THE ABILITY OF CLOCK TO COUNT AT 100KHZ RATE
2382
2383                            ;*
2384                            ;*THIS TEST IS DESIGNED TO TEST THE CLOCK'S ABILITY
2385                            ;*TO COUNT AT 100KHZ RATE.
2386                            ;*
2387                            ;;**************************************************************
2388  005576  000004            TST35:  SCOPE
2389  005600  012737  000005  001160          MOV     #5,$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,@ASR   ;/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.
2399
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     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
2404  005656  013777  001424  173512  MOV     $TMP3,@ASR              ;/LOAD CSR.
```

```
2405                                                            ;/THIS MUST BE DONE IN
2406                                                            ;/ORDER TO XFERR COUNTER
2407                                                            ;/TO BUFFER ON ST2.
2408  005664  052777  001000  173504          BIS     #BIT9,@ASR        ;/GENERATE ON ST2 PULSE
2409  005672  017737  173502  001126          MOV     @ABR,$BDDAT       ;/READ THE PRESET BUFFER,
2410                                                            ;/PREVIOUS COUNTER
2411  005700  012677  173472                  MOV     (6)+,@ASR         ;/CONTENTS ARE IN $BDDAT.
2412  005704  005737  001126                  TST     $BDDAT            ;/RESTORE CSR
2413  005710  001004                          BNE     2$                ;/YES - NEXT TEST.
2414  005712  105766  177776                  TSTB    -2(6)             ;/AT HIGH RATE MAY HAVE HAD OVERFLOW
2415                                                            ;/NOTE: CSR HAD BEEN PUT ON STACK.
2416  005716  100401                          BMI     2$                ;/NEXT TEST IF OVERFLOW.
2417
2418
                                        ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSS

2421  005720  104006                          ERROR   6                 ;/CLOCK FAILED TO COUNT AT
2422                                                            ;/RATE:100KHZ
2423
                                        ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSS

2426
2427  005722  005077  173450          2$:     CLR     @ASR              ;/CLEAR THE CLOCK.
2428
2429
2430
2431                                    ;;*****************************************************************
2432                                    ;*TEST 36        *TEST THE ABILITY OF CLOCK TO COUNT AT 10KHZ RATE
2433
2434                                    ;*
2435                                    ;*THIS TEST IS DESIGNED TO TEST THE CLOCK'S ABILITY
2436                                    ;*TO COUNT AT 10KHZ RATE.
2437                                    ;*
2438                                    ;*****************************************************************
2439  005726  000004                    TST36:  SCOPE
2440  005730  012737  000005  001160          MOV     #5,$TIMES         ;;DO 5 ITERATIONS
2441
2442
2443  005736  005077  173434                  CLR     @ASR              ;/CLEAR CLOCK
2444  005742  005077  173432                  CLR     @ABR              ;/CLEAR PRESET BUFFER
2445  005746  012777  000031  173422          MOV     #BIT0!30,@ASR     ;/START CLOCK, MODE0, RATE:10KHZ
2446  005754  005000                          CLR     R0                ;/NOW WE'LL DO A LITTLE DELAY. THIS DELAY
2447
2448  005756  005200                  1$:     INC     R0                ;/WILL AMOUNT TO APPROXIMATELY
2449  005760  001376                          BNE     1$                ;/369 MS.
2450
2451  005762  017746  173410                  MOV     @ASR,-(6)         ;/SAVE CSR
2452  005766  011637  001424                  MOV     (6),$TMP3         ;/GET CSR.
2453  005772  042737  177707  001424          BIC     #177707,$TMP3     ;/SAVE RATE BITS.
2454  006000  052737  004005  001424          BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
2455  006006  013777  001424  173362          MOV     $TMP3,@ASR                ;/LOAD CSR.
2456                                                            ;/THIS MUST BE DONE IN
2457                                                            ;/ORDER TO XFERR COUNTER
2458                                                            ;/TO BUFFER ON ST2.
2459  006014  052777  001000  173354          BIS     #BIT9,@ASR        ;/GENERATE ON ST2 PULSE
2460  006022  017737  173352  001126          MOV     @ABR,$BDDAT       ;/READ THE PRESET BUFFER,
```

```
2461                                                                    ;/PREVIOUS COUNTER
2462  006030  012677  173342                MOV     (6)+,@ASR         ;/CONTENTS ARE IN $BDDAT.
2463  006034  005737  001126                TST     $BDDAT            ;/RESTORE CSR
2464  006040  001004                        BNE     2$                ;/YES - NEXT TEST.
2465  006042  105766  177776                TSTB    -2(6)             ;/AT HIGH RATE MAY HAVE HAD OVERFLOW
2466                                                                    ;/NOTE: CSR HAD BEEN PUT ON STACK.
2467  006046  100401                        BMI     2$                ;/NEXT TEST IF OVERFLOW.
2468
2469
                                        ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

2472  006050  104006                        ERROR   6                 ;/CLOCK FAILED TO COUNT AT
2473                                                                    ;/RATE:10KHZ
2474
                                        ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

2477
2478  006052  005077  173320          2$:     CLR     @ASR              ;/CLEAR THE CLOCK.
2479
2480
2481
2482                                    ;;*****************************************************************
2483                                    ;*TEST 37        *TEST THE ABILITY OF CLOCK TO COUNT AT 1KHZ RATE
2484
2485                                    ;*
2486                                    ;*THIS TEST IS DESIGNED TO TEST THE CLOCK'S ABILITY
2487                                    ;*TO COUNT AT 1KHZ RATE.
2488                                    ;*
2489                                    ;;*****************************************************************
2490  006056  000004                    TST37:  SCOPE
2491  006060  012737  000005  001160            MOV     #5,$TIMES         ;;DO 5 ITERATIONS
2492
2493
2494  006066  005077  173304                    CLR     @ASR              ;/CLEAR CLOCK
2495  006072  005077  173302                    CLR     @ABR              ;/CLEAR PRESET BUFFER
2496  006076  012777  000041  173272            MOV     #BIT0!40,@ASR     ;/START CLOCK, MODE0, RATE:1KHZ
2497  006104  005000                            CLR     R0                ;/NOW WE'LL DO A LITTLE DELAY.  THIS DELAY
2498
2499  006106  005200                    1$:     INC     R0                ;/WILL AMOUNT TO APPROXIMATELY
2500  006110  001376                            BNE     1$                ;/369 MS.
2501
2502  006112  017746  173260                    MOV     @ASR,-(6)         ;/SAVE CSR
2503  006116  011637  001424                    MOV     (6),$TMP3         ;/GET CSR.
2504  006122  042737  177707  001424            BIC     #177707,$TMP3     ;/SAVE RATE BITS.
2505  006130  052737  004005  001424            BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
2506  006136  013777  001424  173232            MOV     $TMP3,@ASR                ;/LOAD CSR.
2507                                                                    ;/THIS MUST BE DONE IN
2508                                                                    ;/ORDER TO XFERR COUNTER
2509                                                                    ;/TO BUFFER ON ST2.
2510  006144  052777  001000  173224            BIS     #BIT9,@ASR        ;/GENERATE ON ST2 PULSE
2511  006152  017737  173222  001126            MOV     @ABR,$BDDAT       ;/READ THE PRESET BUFFER,
2512                                                                    ;/PREVIOUS COUNTER
2513  006160  012677  173212                    MOV     (6)+,@ASR         ;/CONTENTS ARE IN $BDDAT.
2514  006164  005737  001126                    TST     $BDDAT            ;/RESTORE CSR
2515  006170  001004                            BNE     2$                ;/YES - NEXT TEST.
2516  006172  105766  177776                    TSTB    -2(6)             ;/AT HIGH RATE MAY HAVE HAD OVERFLOW
```

```
2517
2518  006176  100401                          BMI     2$              ;/NOTE: CSR HAD BEEN PUT ON STACK.
2519                                                                  ;/NEXT TEST IF OVERFLOW.
2520
                                        ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS
2523  006200  104006                          ERROR   6               ;/CLOCK FAILED TO COUNT AT
2524                                                                  ;/RATE:1KHZ
2525
                                        ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS
2528
2529  006202  005077  173170          2$:     CLR     @ASR            ;/CLEAR THE CLOCK.
2530
2531
2532
2533                                    ;;**************************************************************
2534                                    ;*TEST 40        *TEST THE ABILITY OF CLOCK TO COUNT AT 100HZ RATE
2535
2536                                    ;*
2537                                    ;*THIS TEST IS DESIGNED TO TEST THE CLOCK'S ABILITY
2538                                    ;*TO COUNT AT 100HZ RATE.
2539                                    ;*
2540                                    ;**************************************************************
2541  006206  000004                    TST40:  SCOPE
2542  006210  012737  000005  001160            MOV     #5,$TIMES       ;;DO 5 ITERATIONS
2543
2544
2545  006216  005077  173154                    CLR     @ASR            ;/CLEAR CLOCK
2546  006222  005077  173152                    CLR     @ABR            ;/CLEAR PRESET BUFFER
2547  006226  012777  000051  173142            MOV     #BIT0!50,@ASR   ;/START CLOCK, MODE0, RATE:100HZ
2548  006234  005000                            CLR     R0              ;/NOW WE'LL DO A LITTLE DELAY. THIS DELAY
2549
2550  006236  005200                    1$:     INC     R0              ;/WILL AMOUNT TO APPROXIMATELY
2551  006240  001376                            BNE     1$              ;/369 MS.
2552
2553  006242  017746  173130                    MOV     @ASR,-(6)       ;/SAVE CSR
2554  006246  011637  001424                    MOV     (6),$TMP3       ;/GET CSR.
2555  006252  042737  177707  001424            BIC     #177707,$TMP3   ;/SAVE RATE BITS.
2556  006260  052737  004005  001424            BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
2557  006266  013777  001424  173102            MOV     $TMP3,@ASR              ;/LOAD CSR.
2558                                                                  ;/THIS MUST BE DONE IN
2559                                                                  ;/ORDER TO XFERR COUNTER
2560                                                                  ;/TO BUFFER ON ST2.
2561  006274  052777  001000  173074            BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
2562  006302  017737  173072  001126            MOV     @ABR,$BODAT     ;/READ THE PRESET BUFFER,
2563                                                                  ;/PREVIOUS COUNTER
2564  006310  012677  173062                    MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BODAT.
2565  006314  005737  001126                    TST     $BODAT          ;/RESTORE CSR
2566  006320  001004                            BNE     2$              ;/YES - NEXT TEST.
2567  006322  105766  177776                    TSTB    -2(6)           ;/AT HIGH RATE MAY HAVE HAD OVERFLOW
2568                                                                  ;/NOTE: CSR HAD BEEN PUT ON STACK.
2569  006326  100401                            BMI     2$              ;/NEXT TEST IF OVERFLOW.
2570
2571
                                        ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS
```

```
                                        ERROR   6               ;/CLOCK FAILED TO COUNT AT
                                                                ;/RATE:100HZ

                                ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSS

                        2S:     CLR     @ASR            ;/CLEAR THE CLOCK.


                                ;;*****************************************************************
                                ;*TEST 41        *TEST THE ABILITY OF CLOCK TO COUNT AT LINEFREQ RATE

                                ;*
                                ;*THIS TEST IS DESIGNED TO TEST THE CLOCK'S ABILITY
                                ;*TO COUNT AT LINEFREQ RATE.
                                ;*
                                ;;*****************************************************************
006336  000004          TST41:  SCOPE
006340  012737  000005  001160          MOV     #5,$TIMES       ;;DO 5 ITERATIONS

006346  032777  010000  172564          BIT     #BIT12,@SWR     ;SHALL WE TEST
                                                                ;LINE FREQ?
                                                                ;SW12=1
006354  001450                          BEQ     TST42           ;;

006356  005077  173014                  CLR     @ASR            ;/CLEAR CLOCK
006362  005077  173012                  CLR     @ABR            ;/CLEAR PRESET BUFFER
006366  012777  000071  173002          MOV     #BIT0!70,@ASR   ;/START CLOCK, MODE0, RATE:LINEFREQ
006374  005000                          CLR     R0              ;/NOW WE'LL DO A LITTLE DELAY. THIS DELAY

006376  005200                  1S:     INC     R0              ;/WILL AMOUNT TO APPROXIMATELY
006400  001376                          BNE     1S              ;/369 MS.

006402  017746  172770                  MOV     @ASR,-(6)       ;/SAVE CSR
006406  011637  001424                  MOV     (6),$TMP3       ;/GET CSR.
006412  042737  177707  001424          BIC     #177707,$TMP3   ;/SAVE RATE BITS.
006420  052737  004005  001424          BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
006426  013777  001424  172742          MOV     $TMP3,@ASR              ;/LOAD CSR.
                                                                ;/THIS MUST BE DONE IN
                                                                ;/ORDER TO XFERR COUNTER
                                                                ;/TO BUFFER ON ST2.
006434  052777  001000  172734          BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
006442  017737  172732  001126          MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
                                                                ;/PREVIOUS COUNTER
006450  012677  172722                  MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
006454  005737  001126                  TST     $BDDAT          ;/RESTORE CSR
006460  001004                          BNE     2S              ;/YES - NEXT TEST.
006462  105766  177776                  TSTB    -2(6)           ;/AT HIGH RATE MAY HAVE HAD OVERFLOW
                                                                ;/NOTE: CSR HAD BEEN PUT ON STACK.
006466  100401                          BMI     2S              ;/NEXT TEST IF OVERFLOW.


                                ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSS
```

```
                                        ERROR   5               ;/CLOCK FAILED TO COUNT AT
                                                                ;/RATE:LINEFREQ

                                ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSS


006472  005077  172700L        2$:     CLR     @ASR            ;/CLEAR THE CLOCK.



                                ;;**************************************************************
                                ;*TEST 42        *TEST THE ABILITY OF CLOCK TO COUNT AT ST1 RATE

                                ;*
                                ;*THIS TEST IS DESIGNED TO TEST THE CLOCK'S ABILITY
                                ;*TO COUNT AT ST1 RATE.
                                ;*
                                ;**************************************************************
006476  000004                  ST42:   SCOPE
006500  012737  000005  001160          MOV     #5,$TIMES       ;;DO 5 ITERATIONS


006506  005077  172664                  CLR     @ASR            ;/CLEAR CLOCK
006512  005077  172662                  CLR     @ABR            ;/CLEAR PRESET BUFFER
006516  012777  000061  172652          MOV     #BIT0!60,@ASR   ;/START CLOCK, MODE0, RATE:ST1
006524  052777  000400  172644          BIS     #BIT8,@ASR
006532  017746  172640                  MOV     @ASR,-(6)       ;/SAVE CSR
006536  011637  001424                  MOV     (6),$TMP3       ;/GET CSR.
006542  042737  177707  001424          BIC     #177707,$TMP3   ;/SAVE RATE BITS.
006550  052737  004005  001424          BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
006556  013777  001424  172612          MOV     $TMP3,@ASR              ;/LOAD CSR.
                                                                ;/THIS MUST BE DONE IN
                                                                ;/ORDER TO XFERR COUNTER
                                                                ;/TO BUFFER ON ST2.
006564  052777  001000  172604          BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
006572  017737  172602  001126          MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
                                                                ;/PREVIOUS COUNTER
006600  012677  172572                  MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
006604  005737  001126                  TST     $BDDAT          ;/RESTORE CSR
006610  001004                          BNE     2$              ;/YES - NEXT TEST.
006612  105766  177776                  TSTB    -2(6)           ;/AT HIGH RATE MAY HAVE HAD OVERFLOW
                                                                ;/NOTE: CSR HAD BEEN PUT ON STACK.
006616  100401                          BMI     2$              ;/NEXT TEST IF OVERFLOW.


                                ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSS

006620  104006                          ERROR   6               ;/CLOCK FAILED TO COUNT AT
                                                                ;/RATE:ST1

                                ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSS


006622  005077  172550          2$:     CLR     @ASR            ;/CLEAR THE CLOCK.
```

```
                                        ;;**********************************************************************
                                        ;*TEST 43        *TEST THAT COUNTER DOESN'T COUNT WHEN "SLAVE IN" RATE IS SELECTED
                                        ;;**********************************************************************
006626  000004                  TST43:  SCOPE
006630  012737  000020  001160          MOV     #20,$TIMES      ;;DO 20 ITERATIONS

006636  005077  172534              CLR     @#ASR           ;CLEAR CSR.
006642  005077  172532              CLR     @#ABR           ;CLEAR PRESET BUFFER.

006646  052777  000001  172522      BIS     #BIT0,@#ASR     ;SET GO BIT, RATE:"SLAVE IN"

006654  052777  000400  172514      BIS     #BIT8,@#ASR     ;GENERATE A MAINTENANCE ST1.
006662  005000                      CLR     R0

006664  005200              1$:     INC     R0              ;NOW DO A SHORT DELAY.
006666  001376                      BNE     1$

006670  017746  172502              MOV     @#ASR,-(6)      ; SAVE CSR
006674  011637  001424              MOV     (6),$TMP3       ;/GET CSR.
006700  042737  177707  001424      BIC     #177707,$TMP3   ;/SAVE RATE BITS.
006706  052737  004005  001424      BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
006714  013777  001424  172454      MOV     $TMP3,@#ASR             ;/LOAD CSR.
                                                        ;/THIS MUST BE DONE IN
                                                        ;/ORDER TO XFERR COUNTER
                                                        ;/TO BUFFER ON ST2.
006722  052777  001000  172446      BIS     #BIT9,@#ASR     ;/GENERATE ON ST2 PULSE
006730  017737  172444  001126      MOV     @#ABR,$BDDAT    ;/READ THE PRESET BUFFER,
                                                        ;/PREVIOUS COUNTER
006736  017677  172434              MOV     (6)+,@#ASR      ;/CONTENTS ARE IN $BDDAT.
006742  005737  001126              TST     $BDDAT          ;/RESTORE CSR
006746  001003                      BNE     2$              ;IF ANY COUNT, ERROR.
006750  105777  172422              TSTB    @#ASR           ;CHECK FOR OVERFLOW.
006754  100003                      BPL     3$              ;IF NO OVERFLOW, NEXT TEST

006756  005037  001124          2$: CLR     $GDDAT          ;CLEAR FOR TYPEOUT, EXP'D ZERO.

                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

006762  104011                      ERROR   11              ;CLOCK COUNTED WHEN RATE:"SLAVE IN"
                                                        ;WAS SELECTED.

                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

006764  005077  172406          3$: CLR     @#ASR           ;CLEAR CLOCK'S CSR.


                                        ;;**********************************************************************
                                        ;*TEST 44        *TEST THAT THE CLOCK WILL COUNT IN MODE 1
                                        ;;**********************************************************************
006770  000004                  TST44:  SCOPE

006772  005077  172400              CLR     @#ASR           ;CLEAR THE CSR.
006776  012777  177777  172374      MOV     #-1,@#ABR       ;PRESET BUFFER REG.
007004  052777  000013  172364      BIS     #13,@#ASR       ;START CLOCK,RATE:1 MHZ,MODE 1.
007012  005000                      CLR     R0
```

```
2741  007014  105200                  1$:     INCB    R0              ;NOW DO A SHORT DELAY SO THAT THE
2742  007016  001376                          BNE     1$              ;CLOCK CAN COUNT TO OVERFLOW.
2743
2744  007020  105777  172352                  TSTB    @#ASR           ;OVERFLOW SHOULD HAVE SET.
2745  007024  100401                          BMI     2$              ;GOTO NEXT TEST IF SO.
2746
2747
                                        ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSS

2750  007026  104006                          ERROR   6               ;CLOCK FAILED TO COUNT UP AND SET
2751                                                                  ;OVERFLOW IN MODE 1.
2752
                                        ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSS

2755  007030                            2$:
2756
2757
2758                                            .SBTTL  *
2759                                            .SBTTL  *PHASE 4 CLOCK INTERRUPT TEST.
2760                                            .SBTTL  *
2761
2762
2763                                    ;;**************************************************************
2764                                    ;*TEST 45       *TEST THAT THE CLOCK WILL INTERRUPT ON OVERFLOW
2765                                    ;;**************************************************************
2766  007030  000004                    TST45:  SCOPE
2767  007032  012737  000020  001160            MOV     #20,$TIMES      ;;DO 20 ITERATIONS
2768
2769
2770  007040  012746  000340                    MOV     #340,-(SP)      ;PUT PRIORITY ON STACK.
2771  007044  012746  007052                    MOV     #64$,-(SP)      ;PUT RETURN ADDRESS ON STACK
2772  007050  000002                            RTI                     ;DO AN RTI, PUTS PRIORITY IN CPU.
2773  007052                            64$:
2774
2775  007052  005077  172320                    CLR     @#ASR           ;CLEAR CLOCK'S CSR.
2776  007056  012777  177777  172314            MOV     #-1,@#ABR       ;SET PRESET BUFFER TO ALL ONES.
2777
2778  007064  012777  000161  172304            MOV     #161,@#ASR      ;START CLOCK, RATE:ST1.
2779  007072  052777  000400  172276            BIS     #BIT8,@#ASR     ;GENERATE A MAINTENANCE ST1.
2780  007100  012777  007140  172274            MOV     #1$,@#VECT1     ;SET INTERRUPT ADDR.
2781
2782  007106  012746  000000                    MOV     #0,-(SP)        ;PUT PRIORITY ON STACK.
2783  007112  012746  007120                    MOV     #65$,-(SP)      ;PUT RETURN ADDRESS ON STACK
2784  007116  000002                            RTI                     ;DO AN RTI, PUTS PRIORITY IN CPU.
2785  007120                            65$:
2786
2787  007120  000240                            NOP                     ;STALL TIME
2788
2789  007122  012746  000340                    MOV     #340,-(SP)      ;PUT PRIORITY ON STACK.
2790  007126  012746  007134                    MOV     #66$,-(SP)      ;PUT RETURN ADDRESS ON STACK
2791  007132  000002                            RTI                     ;DO AN RTI, PUTS PRIORITY IN CPU.
2792  007134                            66$:
2793
                                        ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSS

2796  007134  104007                          ERROR   7               ;CLOCK FAILED TO INTERRUPT.
```

```
2797
                                        ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSS

2800  007136  000402                    BR      2$
2801  007140                          1$:
2802  007140  062706  000004            ADD     #4,SP                   ;/ADD #4 TO STACK POINTER.
2803  007144  005077  172226          2$:   CLR     @#ASR           ;CLEAR THE CLOCK.
2804
2805                                    ;;**************************************************************
2806                                    ;*TEST 46       *TEST THAT ST2 WILL CAUSE AN INTERRUPT
2807                                    ;;**************************************************************
2808  007150  000004                    TST46:  SCOPE
2809
2810
2811  007152  012746  000340            MOV     #340,-(SP)      ;PUT PRIORITY ON STACK.
2812  007156  012746  007164            MOV     #64$,-(SP)      ;PUT RETURN ADDRESS ON STACK
2813  007162  000002                    RTI                     ;DO AN RTI, PUTS PRIORITY IN CPU.
2814  007164                          64$:
2815
2816  007164  005077  172206            CLR     @#ASR           ;CLEAR CLOCKS CSR.
2817  007170  012777  007244  172210    MOV     #1$,@#VECT2     ;SET UP INTERRUPT VECTOR.
2818  007176  012777  040001  172172    MOV     #BIT14!BIT0,@#ASR;SET "INT2",AND GO BIT.
2819  007204  052777  001000  172164    BIS     #BIT9,@#ASR     ;GENERATE A MAINTENANCE ST2.
2820
2821  007212  012746  000000            MOV     #0,-(SP)        ;PUT PRIORITY ON STACK.
2822  007216  012746  007224            MOV     #65$,-(SP)      ;PUT RETURN ADDRESS ON STACK
2823  007222  000002                    RTI                     ;DO AN RTI, PUTS PRIORITY IN CPU.
2824  007224                          65$:
2825
2826  007224  000240                    NOP                     ;STALL TIME
2827
2828  007226  012746  000340            MOV     #340,-(SP)      ;PUT PRIORITY ON STACK.
2829  007232  012746  007240            MOV     #66$,-(SP)      ;PUT RETURN ADDRESS ON STACK
2830  007236  000002                    RTI                     ;DO AN RTI, PUTS PRIORITY IN CPU.
2831  007240                          66$:
2832
                                        ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSS

2835  007240  104007                    ERROR   7               ;CLOCK FAILED TO INTERRUPT ON ST2.
2836
                                        ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSS

2839  007242  000402                    BR      2$
2840  007244                          1$:
2841  007244  062706  000004            ADD     #4,SP                   ;/ADD #4 TO STACK POINTER.
2842  007250  005077  172122          2$:   CLR     @#ASR           ;CLEAR CLOCK'S CSR.
2843
2844                                    .SBTTL  *
2845                                    .SBTTL  *PHASE 5 ADVANCED TESTING
2846                                    .SBTTL  *
2847
2848                                    ;;**************************************************************
2849                                    ;*TEST 47       *TEST THAT THE "FOR" BIT WILL SET ON 2 ST2'S
2850                                    ;;**************************************************************
2851  007254  000004                    TST47:  SCOPE
2852
```

```
2853  007256  005077  172114          CLR     @ASR            ;START WITH CSR CLEAR.
2854  007262  005277  172110          INC     @ASR            ;SET GO BIT.
2855  007266  052777  001000  172102  BIS #BIT9,@ASR          ; GENERATE THE 1ST ST2 PULSE.
2856  007274  052777  001000  172074  BIS     #BIT9,@ASR      ;GENERATE 2ND ST2 PULSE.
2857                                                          ;THIS SHOULD CAUSE FOR BIT TO SET.
2858  007302  032777  010000  172066  BIT     #BIT12,@ASR     ;DID FOR BIT SET?
2859  007310  001007                  BNE     1$              ;YES-THEN NEXT TEST.
2860
2861  007312  017737  172060  001126  MOV     @ASR,$BDDAT     ;RECORD CSR.
2862  007320  012737  110001  001124  MOV     #BIT15!BIT12!BIT0,$GDDAT        ;RECORD S/B.
2863

                                      ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSS

2866  007326  104001                  ERROR   1               ;ERROR-"FOR" BIT FAILED TO SET ON
2867                                                          ;ON TWO SUCCESIVE ST2 PULSES.
2868

                                      ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSS

2871  007330                          1$:
2872
2873                                  ;;**********************************************************************
2874                                  ;*TEST 50        * TST THAT FOR BIT WILL SET ON TWO OVERFLOWS
2875                                  ;;**********************************************************************
2876  007330  000004                  TST50:  SCOPE
2877  007332  012737  000002  001160          MOV     #2,$TIMES       ;;DO 2 ITERATIONS
2878
2879  007340  005077  172032          CLR     @ASR            ;START WITH CSR CLEAR.
2880  007344  012777  177777  172026  MOV     #-1,@ABR        ;PRELOAD BUFFER REG.
2881  007352  052777  000013  172016  BIS     #13,@ASR        ;START CLOCK,MODE 1,1MHZ.
2882  007360  005000                  CLR     R0              ;NOW DO A SHORT DELAY.
2883  007362  105200          1$:     INCB    R0              ;DURING THIS DELAY, THE CLOCK WILL
2884  007364  001376                  BNE     1$              ;HAVE OVERFLOWED TWICE-
2885                                                          ;THIS SHOULD CAUSE THE FOR BIT TO SET.
2886
2887  007366  032777  010000  172002  BIT     #BIT12,@ASR     ;DID FOR SET?
2888  007374  001007                  BNE     2$              ;YES-NEXT TEST.
2889
2890  007376  017737  171774  001126  MOV     @ASR,$BDDAT     ;NO-RECORD THE CSR.
2891  007404  012737  010213  001124  MOV     #G10213,$GDDAT  ;RECORD S/B.
2892

                                      ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSS

2895  007412  104001                  ERROR   1               ;ERROR FOR BIT FAILED TO SET ON
2896                                                          ; TWO SUCCESSIVE OVERFLOWS.
2897

                                      ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSS

2900  007414                          2$:
2901                                  ;;**********************************************************************
2902                                  ;*TEST 51        *TEST THAT FOR BIT WILL CLEAR IF GO BIT IS SET
2903                                  ;;**********************************************************************
2904  007414  000004                  TST51:  SCOPE
2905
2906  007416  012777  000001  171752  MOV     #BIT0,@ASR      ;CLEAR CSR,SET GO BIT.
2907  007424  052777  001000  171744  BIS     #BIT9,@ASR      ;SET 1ST ST2 PULSE.
2908  007432  052777  001000  171736  BIS     #BIT9,@ASR      ;GENERATE 2ND ST2 PULSE.
```

```
 2909                                                          ;FOR BIT SETS HERE.
 2910  007440  042777  000001  171730          BIC     #BIT0,@ASR      ;CLEAR GO BIT.
 2911
 2912  007446  052777  000001  171722          BIS     #BIT0,@ASR      ;SET THE "GO" BIT AGAIN -
```

```
 2913                                                                ;SHOULD CLEAR FOR BIT.
 2914
 2915  007454  017737  171716  001126          MOV     @ASR,$BDDAT   ;READ THE CSR.
```

```
MAINDEC-11-DVKWA-A     MACY11 27(732)  04-OCT-76  14:57  PAGE 73
DVKWA.P11       TST    *TEST THAT FOR BIT WILL CLEAR IF GO BIT IS SET

 2916  007462  012737  100001  001124          MOV     #100001,$GDDAT  ;RECORD WHAT CSR S/B.
 2917  007470  032737  010000  001126          BIT     #BIT12,$BDDAT   ;DID FOR BIT CLEAR?
 2918  007476  001401                          BEQ     1$              ;YES NEXT TEST.
 2919
                                        ;;SSSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

 2922  007500  104001                          ERROR   1               ;FOR BIT FALED TO CLEAR
 2923                                                                  ;WHEN "GO" BIT WAS SET.
 2924
                                        ;;SSSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

 2927  007502  005077  171670          1$:     CLR     @#SR            ;CLEAR THE CSR.
 2928
```

```
MAINDEC-11-DVKWA-A      MACY11 27(732)  04-OCT-76  14:57  PAGE 74
DVKWA.P11       TST     *TEST THAT WE CAN DISABLE THE INTERNAL OSC

                                        ;;*****************************************************************
                                        ;*TEST 52       *TEST THAT WE CAN DISABLE THE INTERNAL OSC
                                        ;;*****************************************************************
                                        TST52:  SCOPE
                                                MOV     #5,$TIMES       ;;DO 5 ITERATIONS

                                                CLR     @ASR            ;CLEAR THE CSR
```

```
  2936  007522  005077  171652                  CLR     @ABR            ;CLEAR THE PRESET BUFFER
```

```
 2937  007526  005037  001124                  CLR     $GDDAT          ;CLEAR EXPED.
 2938
```

```
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     R0
2942  007550  105200                  1$:     INCB    R0              ;DELAY A SHORT TIME.
2943  007552  001376                          BNE     1$
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     $TMP3,@ASR              ;/LOAD CSR.
2949                                                                  ;/THIS MUST BE DONE IN
2950                                                                  ;/ORDER TO XFERR COUNTER
2951                                                                  ;/TO BUFFER ON ST2.
2952  007606  052777  001000  171562          BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
2953  007614  017737  171560  001126          MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
2954                                                                  ;/PREVIOUS COUNTER
2955  007622  012677  171550                  MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
2956  007626  005737  001126                  TST     $BDDAT          ;/RESTORE CSR
2957  007632  001401                          BEQ     2$              ;NO - GOOD - NEXT TEST.
2958

                                        ;;SSSSSSSSSSSSSSSSSSSSSS>>> ERROR <<<SSSSSSSSSSSSSSSSSSSSSSSSSSSS

2961  007634  104011                          ERROR   11              ;CLOCK DISABLE INTERNAL
2962                                                                  ;OSC. DID NOT WORK.
2963

                                        ;;SSSSSSSSSSSSSSSSSSSSSS>>> ERROR <<<SSSSSSSSSSSSSSSSSSSSSSSSSSSS

2966  007636  005077  171534          2$:     CLR     @ASR            ;CLEAR THE CSR.
2967
2968                                    ;;**************************************************************
2969                                    ;*TEST 53        *TEST THAT CLOCK CAN BE COUNTED USING MAINTENANCE OSC
2970                                    ;;**************************************************************
2971  007642  000004                  TST53:  SCOPE
2972
2973  007644  005077  171526                  CLR     @ASR            ;CLEAR THE CSR.
2974  007650  005077  171524                  CLR     @ABR            ;CLEAR THE PRESET BUFFER.
2975  007654  052777  004000  171514          BIS     #BIT11,@ASR     ;DISABLE THE INTERNAL OSC.
2976  007662  052777  000011  171506          BIS     #BIT3!BIT0,@ASR ;START CLOCK, 1MHZ, GO.
2977  007670  012700  177754                  MOV     #-20.,R0        ;SET TO COUNT 20 TIMES
2978
2979  007674  052777  002000  171474  1$:     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     R0              ;DO 20 MAINTENANCE OSC.
2983  007704  001373                          BNE     1$
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     $TMP3,@ASR              ;/LOAD CSR.
2990                                                                  ;/THIS MUST BE DONE IN
2991                                                                  ;/ORDER TO XFERR COUNTER
2992                                                                  ;/TO BUFFER ON ST2.
2993  007740  052777  001000  171430          BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
2994  007746  017737  171426  001126          MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
```

```
                                                                      ;/PREVIOUS COUNTER
 007754  012677  171416                  MOV     (6)+,@ASR            ;/CONTENTS ARE IN $BDDAT.
 007760  005737  001126                  TST     $BDDAT               ;/RESTORE CSR
 007764  001001                          BNE     2$                   ;YES - NEXT TEST.


                                  ;;$$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 007766  104011                          ERROR   11                   ;ERROR COULD NOT COUNT USING
                                                                      ;MAINTENANCE OSC.

                                  ;;$$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 007770  005077  171402          2$:     CLR     @ASR                 ;CLEAR THE CSR.

```

```



                                          ;;*************************************************************
                                          ;*TEST 54        *TEST THE CLOCK'S 1MHZ DIVIDER
                                          ;;*************************************************************
007774  000004                            TST54:  SCOPE
007776  012737  000005  001160                    MOV     #5,$TIMES       ;;DO 5 ITERATIONS

010004  005077  171366                            CLR     @ASR            ;/CLEAR THE CSR.
010010  005077  171364                            CLR     @ABR            ;/CLEAR THE PRESET BUFFER.
010014  052777  004000  171354                    BIS     #BIT11,@ASR     ;/DISABLE THE INTERNAL OSC.
010022  052777  000011  171346                    BIS     #1!10,@ASR      ;/ENABLE CLOCK, RATE:1MHZ



010030  012700  177766            10$:    MOV     #-10.,R0        ;/SET TO GENERATE 10 OSC PULSES.

010034  052777  002000  171334    1$:     BIS     #BIT10,@ASR     ;/GENERATE ONE OSC PULSE.
010042  005200                            INC     R0              ;/DONE 10 OSC PULSES?
010044  001373                            BNE     1$              ;/NO - DO ANOTHER ONE.


010046  012737  000001  001124    2$:     MOV     #1,$GDDAT       ;/SET FOR ERROR TYPEOUT - IF ANY.
010054  017746  171316                    MOV     @ASR,-(6)       ;/SAVE CSR
010060  011637  001424                    MOV     (6),$TMP3       ;/GET CSR.
010064  042737  177707  001424            BIC     #177707,$TMP3   ;/SAVE RATE BITS.
010072  052737  004005  001424            BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
010100  013777  001424  171270            MOV     $TMP3,@ASR               ;/LOAD CSR.
                                                                  ;/THIS MUST BE DONE IN
                                                                  ;/ORDER TO XFERR COUNTER
                                                                  ;/TO BUFFER ON ST2.
010106  052777  001000  171262            BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
010114  017737  171260  001126            MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
                                                                  ;/PREVIOUS COUNTER
010122  012677  171250                    MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
010126  005737  001126                    TST     $BDDAT          ;/RESTORE CSR
010132  013737  001124  001420            MOV     $GDDAT,$TMP0    ;/$TMP0 USED IN ERROR TYPEOUT.

010140  023737  001124  001126            CMP     $GDDAT,$BDDAT   ;/DID CLOCK ADVANCE ONCE?


010146  001402                            BEQ     3$              ;/YES - NEXT TEST.

                                  ;;$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

010150  104011                            ERROR   11              ;/ERROR ON CLOCK1MHZ PULSE
                                                                  ;/NOT GENERATED WHEN 10
                                                                  ;/OSC PULSES GENERATED.

                                  ;;$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

010152  000443                            BR      5$
010154  012700  000011            3$:     MOV     #9.,R0  ;/GET THE NUMBER OF MORE OSC PULSES
                                                                  ;/TO BE GENERATED.

010160  052777  002000  171210    4$:     BIS     #BIT10,@ASR     ;/GENERATE ANOTHER OSC PULSE.
```

```
MACRO-11-DVKWA-A     MACY11 27(732)  04-OCT-76  14:57  PAGE 80
DVKWA.P11       T54     *TEST THE CLOCK'S 1MHZ DIVIDER

 3056  010166  005300                        DEC     R0              ;/WHAT WE WANT TO CHECK
 3057  010170  001373                        BNE     4$              ;/1MHZ PULSE ON 9 OSC PULSES.
 3058
 3059
 3060  010172  017746  171200                MOV     @ASR,-(6)       ;/SAVE CSR
 3061  010176  011637  001424                MOV     (6),$TMP3       ;/GET CSR.
 3062  010202  042737  177707  001424        BIC     #177707,$TMP3   ;/SAVE RATE BITS.
 3063  010210  052737  004005  001424        BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
 3064  010216  013777  001424  171152        MOV     $TMP3,@ASR              ;/LOAD CSR.
 3065                                                                ;/THIS MUST BE DONE IN
 3066                                                                ;/ORDER TO XFERR COUNTER
 3067                                                                ;/TO BUFFER ON ST2.
 3068  010224  052777  001000  171144        BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
 3069  010232  017737  171142  001126        MOV     @ASR,$BDDAT     ;/READ THE PRESET BUFFER,
 3070                                                                ;/PREVIOUS COUNTER
 3071  010240  012677  171132                MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
 3072  010244  005737  001126                TST     $BDDAT          ;/RESTORE CSR
 3073  010250  023737  001124  001126        CMP     $GDDAT,$BDDAT   ;/WAS ANOTHER 1MHZ PULSE GENERATED?
 3074  010256  001401                        BEQ     5$              ;/NO - NEXT TEST.
 3075
                                ;;SSSSSSSSSSSSSSSSSSSSSSS>>> ERROR <<<SSSSSSSSSSSSSSSSSSSSSSSSSSSSS

 3076  010260  104011                        ERROR   11              ;/WE SEEM TO HAVE GENERATED
 3077                                                                ;/ANOTHER 1MHZ PULSE ON
 3078                                                                ;/ONLY 9 MAINTENANCE
 3079                                                                ;/OSC PULSES.
 3080
                                ;;SSSSSSSSSSSSSSSSSSSSSSS>>> ERROR <<<SSSSSSSSSSSSSSSSSSSSSSSSSSSSS

 3081  010262  005077  171110        5$:     CLR     @ASR            ;/CLEAR THE CSR.
 3082
 3083
 3084                                ;;**************************************************************
                                     ;*TEST 55        *TEST THE CLOCK'S 100KHZ DIVIDER
                                     ;;**************************************************************
 3085  010266  000004                TST55:  SCOPE
 3086  010270  012737  000005  001160        MOV     #5,$TIMES       ;;DO 5 ITERATIONS
 3087
 3088  010276  005077  171074                CLR     @ASR            ;/CLEAR THE CSR.
 3089  010302  005077  171072                CLR     @#BR            ;/CLEAR THE PRESET BUFFER.
 3090  010306  052777  004000  171062        BIS     #BIT11,@ASR     ;/DISABLE THE INTERNAL OSC.
 3091  010314  052777  000021  171054        BIS     #1!20,@ASR      ;/ENABLE CLOCK, RATE:100KHZ
 3092
 3093
 3094  010322  012700  177634        10$:    MOV     #-100.,R0       ;/SET TO GENERATE 100 OSC PULSES.
 3095
 3096  010326  052777  002000  171042  1$:     BIS     #BIT10,@ASR     ;/GENERATE ONE OSC PULSE.
 3097  010334  005200                        INC     R0              ;/DONE 100 OSC PULSES?
 3098  010336  001373                        BNE     1$              ;/NO - DO ANOTHER ONE.
 3099
 3100
 3101  010340  012737  000001  001124  2$:     MOV     #1,$GDDAT       ;/SET FOR ERROR TYPEOUT - IF ANY.
 3102  010346  017746  171024                MOV     @ASR,-(6)       ;/SAVE CSR
 3103  010352  011637  001424                MOV     (6),$TMP3       ;/GET CSR.
 3104  010356  042737  177707  001424        BIC     #177707,$TMP3   ;/SAVE RATE BITS.
```

```
010364  052737  004005  001424          BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
010372  013777  001424  170776          MOV     $TMP3,@ASR              ;/LOAD CSR.
                                                                        ;/THIS MUST BE DONE IN
                                                                        ;/ORDER TO XFERR COUNTER
                                                                        ;/TO BUFFER ON ST2.
010400  052777  001000  170770          BIS     #BIT9,@ASR              ;/GENERATE ON ST2 PULSE
010406  017737  170766  001126          MOV     @ABR,$BDDAT             ;/READ THE PRESET BUFFER,
                                                                        ;/PREVIOUS COUNTER
010414  012677  170756                  MOV     (6)+,@ASR               ;/CONTENTS ARE IN $BDDAT.
010420  005737  001126                  TST     $BDDAT                  ;/RESTORE CSR
010424  013737  001124  001420          MOV     $GDDAT,$TMP0            ;/$TMP0 USED IN ERROR TYPEOUT.

010432  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;/DID CLOCK ADVANCE ONCE?


010440  001402                          BEQ     3$                      ;/YES   NEXT TEST.

                ;;$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

010442  104011                          ERROR   11                      ;/ERROR ON CLOCK100KHZ PULSE
                                                                        ;/NOT GENERATED WHEN 100
                                                                        ;/OSC PULSES GENERATED.

                ;;$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

010444  000443                          BR      5$
010446  012700  000143          3$:     MOV     #99.,R0 ;/GET THE NUMBER OF MORE OSC PULSES
                                                                        ;/TO BE GENERATED.

010452  052777  002000  170716  4$:     BIS     #BIT10,@ASR             ;/GENERATE ANOTHER OSC PULSE.
010460  005300                          DEC     R0                      ;/WHAT WE WANT TO CHECK
010462  001373                          BNE     4$                      ;/100KHZ PULSE ON 99 OSC PULSES.


010464  017746  170706                  MOV     @ASR,-(6)               ;/SAVE CSR
010470  011637  001424                  MOV     (6),$TMP3               ;/GET CSR.
010474  042737  177707  001424          BIC     #177707,$TMP3           ;/SAVE RATE BITS.
010502  052737  004005  001424          BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
010510  013777  001424  170660          MOV     $TMP3,@ASR              ;/LOAD CSR.
                                                                        ;/THIS MUST BE DONE IN
                                                                        ;/ORDER TO XFERR COUNTER
                                                                        ;/TO BUFFER ON ST2.
010516  052777  001000  170652          BIS     #BIT9,@ASR              ;/GENERATE ON ST2 PULSE
010524  017737  170650  001126          MOV     @ABR,$BDDAT             ;/READ THE PRESET BUFFER,
                                                                        ;/PREVIOUS COUNTER
010532  012677  170640                  MOV     (6)+,@ASR               ;/CONTENTS ARE IN $BDDAT.
010536  005737  001126                  TST     $BDDAT                  ;/RESTORE CSR
010542  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;/WAS ANOTHER 100KHZ PULSE GENERATED?
010550  001401                          BEQ     5$                      ;/NO - NEXT TEST.

                ;;$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

010552  104011                          ERROR   11                      ;/WE SEEM TO HAVE GENERATED
                                                                        ;/ANOTHER 100KHZ PULSE ON
                                                                        ;/ONLY 99 MAINTENANCE
                                                                        ;/OSC PULSES.
```

```
3178
                                        ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

3181  010554  005077  170616      5$:   CLR   @#SR             ;/CLEAR THE CSR.
3182
3183
3184                                  ;;**********************************************************************
3185                                  ;*TEST 56        *TEST THE CLOCK'S 10KHZ DIVIDER
3186                                  ;;**********************************************************************
3187  010560  000004                  TST56:  SCOPE
3188  010562  012737  000005  001160          MOV   #5,$TIMES         ;;DO 5 ITERATIONS
3189
3190  010570  005077  170602          CLR   @#SR              ;/CLEAR THE CSR.
3191  010574  005077  170600          CLR   @#BR              ;/CLEAR THE PRESET BUFFER.
3192  010600  052777  004000  170570  BIS   #BIT11,@#SR       ;/DISABLE THE INTERNAL OSC.
3193  010606  052777  000031  170562  BIS   #1!30,@#SR        ;/ENABLE CLOCK, RATE:10KHZ
3194
3195
3196
3197  010614  012700  176030      10$:  MOV   #-1000.,R0        ;/SET TO GENERATE 1000 OSC PULSES.
3198
3199  010620  052777  002000  170550  1$:   BIS   #BIT10,@#SR       ;/GENERATE ONE OSC PULSE.
3200  010626  005200                  INC   R0                ;/DONE 1000 OSC PULSES?
3201  010630  001373                  BNE   1$                ;/NO - DO ANOTHER ONE.
3202
3203
3204  010632  012737  000001  001124  2$:   MOV   #1,$GOOAT         ;/SET FOR ERROR TYPEOUT - IF ANY.
3205  010640  017746  170532          MOV   @#SR,-(6)         ;/SAVE CSR
3206  010644  011637  001424          MOV   (6),$TMP3         ;/GET CSR.
3207  010650  042737  177707  001424  BIC   #177707,$TMP3     ;/SAVE RATE BITS.
3208  010656  052737  004005  001424  BIS   #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
3209  010664  013777  001424  170504  MOV   $TMP3,@#SR                ;/LOAD CSR.
3210                                  ;/THIS MUST BE DONE IN
3211                                  ;/ORDER TO XFERR COUNTER
3212                                  ;/TO BUFFER ON ST2.
3213  010672  052777  001000  170476  BIS   #BIT9,@#SR        ;/GENERATE ON ST2 PULSE
3214  010700  017737  170474  001126  MOV   @#BR,$BOOAT       ;/READ THE PRESET BUFFER,
3215                                  ;/PREVIOUS COUNTER
3216  010706  012677  170464          MOV   (6)+,@#SR         ;/CONTENTS ARE IN $BOOAT.
3217  010712  005737  001126          TST   $BOOAT            ;/RESTORE CSR
3218  010716  013737  001124  001420  MOV   $GOOAT,$TMP0      ;/$TMP0 USED IN ERROR TYPEOUT.
3219
3220  010724  023737  001124  001126  CMP   $GOOAT,$BOOAT     ;/DID CLOCK ADVANCE ONCE?
3221
3222
3223  010732  001402                  BEQ   3$                ;/YES - NEXT TEST.
3224
                                        ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

3227  010734  104011                  ERROR 11                ;/ERROR ON CLOCK10KHZ PULSE
3228                                  ;/NOT GENERATED WHEN 1000
3229                                  ;/OSC PULSES GENERATED.
3230
                                        ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

3233  010736  000443                  BR    5$
```

```
3234  010740  012700  001747          3$:   MOV   #999.,R0        ;/GET THE NUMBER OF MORE OSC PULSES
3235                                                              ;/TO BE GENERATED.
3236
3237  010744  052777  002000  170424  4$:   BIS   #BIT10,@ASR     ;/GENERATE ANOTHER OSC PULSE.
3238  010752  005300                        DEC   R0              ;/WHAT WE WANT TO CHECK
3239  010754  001373                        BNE   4$              ;/10KHZ PULSE ON 999 OSC PULSES.
3240
3241
3242  010756  017746  170414                MOV   @ASR,-(6)       ;/SAVE CSR
3243  010762  011637  001424                MOV   (6),$TMP3       ;/GET CSR.
3244  010766  042737  177707  001424        BIC   #177707,$TMP3   ;/SAVE RATE BITS.
3245  010774  052737  004005  001424        BIS   #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
3246  011002  013777  001424  170366        MOV   $TMP3,@ASR              ;/LOAD CSR.
3247                                                              ;/THIS MUST BE DONE IN
3248                                                              ;/ORDER TO XFERR COUNTER
3249                                                              ;/TO BUFFER ON ST2.
3250  011010  052777  001000  170360        BIS   #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
3251  011016  017737  170356  001126        MOV   @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
3252                                                              ;/PREVIOUS COUNTER
3253  011024  012677  170346                MOV   (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
3254  011030  005737  001126                TST   $BDDAT          ;/RESTORE CSR
3255  011034  023737  001124  001126        CMP   $GDDAT,$BDDAT   ;/WAS ANOTHER 10KHZ PULSE GENERATED?
3256  011042  001401                        BEQ   5$              ;/NO - NEXT TEST.
3257

                                    ;;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

3260  011044  104011                        ERROR  11             ;/WE SEEM TO HAVE GENERATED
3261                                                              ;/ANOTHER 10KHZ PULSE ON
3262                                                              ;/ONLY 999 MAINTENANCE
3263                                                              ;/OSC PULSES.
3264

                                    ;;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

3267  011046  005077  170324          5$:   CLR   @ASR              ;/CLEAR THE CSR.
3268
3269
3270
3271                                ;;*****************************************************************
3272                                ;*TEST 57       *TEST THE CLOCK'S 1KHZ DIVIDER
3273                                ;;*****************************************************************
3274  011052  000004                ST57:  SCOPE
3275  011054  012737  000005  001160        MOV   #5,$TIMES       ;;DO 5 ITERATIONS
3276
3277  011062  005077  170310                CLR   @ASR            ;/CLEAR THE CSR.
3278  011066  005077  170306                CLR   @ABR            ;/CLEAR THE PRESET BUFFER.
3279  011072  052777  004000  170276        BIS   #BIT11,@ASR     ;/DISABLE THE INTERNAL OSC.
3280  011100  052777  000041  170270        BIS   #1!40,@ASR      ;/ENABLE CLOCK, RATE:1KHZ
3281
3282
3283
3284  011106  012700  154360          10$:  MOV   #-10000.,R0     ;/SET TO GENERATE 10000 OSC PULSES.
3285
3286  011112  052777  002000  170256  1$:   BIS   #BIT10,@ASR     ;/GENERATE ONE OSC PULSE.
3287  011120  005200                        INC   R0              ;/DONE 10000 OSC PULSES?
3288  011122  001373                        BNE   1$              ;/NO - DO ANOTHER ONE.
3289
```

```
3290
3291  011124  012737  000001  001124  2$:     MOV     #1,$GDDAT       ;/SET FOR ERROR TYPEOUT - IF ANY.
3292  011132  017746  170240                  MOV     @ASR,-(6)       ;/SAVE CSR
3293  011136  011637  001424                  MOV     (6),$TMP3       ;/GET CSR.
3294  011142  042737  177707  001424          BIC     #177707,$TMP3   ;/SAVE RATE BITS.
3295  011150  052737  004005  001424          BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
3296  011156  013777  001424  170212          MOV     $TMP3,@ASR              ;/LOAD CSR.
3297                                                                  ;/THIS MUST BE DONE IN
3298                                                                  ;/ORDER TO XFERR COUNTER
3299                                                                  ;/TO BUFFER ON ST2.
3300  011164  052777  001000  170204          BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
3301  011172  017737  170202  001126          MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
3302                                                                  ;/PREVIOUS COUNTER
3303  011200  012677  170172                  MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
3304  011204  005737  001126                  TST     $BDDAT          ;/RESTORE CSR
3305  011210  013737  001124  001420          MOV     $GDDAT,$TMP0    ;/$TMP0 USED IN ERROR TYPEOUT.
3306
3307  011216  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;/DID CLOCK ADVANCE ONCE?
3308
3309
3310  011224  001402                          BEQ     3$              ;/YES - NEXT TEST.
3311

                                ;;SSSSSSSSSSSSSSSSSSSSSS))) ERROR (((SSSSSSSSSSSSSSSSSSSSSSSSSSSSS

3314  011226  104011                          ERROR   11              ;/ERROR ON CLOCK1KHZ PULSE
3315                                                                  ;/NOT GENERATED WHEN 10000
3316                                                                  ;/OSC PULSES GENERATED.
3317

                                ;;SSSSSSSSSSSSSSSSSSSSSS))) ERROR (((SSSSSSSSSSSSSSSSSSSSSSSSSSSSS

3320  011230  000443                          BR      5$
3321  011232  012700  023417          3$:     MOV     #9999.,R0       ;/GET THE NUMBER OF MORE OSC PULSES
3322                                                                  ;/TO BE GENERATED.
3323
3324  011236  052777  002000  170132  4$:     BIS     #BIT10,@ASR     ;/GENERATE ANOTHER OSC PULSE.
3325  011244  005300                          DEC     R0              ;/WHAT WE WANT TO CHECK
3326  011246  001373                          BNE     4$              ;/1KHZ PULSE ON 9999 OSC PULSES.
3327
3328
3329  011250  017746  170122                  MOV     @ASR,-(6)       ;/SAVE CSR
3330  011254  011637  001424                  MOV     (6),$TMP3       ;/GET CSR.
3331  011260  042737  177707  001424          BIC     #177707,$TMP3   ;/SAVE RATE BITS.
3332  011266  052737  004005  001424          BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
3333  011274  013777  001424  170074          MOV     $TMP3,@ASR              ;/LOAD CSR.
3334                                                                  ;/THIS MUST BE DONE IN
3335                                                                  ;/ORDER TO XFERR COUNTER
3336                                                                  ;/TO BUFFER ON ST2.
3337  011302  052777  001000  170066          BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
3338  011310  017737  170064  001126          MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
3339                                                                  ;/PREVIOUS COUNTER
3340  011316  012677  170054                  MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
3341  011322  005737  001126                  TST     $BDDAT          ;/RESTORE CSR
3342  011326  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;/WAS ANOTHER 1KHZ PULSE GENERATED?
3343  011334  001401                          BEQ     5$              ;/NO - NEXT TEST.
3344

                                ;;SSSSSSSSSSSSSSSSSSSSSS))) ERROR (((SSSSSSSSSSSSSSSSSSSSSSSSSSSSS
```

```
3347  011336  104011                          ERROR   11              ;/WE SEEM TO HAVE GENERATED
3348                                                                  ;/ANOTHER 1KHZ PULSE ON
3349                                                                  ;/ONLY 9999 MAINTENANCE
3350                                                                  ;/OSC PULSES.
3351
                                        ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSS

3354  011340  005077  170032          5$:     CLR     @ASR            ;/CLEAR THE CSR.
3355
3356
3357                                    ;;**************************************************************
3358                                    ;*TEST 60        *TEST THE CLOCK'S 100HZ DIVIDER
3359                                    ;;**************************************************************
3360  011344  000004                    TST60:  SCOPE
3361  011346  012737  000005  001160            MOV     #5,$TIMES       ;;DO 5 ITERATIONS
3362
3363  011354  005077  170016                    CLR     @ASR            ;/CLEAR THE CSR.
3364  011360  005077  170014                    CLR     @ASR            ;/CLEAR THE PRESET BUFFER.
3365  011364  052777  004000  170004            BIS     #BIT11,@ASR     ;/DISABLE THE INTERNAL OSC.
3366  011372  052777  000051  167776            BIS     #1!50,@ASR      ;/ENABLE CLOCK, RATE:100HZ
3367
3368
3369  011400  012701  000012                    MOV     #10.,R1
3370
3371  011404  012700  154360          10$:    MOV     #-10000.,R0     ;/SET TO GENERATE 10000 OSC PULSES.
3372
3373  011410  052777  002000  167760  1$:     BIS     #BIT10,@ASR     ;/GENERATE ONE OSC PULSE.
3374  011416  005200                    INC     R0              ;/DONE 10000 OSC PULSES?
3375  011420  001373                    BNE     1$              ;/NO - DO ANOTHER ONE.
3376
3377  011422  005301                    DEC     R1
3378  011424  001367                    BNE     10$
3379
3380  011426  013737  000001  001124  2$:     MOV     #1,$GDDAT       ;/SET FOR ERROR TYPEOUT - IF ANY.
3381  011434  017746  167736                    MOV     @ASR,-(6)       ;/SAVE CSR
3382  011440  011637  001424                    MOV     (6),$TMP3       ;/GET CSR.
3383  011444  042737  177707  001424            BIC     #177707,$TMP3   ;/SAVE RATE BITS.
3384  011452  052737  004005  001424            BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
3385  011460  013777  001424  167710            MOV     $TMP3,@ASR              ;/LOAD CSR.
3386                                                                  ;/THIS MUST BE DONE IN
3387                                                                  ;/ORDER TO XFERR COUNTER
3388                                                                  ;/TO BUFFER ON ST2.
3389  011466  052777  001000  167702            BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
3390  011474  017737  167700  001126            MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
3391                                                                  ;/PREVIOUS COUNTER
3392  011502  012677  167670                    MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
3393  011506  005737  001126                    TST     $BDDAT          ;/RESTORE CSR
3394  011512  013737  001124  001420            MOV     $GDDAT,$TMP0    ;/$TMP0 USED IN ERROR TYPEOUT.
3395
3396  011520  023737  001124  001126            CMP     $GDDAT,$BDDAT   ;/DID CLOCK ADVANCE ONCE?
3397
3398  011526  001402                    BEQ     66$             ;/YES - NEXT TEST
3399
3400
                                        ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSS
```

```
3403  011530  104011                          ERROR   11              ;/ERROR ON CLOCK100HZ PULSE
3404                                                                  ;/NOT GENERATED WHEN 10000
3405                                                                  ;/OSC PULSES GENERATED.
3406

                                        ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSS

3409  011532  000447                          BR      5$
3410  011534  012701  000012          66$:    MOV     #10.,R1
3411  011540  012700  023417          3$:     MOV     #9999.,R0       ;/GET THE NUMBER OF MORE OSC PULSES
3412                                                                  ;/TO BE GENERATED.
3413
3414  011544  052777  002000  167624  4$:     BIS     #BIT10,@ASR     ;/GENERATE ANOTHER OSC PULSE.
3415  011552  005300                          DEC     R0              ;/WHAT WE WANT TO CHECK
3416  011554  001373                          BNE     4$              ;/100HZ PULSE ON 9999 OSC PULSES.
3417  011556  005301                          DEC     R1
3418  011560  001367                          BNE     3$
3419
3420
3421  011562  017746  167610                  MOV     @ASR,-(6)       ;/SAVE CSR
3422  011566  011637  001424                  MOV     (6),$TMP3       ;/GET CSR.
3423  011572  042737  177707  001424          BIC     #177707,$TMP3   ;/SAVE RATE BITS.
3424  011600  052737  004005  001424          BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
3425  011606  013777  001424  167562          MOV     $TMP3,@ASR              ;/LOAD CSR.
3426                                                                  ;/THIS MUST BE DONE IN
3427                                                                  ;/ORDER TO XFERR COUNTER
3428                                                                  ;/TO BUFFER ON ST2.
3429  011614  052777  001000  167554          BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
3430  011622  017737  167552  001126          MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
3431                                                                  ;/PREVIOUS COUNTER
3432  011630  012677  167542                  MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
3433  011634  005737  001126                  TST     $BDDAT          ;/RESTORE CSR
3434  011640  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;/WAS ANOTHER 100HZ PULSE GENERATED?
3435  011646  001401                          BEQ     5$              ;/NO - NEXT TEST.
3436

                                        ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSS

3439  011650  104011                          ERROR   11              ;/WE SEEM TO HAVE GENERATED
3440                                                                  ;/ANOTHER 100HZ PULSE ON
3441                                                                  ;/ONLY 9999 MAINTENANCE
3442                                                                  ;/OSC PULSES.
3443

                                        ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSS

3446  011652  005077  167520          5$:     CLR     @ASR            ;/CLEAR THE CSR.
3447
3448
3449
3450                                    ;;**********************************************************
3451                                    ;*TEST 61        *TES  THE CLOCK'S MODE 2 OPERATION
3452                                            ;*
3453                                            ;*IN THIS TEST WE'LL CHECK MODE 2 OPERATION
3454                                            ;*MODE 2: EXTERNAL EVENTS TIMING MODE
3455                                            ;*SETTING THE GO BIT CAUSES THE COUNTER TO BEGIN COUNTING FROM
3456                                            ;*ZERO AND TO FREE-RUN UNTIL THE GO BIT IS WRITTEN
3457                                            ;*TO A ZERO THE COUNTER WILL CONTINUE COUNTING AFTER
```

```
MAINDEC-11-DVKWA-A     MACY11 27(732)  04-OCT-76  14:57  PAGE 87
DVKWA.P11       T61    *TEST THE CLOCK'S MODE 2 OPERATION

                                                ;*OVERFLOW. AN EXTERNAL PULSE FROM SCHMITZ TRIGGER 2
                                                ;*(WHEN ST2 GO ENABLE IS A "0") CAUSES DATA TO
                                                ;*TRANSFER FROM THE COUNTER TO THE BUFFER/PRESET REG.
                                                ;*WHILE THE COUNTER CONTINUES TO RUN.
                                                ;*
                                                ;*TO TEST THIS MODE, WE'LL DISABLE THE INTERNAL OSC AND USE
                                                ;*MAINTENANCE OSC PULSES AS WELL AS A MAINTENANCE
                                                ;*ST2.
                                                ;*
                                        ;;**************************************************************
011656  000004                        ST61:   SCOPE
011660  012737  000020  001160                MOV     #20,$TIMES      ;;DO 20 ITERATIONS

011666  005077  167504                        CLR     @#ASR           ;CLEAR THE CSR.
011672  005077  167502                        CLR     @#BR            ;CLEAR THE PRESET REG.
011676  012777  004015  167472                MOV     #004015,@#ASR   ;START CLOCK.

011704  012700  177754                1$:     MOV     #-20.,R0        ;SET TO GIVE 20 MAINTENANCE OSC.
011710  052777  002000  167460        2$:     BIS     #BIT10,@#ASR    ;GENERATE A MAINTENANCE OSC.
011716  005200                        2$:     INC     R0
011720  001373                                BNE     2$              ;IF NOT DONE 20 TIMES, LOOP.

011722  052777  001000  167446        3$:     BIS     #BIT9,@#ASR     ;HERE'S THE BIGGIE! AN ST2 HAS BEEN GENERATED
011730  012737  000002  001124                MOV     #2,$GDDAT       ;THE PRESET BUFFER SHOULD BE 2.
011736  017737  167436  001126                MOV     @#BR,$BDDAT     ;READ THE PRESET BUFFER.
011744  023737  001126  001124                CMP     $BDDAT,$GDDAT   ;DID A COUNTER TO PRESET BUFFER OCCUR?
011752  001402                                BEQ     4$              ;YES - NEXT SUBTEST.

                                        ;;SSSSSSSSSSSSSSSSSSSSSS>>> ERROR <<<SSSSSSSSSSSSSSSSSSSSSSSSSSSS

011754  104005                                ERROR   5               ;A COUNTER TO PRESET BUFFER DID NOT
                                                                        ;HAPPEN PROPERLY.

                                        ;;SSSSSSSSSSSSSSSSSSSSSS>>> ERROR <<<SSSSSSSSSSSSSSSSSSSSSSSSSSSS

011756  000434                                BR      5$

011760                                  4$:
011760  017746  167412                        MOV     @#ASR,-(6)      ;/SAVE CSR
011764  011637  001424                        MOV     (6),$TMP3       ;/GET CSR.
011770  042737  177707  001424                BIC     #177707,$TMP3   ;/SAVE RATE BITS.
011776  052737  004005  001424                BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
012004  013777  001424  167364                MOV     $TMP3,@#ASR             ;/LOAD CSR.
                                                ;/THIS MUST BE DONE IN
                                                ;/ORDER TO XFERR COUNTER
                                                ;/TO BUFFER ON ST2.
012012  052777  001000  167356                BIS     #BIT9,@#ASR     ;/GENERATE ON ST2 PULSE
012020  017737  167354  001126                MOV     @#BR,$BDDAT     ;/READ THE PRESET BUFFER,
                                                ;/PREVIOUS COUNTER
012026  012677  167344                        MOV     (6)+,@#ASR      ;/CONTENTS ARE IN $BDDAT.
012032  005737  001126                        TST     $BDDAT          ;/RESTORE CSR
012036  023737  001124  001126                CMP     $GDDAT,$BDDAT   ;WAS THE COUNTER ACCIDENTLY ZEROED?
012044  001401                                BEQ     5$              ;NO - NEXT TEST.

                                        ;;SSSSSSSSSSSSSSSSSSSSSS>>> ERROR <<<SSSSSSSSSSSSSSSSSSSSSSSSSSSS
```

```
                                        ERROR   5               ;THE COUNT REGISTER SHOULD NOT
                                                                ;HAVE BEEN EFFECTED BY THE ST2
                                                                ;IN MODE 2.

                                ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSS

012050                          5$:


                                ;;**************************************************************
                                ;*TEST 62        *TEST THE CLOCK'S MODE 3 OPERATION
                                        ;*
                                        ;*IN THIS TEST WE'LL CHECK MODE 3 OPERATION.
                                        ;*MODE 3 IS JUST LIKE MODE 2 EXCEPT THAT THE COUNT
                                        ;*REG IS ZEROED AFTER AN ST2.
                                        ;*
                                ;;**************************************************************
012050  000004                  TST62:  SCOPE
012052  012737  000020  001160          MOV     #20,$TIMES      ;;DO 20 ITERATIONS

012060  005077  167312                  CLR     @#SR            ;CLEAR THE CSR.
012064  005077  167310                  CLR     @#BR            ;CLEAR THE BUFFER REG.
012070  012777  004017  167300          MOV     #4017,@#SR      ;START CLOCK.

012076  012700  177754          1$:     MOV     #-20.,R0        ;SET TO GIVE 20 MAINTENANCE OSC.
012102  052777  002000  167266  2$:     BIS     #BIT10,@#SR     ;GENERATE A MAINTENANCE OSC.
012110  005200                          INC     R0
012112  001373                          BNE     2$              ;IF NOT DONE 20 TIMES, LOOP.

012114  052777  001000  167254  3$:     BIS     #BIT9,@#SR      ;HERE'S THE BIGGIE! AN ST2 HAS BEEN GENERATED
012122  012737  000002  001124          MOV     #2,$GDDAT       ;THE PRESET BUFFER SHOULD BE 2.
012130  017737  167244  001126          MOV     @#BR,$BDDAT     ;READ THE PRESET BUFFER.
012136  023737  001126  001124          CMP     $BDDAT,$GDDAT   ;DID A COUNTER TO PRESET BUFFER OCCUR?
012144  001402                          BEQ     4$              ;YES - NEXT SUBTEST.

                                ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSS

012146  104005                          ERROR   5               ;A COUNTER TO PRESET BUFFER DID NOT
                                                                ;HAPPEN PROPERLY.

                                ;;SSSSSSSSSSSSSSSSSSSSSS››› ERROR ‹‹‹SSSSSSSSSSSSSSSSSSSSSSSSSSSSS

012150  000445                          BR      TST63           ;;

012152  005037  001124          4$:     CLR     $GDDAT          ;EXPECT ZERO BACK FROM COUNT REG.
012156  017746  167214                  MOV     @#SR,-(6)       ;/SAVE CSR
012162  011637  001424                  MOV     (6),$TMP3       ;/GET CSR.
012166  042737  177707  001424          BIC     #177707,$TMP3   ;/SAVE RATE BITS.
012174  052737  004005  001424          BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
012202  013777  001424  167166          MOV     $TMP3,@#SR              ;/LOAD CSR.
                                                                ;/THIS MUST BE DONE IN
                                                                ;/ORDER TO XFERR COUNTER
                                                                ;/TO BUFFER ON ST2.
012210  052777  001000  167160          BIS     #BIT9,@#SR      ;/GENERATE ON ST2 PULSE
012216  017737  167156  001126          MOV     @#BR,$BDDAT     ;/READ THE PRESET BUFFER,
                                                                ;/PREVIOUS COUNTER
```

```
012224  012677  167146                  MOV     (6)+,@ASR          ;/CONTENTS ARE IN $BDDAT.
012230  005737  001126                  TST     $BDDAT             ;/RESTORE CSR
012234  001402                          BEQ     5$                 ;IF SO - NEXT TEST.

                                ;;$$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

012236  104005                          ERROR   5                  ;THE CLOCK FORGOT TO ZERO THE COUNT
                                                                   ;REG. AFTER AN ST2 OCCURRED ON
                                                                   ;A MODE 3 COUNT.

                                ;;$$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

012240  000411                          BR      TST63              ;;
012242                          5$:
012242  005077  167130                  CLR     @ASR               ;NOW TRY CLEARING THE CSR.
012246  017737  167124  001126          MOV     @ASR,$BDDAT        ;READ THE CSR - DID IT CLEAR?
012254  001403                          BEQ     TST63              ;;
012256  005037  001124                  CLR     $GDDAT             ;NO - RECORD S/B.

                                ;;$$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

012262  104002                          ERROR   2                  ;CSR FAILED TO CLEAR

                                ;;$$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$


                                ;;***********************************************************************
                                ;*TEST 63       *IF ENABLED, CHECK ST1,ST2 IN FROM TESTOR
                                ;;***********************************************************************
012264  000004                  TST63:  SCOPE
012266  012737  000002  001160          MOV     #2,$TIMES          ;;DO 2 ITERATIONS

012274  005737  001440                  TST     EXS                ;OPERATING IN TESTOR MODE?
012300  001522                          BEQ     ENDP               ;NO-EXIT THIS TEST.

012302  005077  167070                  CLR     @ASR               ;CLEAR THE CSR
012306  012777  177775  167064          MOV     #-3,@ABR           ;SET TO COUNT THREE TIMES
012314  012777  000061  167054          MOV     #61,@ASR           ;SET RATE:ST1, MODE1, GO
012322  005777  166620                  TST     @STKB              ;CLEAR RDY FLG ON KEYBOARD
012326  104400  012334                  TYPE    ,65$               ;;TYPE ASCIZ STRING
012332  000433                          BR      64$                ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <15><12><7><7>**SWITCH ST1 3 TIMES, TYPE ANY KEY WHEN DONE..* 7><7>
012422                          64$:
012422  105777  166516          1$:     TSTB    @STKS              ;OPERATOR DONE IT?
012426  100375                          BPL     1$                 ;NO-WAIT.
012430  005777  166512                  TST     @STKB
012434  104400  012442                  TYPE    ,67$               ;;TYPE ASCIZ STRING
012440  000402                          BR      66$                ;;GET OVER THE ASCIZ
                                ;;67$:  .ASCIZ  <15><12>**
012446                          66$:
012446  005037  001124                  CLR     $GDDAT             ;EXPECT COUNTER TO BE CLEAR
012452  017746  166720                  MOV     @ASR,-(6)          ;/SAVE CSR
012456  011637  001424                  MOV     (6),$TMP3          ;/GET CSR.
012462  042737  177707  001424          BIC     #177707,$TMP3      ;/SAVE RATE BITS.
012470  052737  004005  001424          BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
012476  013777  001424  166672          MOV     $TMP3,@ASR              ;/LOAD CSR.
```

```
                                                                        ;/THIS MUST BE DONE IN
                                                                        ;/ORDER TO XFERR COUNTER
                                                                        ;/TO BUFFER ON ST2,
012504  052777  001000  166664          BIS     #BIT9,@ASR              ;/GENERATE ON ST2 PULSE
012512  017737  166662  001126          MOV     @ABR,$BDDAT             ;/READ THE PRESET BUFFER,
                                                                        ;/PREVIOUS COUNTER
012520  012677  166652                  MOV     (6)+,@ASR               ;/CONTENTS ARE IN $BDDAT.
012524  005737  001126                  TST     $BDDAT                  ;/RESTORE CSR
                                                                        ;OF NON-ZERO - REPORT ERROR
012530  001402                          BEQ     2$

                                ;;SSSSSSSSSSSSSSSSSSSSSS>>> ERROR <<<SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

012532  104002                          ERROR   2                       ;INCORRECT # OF ST1'S
                                                                        ;RECEIVED BY CLOCK

                                ;;SSSSSSSSSSSSSSSSSSSSSS>>> ERROR <<<SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

012534  000404                          BR      ENDP

012536                          2$:
012536  005777  166634                  TST     @ASR                    ;DID ST2 FLG SET?
012542  100401                          BMI     ENDP

                                ;;SSSSSSSSSSSSSSSSSSSSSS>>> ERROR <<<SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

012544  104006                          ERROR   6                       ;ST2 FLAG DID SET EVEN THOUGH
                                                                        ;SWITCH WAS TOGGLED.

                                ;;SSSSSSSSSSSSSSSSSSSSSS>>> ERROR <<<SSSSSSSSSSSSSSSSSSSSSSSSSSSSSS


                                        ;
                                        ;NOW WE'LL DETERMINE IF WE ARE ALLOWED TO AUTO-SIZE.
                                        ;IF SO, WE'LL FIND OUT IF THERE ARE OTHER CLOCKS OUT THERE
                                        ;TO TEST.
                                        ;

012546  000004          ENDP:   SCOPE

012550  105737  001215                  TSTB    $ENVM                   ;SEE IF APT WILL LET UP AUTO-SIZE.
012554  100537                          BMI     2$                      ;NO - EXIT.

012556  023737  001434  001436          CMP     MDEVCT,TSTCNT           ;TESTED MAX. UNITS?
012564  001507                          BEQ     4$                      ;YES EXIT.
012566  006337  001426                  ASL     ROTATE                  ;POINT NEXT UNIT.
012572  005237  001434                  INC     MDEVCT

012576  062737  000004  001376          ADD     #4,ASR                  ;YES, ADD T. BASE ADDR.
012604  013746  000004                  MOV     ERRVEC,-(6)             ;SAVE CONTENTS OF LOC 4.
012610  012737  012764  000004          MOV     #1$,ERRVEC              ;SET UP IN CASE NO MORE CLOCKS.

012616  005777  166554                  TST     @ASR                    ;TIME OUT HERE IF NO MORE CLOCKS.
```

```
                                                                ;IF HERE, ANOTHER CLOCK FOUND.
012622  005737  001202          TST     $PASS                 ;IS THIS 1ST PASS?
012626  001003                  BNE     3$                    ;NO-GET OUT.
012630  053737  001426  001430  BIS     ROTATE,UTEST          ;YES-RECORD THIS UNIT.
012636                  3$:
012636  104400  012644          TYPE    ,65$                  ;;TYPE ASCIZ STRING
012642  000405                  BR      64$                   ;;GET OVER THE ASCIZ
                        ;;65$:  .ASCIZ  <15><12>"UNIT #"
012656                  64$:
012656  013746  001204          MOV     $DEVCT,-(SP)          ;;SAVE $DEVCT FOR TYPEOUT
012662  104401                  TYPOC                         ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
012664  104400  012672          TYPE    ,67$                  ;;TYPE ASCIZ STRING
012670  000406                  BR      66$                   ;;GET OVER THE ASCIZ
                        ;;67$:  .ASCIZ  " COMPLETED "
012706                  66$:
012706  005237  001204          INC     $DEVCT
012712  104400  012720          TYPE    ,69$                  ;;TYPE ASCIZ STRING
012716  000410                  BR      68$                   ;;GET OVER THE ASCIZ
                        ;;69$:  .ASCIZ  " TESTING UNIT #"
012740                  68$:
012740  013746  001204          MOV     $DEVCT,-(SP)          ;;SAVE $DEVCT FOR TYPEOUT
012744  104401                  TYPOC                         ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
012746  012637  000004          MOV     (6)+,ERRVEC           ;RESETORE LOC 4.
012752  062737  000010  001402  ADD     #10,VECT1             ;UPDATE VECTOR ADDR.
012760  000137  002260          JMP     LOOP                  ;TEST NEW UNIT.

012764                  1$:
012764  062706  000004          ADD     #4,SP                         ;/ADD #4 TO STACK POINTER.
012770  012637  000004          MOV     (6)+,ERRVEC           ;RESTORE LOC 4
012774  022737  000001  001204  CMP     #1,$DEVCT             ;TESTED ONLY ONE UNIT?
013002  001424                  BEQ     2$                    ;YES - NO NEED FOR TYPEOUT.

013004                  4$:
013004  104400  013012          TYPE    ,71$                  ;;TYPE ASCIZ STRING
013010  000405                  BR      70$                   ;;GET OVER THE ASCIZ
                        ;;71$:  .ASCIZ  <15><12>"UNIT #"
013024                  70$:
013024  013746  001204          MOV     $DEVCT,-(SP)          ;;SAVE $DEVCT FOR TYPEOUT
013030  104401                  TYPOC                         ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
013032  104400  013040          TYPE    ,73$                  ;;TYPE ASCIZ STRING
013036  000406                  BR      72$                   ;;GET OVER THE ASCIZ
                        ;;73$:  .ASCIZ  " COMPLETED "
013054                  72$:

013054  013737  001250  001376  2$:     MOV     $BASE,ASR
013062  013737  001244  001402          MOV     $VECT1,VECT1
013070  012737  000001  001204          MOV     #1,$DEVCT

013076  005037  001434          CLR     MDEVCT                ;BEGIN TESTING 1ST UNIT.
013102  012737  000001  001426  MOV     #1,ROTATE             ;POINT TO IT.


                        .SBTTL  END OF PASS ROUTINE

                        ;;;;;;**********************************************************************
                        ;*INCREMENT THE PASS NUMBER ($PASS)
```

```
                                        ;*IF THERES A MONITOR GO TO IT
                                        ;*IF THERE ISN'T JUMP TO LOOP
                                        ;
013110                                  $EOP:
013110  000240                                  NOP
013112  005037  001102                          CLR     $TSTNM          ;;ZERO THE TEST NUMBER
013116  005037  001160                          CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
013122  005237  001202                          INC     $PASS           ;;INCREMENT THE PASS NUMBER
013126  042737  100000  001202                  BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
013134  005327                                  DEC     (PC)+           ;;LOOP?
013136  000001                          $EOPCT: .WORD   1
013140  003064                                  BGT     $DOAGN          ;;YES
013142  012737                                  MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
013144  000001                          $ENDCT: .WORD   1
013146  013136                                  $EOPCT
013150  104400  013156                          TYPE    ,65$            ;;TYPE ASCIZ STRING
013154  000406                                  BR      64$             ;;GET OVER THE ASCIZ
                                        ;;65$:  .ASCIZ  <15><12>#ENDPASS #
013172                                  64$:
013172  013746  001202                          MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
013176  104401                                  TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
013200  104400  013206                          TYPE    ,67$            ;;TYPE ASCIZ STRING
013204  000411                                  BR      66$             ;;GET OVER THE ASCIZ
                                        ;;67$:  .ASCIZ  #  TOTAL ERRORS  #
013230                                  66$:
013230  013746  001432                          MOV     ERCNT,-(SP)     ;;SAVE ERCNT FOR TYPEOUT
013234  104401                                  TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
013236  104400  013244                          TYPE    ,69$            ;;TYPE ASCIZ STRING
013242  000410                                  BR      68$             ;;GET OVER THE ASCIZ
                                        ;;69$:  .ASCIZ  #  GOOD UNITS  #
013264                                  68$:
013264  013746  001430                          MOV     UTEST,-(SP)     ;;SAVE UTEST FOR TYPEOUT
013270  104404                                  TYPBN                   ;;GO TYPE--BINARY ASCII
013272  013700  000042                  $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
013276  001405                                  BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
013300  000005                                  RESET                   ;;CLEAR THE WORLD
013302  004710                          $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
013304  000240                                  NOP                     ;;SAVE ROOM
013306  000240                                  NOP                     ;;FOR
013310  000240                                  NOP                     ;;ACT11
013312                                  $DOAGN:
013312  000137                                  JMP     @(PC)+          ;;RETURN
013314  002260                          $RTNAD: .WORD   LOOP
013316     377     377     000          $ENULL: .BYTE   -1,-1,0         ;;NULL CHARACTER STRING
013322                                          .EVEN
                                        ;
                                        ;
                                        .SBTTL          ;I/O SIGNAL TEST #1 ST1 IN AND ST2 OUT IN AND OUT
                                        ;
                                        ;
                                        ;SWITCH PACK S2 MUST BE SET UP AS FOLLOWS:
                                        ;
                                        ;       SWITCH  1 - OFF
                                        ;               2 - ON
```

```
                        ;I/O SIGNAL TEST #1 ST1 IN AND ST2 OUT IN AND OUT

                                                ;              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 - VV (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

013322  104406                  IOTST1: CKSWR                   ;CHECK THE SWR
013324  005077  166046          1$:     CLR     @#SR            ;CLEAR THE CSR
013330  005077  166044                  CLR     @#BR            ;CLEAR THE BUFFER REG.
013334  012777  000061  166034          MOV     #61,@#SR        ;RATE ST1, MODE 0, GO.
013342  052777  001000  166026          BIS     #BIT9,@#SR      ;GENERATE A MAINTENANCE ST2.
013350  012777  000005  166020          MOV     #5,@#SR         ;NOW SET TO READ COUNT REG
013356  052777  001000  166012          BIS     #BIT9,@#SR      ;FORCE COUNT -> BUFFER REG.
013364  027727  166010  000001          CMP     @#BR,#1         ;DID COUNT REG ADVANCE ONCE?
013372  001753                          BEQ     IOTST1          ;YES - LOOP.
013374  104000                          ERROR                   ;ST2 OUT TO ST1 IN FAILED.
013376  000751                          BR      IOTST1

                        ;
                        .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)
                                        ;
```

```
                                        ;LOAD AND START AT LOCATION 214.
                                        ;END PASSES OCCUR IMMEDIATELY AND ARE NOT REPORTED.
                                        ;EPRORS ARE REPORTED AS IN TH REGULAR LOGIC TEST AND
                                        ;THEIR PRINTOUT MAY BE INHIBITED.
                                        ;
013400  104406                  IOTST2: CKSWR                   ;CHECK THE SWR.
013402  005077  165770                  CLR     @#SR            ;CLEAR THE CSR.
013406  012777  177777  165764          MOV     #-1,@#BR        ;PRELOAD PRESET BUFFER.
013414  012777  000063  165754          MOV     #63,@#SR        ;RATE ST1, MODE 1, GO.
013422  052777  000400  165746          BIS     #BIT8,@#SR      ;GENERATE A MAIN. ST1.
013430  000240                          NOP
013432  000240                          NOP
013434  005777  165736                  TST     @#SR            ;DID OVERFLOW SET ST2 FLAG?
013440  100757                          BMI     IOTST2          ;YES - LOOP
013442  104000                          ERROR                   ;CLK OV OUT TO ST2 IN FAILED.
013444  000755                          BR      IOTST2          ;LOOP

                                        ;
                                .SBTTL  ;I/O SIGNAL TEST #3 ST1 OUT AND ST2 IN
                                        ;
                                        ;
                                        ;SWITCH PACK S2 MUST BE SET UP AS FOLLOWS:
                                        ;       SWITCH  1 - OFF
                                        ;               2 - OFF
                                        ;               3 - OFF
                                        ;               4 - ON
                                        ;               5 - ON
                                        ;               6 - ON
                                        ;               7 - NOT USED
                                        ; THIS SELECTS TTL THRESHOLD AND POSITIVE SLOPE FOR
                                        ; SCHMITT TRIGGER 2.
                                        ;
                                        ;PLEASE REMOVE ANY PREVIOUS JUMPERS.
                                        ;
                                        ;JUMPER THE FOLLOWING PINS TOGETHER:
                                        ;       J1 - UU (ST1 OUT)       TO J1 - TT (ST2-IN)
                                        ;
                                        ;LOAD AND START AT LOCATION 220
                                        ;END PASSES OCCUR IMMEDIATELY AND ARE NOT REPORTED
                                        ;ERRORS ARE REPORTED AS IN THE REGULAR LOGIC TEST AND
                                        ;THEIR PRINTOUT MAY BE INHIBITED
                                        ;
013446  104406                  IOTST3: CKSWR                   ;CHECK THE SWR
013450  012777  000001  165720          MOV     #1,@#SR         ;SET GO BIT.
013456  052777  000400  165712          BIS     #BIT8,@#SR      ;GENERATE A MAIN. ST1.
013464  005777  165706                  TST     @#SR            ;DID ST2 FLAG SET?
013470  100401                          BMI     1$
013472  104000                          ERROR                   ;ST1 OUT TO ST2 IN FAILED
                                        ;
013474  032777  010000  165674  1$:     BIT     #BIT12,@#SR     ;DID "FOR" BIT SET?
013502  001761                          BEQ     IOTST3          ;NO - GOOD!
013504  104000                          ERROR                   ;"FOR" BIT SET ON ONLY 1 ST2.
013506  000757                          BR      IOTST3          ;LOOP
                                        ;
                                .SBTTL
```

```
                                        .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.
                                ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
                                ;*CALL:
                                ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                                ;*      TYPOS                   ;;CALL FOR TYPEOUT
                                ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
                                ;*      .BYTE   M               ;;M=1 OR 0
                                ;*                                      ;;1=TYPE LEADING ZEROS
                                ;*                                      ;;0=SUPPRESS LEADING ZEROS
                                ;*
                                ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
                                ;*$TYPOS OR $TYPOC
                                ;*CALL:
                                ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                                ;*      TYPON                   ;;CALL FOR TYPEOUT
                                ;*
                                ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
                                ;*CALL:
                                ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                                ;*      TYPOC                   ;;CALL FOR TYPEOUT
                                
013510  017646  000000  STYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
013514  116637  000001  013733         MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
013522  112637  013735         MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
013526  062716  000002         ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
013532  000406                 BR      $TYPON
013534  112737  000001  013733 STYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
013542  112737  000006  013735         MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
013550  112737  000005  013732 STYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
013556  010346                 MOV     R3,-(SP)        ;;SAVE R3
013560  010446                 MOV     R4,-(SP)        ;;SAVE R4
013562  010546                 MOV     R5,-(SP)        ;;SAVE R5
013564  113704  013735         MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
013570  005404                 NEG     R4
013572  062704  000006         ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
013576  110437  013734         MOVB    R4,$OMCDE       ;;SAVE IT FOR USE
013602  113704  013733         MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
013606  016605  000012         MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
013612  005003                 CLR     R3              ;;CLEAR THE OUTPUT WORD
013614  006105         1$:     ROL     R5              ;;ROTATE MSB INTO "C"
013616  000404                 BR      3$              ;;GO DO MSB
013620  006105         2$:     ROL     R5              ;;FORM THIS DIGIT
013622  006105                 ROL     R5
013624  006105                 ROL     R5
013626  010503                 MOV     R5,R3
013630  006103         3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
013632  105337  013734         DECB    $OMODE          ;;TYPE THIS DIGIT?
013636  100016                 BPL     7$              ;;BR IF NO
013640  042703  177770         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$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
  3966  013654  052703  000060              5$:     BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
  3967  013660  052703  000040                      BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
  3968  013664  110337  013730                      MOVB    R3,8$           ;;SAVE FOR TYPING
  3969  013670  104400  013730                      TYPE    ,8$             ;;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  000744                        BR      2$              ;;GO DO THE LAST DIGIT
  3975  013710  012605                6$:     MOV     (SP)+,R5        ;;RESTORE R5
  3976  013712  012604                        MOV     (SP)+,R4        ;;RESTORE R4
  3977  013714  012603                        MOV     (SP)+,R3        ;;RESTORE R3
  3978  013716  016666  000002  000004        MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
  3979  013724  012616                        MOV     (SP)+,(SP)
  3980  013726  000002                        RTI                     ;;RETURN
  3981  013730     000                 8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
  3982  013731     000                         .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
  3983  013732     000                 $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
  3984  013733     000                 $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
  3985  013734  000000                 $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
  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                  $TYPBN: MOV     R1,-(SP)        ;;SAVE R1 ON THE STACK
  3996  013740  016601  000006                  MOV     6(SP),R1        ;;GET THE INPUT NUMBER
  3997  013744  000261                          SEC                     ;;SET "C" SO CAN KEEP TRACK OF THE NUMBER OF BITS
  3998  013746  112737  000060  014010  1$:     MOVB    #'0,$BIN        ;;SET CHARACTER TO AN ASCII "0".
  3999  013754  006101                          ROL     R1              ;;GET THIS BIT
  4000  013756  001406                          BEQ     2$              ;;DONE?
  4001  013760  105537  014010                  ADCB    $BIN            ;;NO--SET THE CHARACTER EQUAL TO THIS BIT
  4002  013764  104400  014010                  TYPE    ,$BIN           ;;GO TYPE THIS BIT
  4003  013770  000241                          CLC                     ;;CLEAR "C" SO CAN KEEP TRACK OF BITS
  4004  013772  000765                          BR      1$              ;;GO DO THE NEXT BIT
  4005  013774  012601                  2$:     MOV     (SP)+,R1        ;;POP THE STACK INTO R1
  4006  013776  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
  4007  014004  012616                          MOV     (SP)+,(SP)
  4008  014006  000002                          RTI                     ;;RETURN TO USER
  4009  014010     000     000          $BIN:   .BYTE   0,0             ;;STORAGE FOR ASCII CHAR. AND TERMINATOR
  4010
  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 $ERRTYP ON ERROR
```

```
                                        ;*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

                                        $ERROR:
014012                                          CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
014012  104406                          7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
014014  105237  001103                         BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
014020  001775                                  MOV     $TSTNM,@#DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
014022  013777  001102  165112                 BIT     #BIT10,@#SWR    ;;BELL ON ERROR?
014030  032777  002000  165102                 BEQ     1$              ;;NO - SKIP
014036  001402                                  TYPE    ,$BELL          ;;RING BELL
014040  104400  001164                 1$:     INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
014044  005237  001112                         MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
014050  011637  001116                         SUB     #2,$ERRPC
014054  162737  000002  001116                 MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
014062  117737  165030  001114                 BIT     #BIT13,@#SWR    ;;SKIP TYPEOUT IF SET
014070  032777  020000  165042                 BNE     20$             ;;SKIP TYPEOUTS
014076  001004                                  JSR     PC,$ERRTYP      ;;GO TO USER ERROR ROUTINE
014100  004737  014220                         TYPE    ,$CRLF
014104  104400  001171
014110                                  20$:
014110  122737  000001  001214                 CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
014116  001007                                  BNE     2$              ;;NO,SKIP APT ERROR REPORT
014120  113737  001114  014132                 MOVB    $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
014126  004737  015712                         JSR     PC,$ATY4        ;;REPORT FATAL ERROR TO APT
014132     000                  21$:    .BYTE   0
014133     000                          .BYTE   0
014134  000777                  22$:    BR      22$             ;;APT ERROR LOOP
014136  005777  164776           2$:    TST     @SWR            ;;HALT ON ERROR
014142  100002                          BPL     3$              ;;SKIP IF CONTINUE
014144  000000                          HALT                    ;;HALT ON ERROR!
014146  104406                          CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
014150  032777  001000  164762   3$:    BIT     #BIT09,@#SWR    ;;LOOP ON ERROR SWITCH SET?
014156  001402                          BEQ     4$              ;;BR IF NO
014160  013716  001110                  MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
014164  005737  001162           4$:    TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
014170  001402                          BEQ     5$              ;;BR IF NONE
014172  013716  001162                  MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
014176                           5$:

014176  005237  001432                  INC     ERCNT           ;/UPDATE ERROR COUNT.
014202  001002                          BNE     10$             ;/BUT DON'T LET IT OVERFLOW.
014204  005337  001432                  DEC     ERCNT           ;/KEEP AT 177777 IF OVERFLOW.
014210                           10$:
014210  043737  001426  001430         BIC     ROTATE,UTEST    ;/REMOVE UNIT FROM LIST OF GOOD ONES.
014216  000002                          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),
```

```
                                        ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
                                        ;
014220                                  $ERRTYP:
014220  104400  001171                        TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
014224  010046                                MOV     R0,-(SP)        ;;SAVE R0
014226  005000                                CLR     R0              ;;PICKUP THE ITEM INDEX
014230  153700  001114                        BISB    @#$ITEMB,R0
014234  001004                                BNE     1$              ;;IF ITEM NUMBER IS ZERO, JUST
                                                                        ;;TYPE THE PC OF THE ERROR
014236  013746  001116                        MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
                                                                        ;;ERROR ADDRESS
014242  104401                                TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
014244  000426                                BR      6$              ;;GET OUT
014246  005300                          1$:   DEC     R0              ;;ADJUST THE INDEX SO THAT IT WILL
014250  006300                                ASL     R0              ;;      WORK FOR THE ERROR TABLE
014252  006300                                ASL     R0
014254  006300                                ASL     R0
014256  062700  001256                        ADD     #$ERRTB,R0      ;;FORM TABLE POINTER
014262  012037  014272                        MOV     (R0)+,2$        ;;PICKUP "ERROR MESSAGE" POINTER
014266  001404                                BEQ     3$              ;;SKIP TYPEOUT IF NO POINTER
014270  104400                                TYPE                    ;;TYPE THE "ERROR MESSAGE"
014272  000000                          2$:   .WORD   0               ;;"ERROR MESSAGE" POINTER GOES HERE
014274  104400  001171                        TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
014300  012037  014310                  3$:   MOV     (R0)+,4$        ;;PICKUP "DATA HEADER" POINTER
014304  001404                                BEQ     5$              ;;SKIP TYPEOUT IF 0
014306  104400                                TYPE                    ;;TYPE THE "DATA HEADER"
014310  000000                          4$:   .WORD   0               ;;"DATA HEADER" POINTER GOES HERE
014312  104400  001171                        TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
014316  011000                          5$:   MOV     (R0),R0         ;;PICKUP "DATA TABLE" POINTER
014320  001004                                BNE     7$              ;;GO TYPE THE DATA
014322  012600                          6$:   MOV     (SP)+,R0        ;;RESTORE R0
014324  104400  001171                        TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
014330  000207                                RTS     PC              ;;RETURN
014332                                  7$:
014332  013046                                MOV     @(R0)+,-(SP)    ;;SAVE @(R0)+ FOR TYPEOUT
014334  104401                                TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
014336  005710                                TST     (R0)            ;;IS THERE ANOTHER NUMBER?
014340  001770                                BEQ     6$              ;;BR IF NO
014342  104400  014350                        TYPE    ,8$             ;;TYPE TWO(2) SPACES
014346  000771                                BR      7$              ;;LOOP
014350  020040  000               8$:   .ASCIZ  /  /                ;;TWO(2) SPACES
        014354                          .EVEN
                                .SBTTL  SCOPE HANDLER ROUTINE

                                ;;**************************************************************
                                ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
                                ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
                                ;*AND LOAD THE ERROR FLAG ($ERFLG) IN;O 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                          $SCOPE:
4131  014354  104406                          CKSWR                   ;;TEST  FOR CHANGE IN SOFT-SWR
4132  014356  104406                          CKSWR
4133  014360  032777  040000  164552  1$:     BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
4134  014366  001114                          BNE     $OVER           ;;YES IF SW14=1
4135                                  ;*****START OF CODE FOR THE XOR TESTER*****
4136  014370  000416                  $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
4137                                                                  ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
4138  014372  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
4139  014376  012737  014416  000004          MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
4140  014404  005737  177060                  TST     @#177060        ;;TIME OUT ON XOR?
4141  014410  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
4142  014414  000463                          BR      $SVLAD          ;;GO TO THE NEXT TEST
4143  014416  022626                  5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
4144  014420  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
4145  014424  000423                          BR      7$              ;;LOOP ON THE PRESENT TEST
4146  014426                          6$:;*****END OF CODE FOR THE XOR TESTER*****
4147  014426  032777  000400  164504          BIT     #BIT08,@SWR     ;;LOOP ON SPEC. TEST?
4148  014434  001404                          BEQ     2$              ;;BR IF NO
4149  014436  127737  164476  001102          CMPB    @SWR,$TSTNM     ;;ON THE RIGHT TEST?    SWR<7:0>
4150  014444  001465                          BEQ     $OVER           ;;BR IF YES
4151  014446  105737  001103          2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
4152  014452  001421                          BEQ     3$              ;;BR IF NO
4153  014454  123737  001115  001103          CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
4154  014462  101015                          BHI     3$              ;;BR IF NO
4155  014464  032777  001000  164446          BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
4156  014472  001404                          BEQ     4$              ;;BR IF NO
4157  014474  013737  001110  001106  7$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
4158  014502  000446                          BR      $OVER
4159  014504  105037  001103          4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
4160  014510  005037  001160                  CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
4161  014514  000415                          BR      1$              ;;ESCAPE TO THE NEXT TEST
4162  014516  032777  004000  164414  3$:     BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
4163  014524  001011                          BNE     1$              ;;BR IF YES
4164  014526  005737  001202                  TST     $PASS           ;;IF FIRST PASS OF PROGRAM
4165  014532  001406                          BEQ     :$              ;;      INHIBIT ITERATIONS
4166  014534  005237  001104                  INC     $ICNT           ;;INCREMENT ITERATION COUNT
4167  014540  023737  001160  001104          CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
4168  014546  002024                          BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
4169  014550  012737  000001  001104  1$:     MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
4170  014556  013737  014634  001160          MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
4171  014564  105237  001102          $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
4172  014570  113737  001102  001200          MOVB    $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
4173  014576  011637  001106                  MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
4174  014602  011637  001110                  MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
4175  014606  005037  001162                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
4176  014612  112737  000001  001115          MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
4177  014620  013777  001102  164314  $OVER:  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
4178  014626  013716  001106                  MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
4179  014632  000002                          RTI                     ;;FIXES PS
4180  014634  003720                  $MXCNT: 2000.                   ;;MAX. NUMBER OF ITERATIONS
4181                                          .SBTTL  TTY INPUT ROUTINE
4182
4183                                  ;;**************************************************************
4184                                          .ENABL  LSB
4185
```

```
4186                                    ;;*****************************************************************
4187                                    ;*SOFTWARE SWITCH REGISTER CHANGE ROLTINE.
4188                                    ;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
4189                                    ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
4190                                    ;*WHEN OPERATING IN TTY FLAG MODE.
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    @#TKS           ;;CHAR THERE?
4194  014652  100071                          BPL     15$             ;;IF NO, DON'T WAIT AROUND
4195  014654  117746  164266                  MOVB    @#TKB,-(SP)     ;;SAVE THE CHAR
4196  014660  042716  177600                  BIC     #^C177,(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  $GTSWR: TYPE    ,$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$:     TSTB    @#TKS           ;;CHAR THERE?
4210  014734  100375                          BPL     7$              ;;IF NOT TRY AGAIN
4211
4212  014736  117746  164204                  MOVB    @#TKB,-(SP)     ;;PICK UP CHAR
4213  014742  042716  177600                  BIC     #^C177,(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),@#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,@#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                                .DSABL  LSB
4252
4253
4254                                ;;*****************************************************************
4255                                ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
4256                                ;*CALL:
4257                                ;*      RDCHR                   ;;INPUT A SINGLE CHARACTER FROM THE TTY
4258                                ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
4259                                ;*                              ;;WITH PARITY BIT STRIPPED OFF
4260                                ;
4261
4262  015120  011646                $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
4263  015122  016666  000004  000002        MOV     4(SP),2(SP)     ;;SAVE THE PS
4264  015130  105777  164010        1$:     TSTB    @$TKS           ;;WAIT FOR
4265  015134  100375                        BPL     1$              ;;A CHARACTER
4266  015136  117766  164004  000004        MOVB    @$TKB,4(SP)     ;;READ THE TTY
4267  015144  042766  177600  000004        BIC     #^C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
4268  015152  026627  000004  000023        CMP     4(SP),#23       ;;IS IT A CONTROL-S?
4269  015160  001013                        BNE     3$              ;;BRANCH IF NO
4270  015162  105777  163756        2$:     TSTB    @$TKS           ;;WAIT FOR A CHARACTER
4271  015166  100375                        BPL     2$              ;;LOOP UNTIL ITS THERE
4272  015170  117746  163752                MOVB    @$TKB,-(SP)     ;;GET CHARACTER
4273  015174  042716  177600                BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
4274  015200  022627  000021                CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
4275  015204  001366                        BNE     2$              ;;IF NOT DISCARD IT
4276  015206  000750                        BR      1$              ;;YES, RESUME
4277  015210  026627  000004  000140  3$:   CMP     4(SP),#140      ;;IS IT UPPER CASE?
4278  015216  002407                        BLT     4$              ;;BRANCH IF YES
4279  015220  026627  000004  000175        CMP     4(SP),#175      ;;IS IT A SPECIAL CHAR?
4280  015226  003003                        BGT     4$              ;;BRANCH IF YES
4281  015230  042766  000040  000004        BIC     #40,4(SP)       ;;MAKE IT UPPER CASE
4282  015236  000002                4$:     RTI                     ;;GO BACK TO USER
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                                ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
4289
4290  015240  010346                $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
4291  015242  012703  015346        1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
4292  015246  022703  015356        2$:     CMP     #$TTYIN+8.,R3   ;;BUFFER FULL?
4293  015252  101405                        BLOS    4$              ;;BR IF YES
4294  015254  104407                        RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
4295  015256  112613                        MOVB    (SP)+,(R3)      ;;GET CHARACTER
4296  015260  122713  000177        10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
4297  015264  001003                        BNE     3$              ;;SKIP IF NOT
```

```
4298  015266  104400  001170          4$:     TYPE    $QUES           ;;TYPE A '?'
4299  015272  000763                  BR      1$              ;;CLEAR THE BUFFER AND LOOP
4300  015274  111337  015344          3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
4301  015300  104400  015344                  TYPE    9$
4302  015304  122723  000015                  CMPB    #15,(R3)+       ;;CHECK FOR RETURN
4303  015310  001356                  BNE     2$              ;;LOOP IF NOT RETURN
4304  015312  105063  177777                  CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
4305  015316  104400  001172                  TYPE    $LF             ;;TYPE A LINE FEED
4306  015322  012603                  MOV     (SP)+,R3        ;;RESTORE R3
4307  015324  011646                  MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
4308  015326  016666  000004  000002          MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
4309  015334  012766  015346  000004          MOV     #STTYIN,4(SP)
4310  015342  000002                  RTI                     ;;RETURN
4311  015344     000          9$:     .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    $TPFLG          ;;IS THERE A TERMINAL?
4338  015416  100002                  BPL     1$              ;;BR IF YES
4339  015420  000000                  HALT                    ;;HALT HERE IF NO TERMINAL
4340  015422  000430                  BR      3$              ;;LEAVE
4341  015424  010046          1$:     MOV     R0,-(SP)        ;;SAVE R0
4342  015426  017600  000002          MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
4343  015432  122737  000001  001214  CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
4344  015440  001011                  BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
4345  015442  132737  000100  001215  BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
4346  015450  001405                  BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
4347  015452  010037  015462          MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
4348  015456  004737  015702          JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
4349  015462  000000          61$:    .WORD   0               ;;MESSAGE ADDRESS
4350  015464  132737  000040  001215  62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
4351  015472  001003                  BNE     60$             ;;YES,SKIP TYPE OUT
4352  015474  112046          2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
4353  015476  001005                  BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
```

```
4354  015500  005726                        TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
4355  015502  012600                60$:    MOV     (SP)+,R0        ;;RESTORE R0
4356  015504  062716  000002        3$:     ADD     #2,(SP)          ;ADJUST RETURN PC
4357  015510  000002                        RTI                     ;;RETURN
4358  015512  122716  000011        4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
4359  015516  001430                        BEQ     8$
4360  015520  122716  000200                CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
4361  015524  001006                        BNE     5$
4362  015526  005726                        TST     (SP)+           ;;POP  <CR><LF> EQUIV
4363  015530  104400                        TYPE                    ;;TYPE A CR AND LF
4364  015532  001171                        $CRLF
4365  015534  105037  015670                CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
4366  015540  000755                        BR      2$              ;;GET NEXT CHARACTER
4367  015542  004737  015624        5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
4368  015546  123726  001156        6$:     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     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
4371                                                                ;;AND THE NULL CHAR.
4372  015560  105366  000001        7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
4373  015564  002770                        BLT     6$              ;;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
4377
4378                                ;HORIZONTAL TAB PROCESSOR
4379
4380  015600  112716  000040        8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
4381  015604  004737  015624        9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
4382  015610  132737  000007  015670        BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
4383  015616  001372                        BNE     9$              ;;TAB STOP
4384  015620  005726                        TST     (SP)+           ;;POP SPACE OFF STACK
4385  015622  000724                        BR      2$              ;;GET NEXT CHARACTER
4386  015624  105777  163320        $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
4387  015630  100375                        BPL     $TYPEC
4388  015632  116677  000002  163312        MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
4389  015640  122766  000015  000002        CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
4390  015646  001003                        BNE     1$              ;;BRANCH IF NO
4391  015650  105037  015670                CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
4392  015654  000406                        BR      $TYPEX          ;;EXIT
4393  015656  122766  000012  000002  1$:   CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
4394  015664  001402                        BEQ     $TYPEX          ;;BRANCH IF YES
4395  015666  105227                        INCB    (PC)+           ;;COUNT THE CHARACTER
4396  015670  000000        $CHARCNT:.WORD  0                       ;;CHARACTER COUNT STORAGE
4397  015672  000207        $TYPEX: RTS     PC
4398
4399                                .SBTTL  APT COMMUNICATIONS ROUTINE
4400
4401                                ;;**************************************************************
4402  015674  112737  000001  016140  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
4403  015702  112737  000001  016136  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
4404  015710  000403                        BR      $ATYC
4405  015712  112737  000001  016140  $ATY4:  MOVB    #1,$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     5$              ;;IF NOT:  BR
4411  015732  122737  000001  001214          CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
4412  015740  001031                          BNE     3$              ;;IF NOT:  BR
4413  015742  132737  000100  001215          BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
4414  015750  001425                          BEQ     3$              ;;IF NOT:  BR
4415  015752  017600  000004                  MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
4416  015756  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
4417  015764  005737  001174          1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
4418  015770  001375                          BNE     1$              ;;IF NOT:  WAIT
4419  015772  010037  001210                  MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
4420  015776  105720                  2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
4421  016000  001376                          BNE     2$
4422  016002  163700  001210                  SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
4423  016006  006200                          ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
4424  016010  010037  001212                  MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
4425  016014  012737  000004  001174          MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
4426  016022  000413                          BR      5$
4427  016024  017637  000004  016050  3$:     MOV     @4(SP),4$       ;;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,$TYPE        ;;CALL TYPE MACRO
4431  016050  000000                  4$:     .WORD   0
4432  016052                          5$:
4433  016052  105737  016140          10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
4434  016056  001416                          BEQ     12$             ;;IF NOT:  BR
4435  016060  005737  001214                  TST     $ENV            ;;RUNNING UNDER APT?
4436  016064  001413                          BEQ     12$             ;;IF NOT:  BR
4437  016066  005737  001174          11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
4438  016072  001375                          BNE     11$             ;;IF NOT:  WAIT
4439  016074  017637  000004  001176          MOV     @4(SP),$FATAL   ;;GET ERROR #
4440  016102  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
4441  016110  005237  001174                  INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
4442  016114  105037  016140          12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
4443  016120  105037  016137                  CLRB    $LFLG           ;;CLEAR LOG FLAG
4444  016124  105037  016136                  CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
4445  016130  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
4446  016132  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
4447  016134  000207                          RTS     PC              ;;RETURN
4448  016136     000          $MFLG:  .BYTE   0               ;;MESSG. FLAG
4449  016137     000          $LFLG:  .BYTE   0               ;;LOG FLAG
4450  016140     000          $FFLG:  .BYTE   0               ;;FATAL FLAG
4451             016142                  .EVEN
4452             000200          APTSIZE=200
4453             000001          APTENV=001
4454             000100          APTSPOOL=100
4455             000040          APTCSUP=040
4456                             .SBTTL  POWER DOWN AND UP ROUTINES
4457
4458                             ;;******************************************************************
4459                             ;POWER DOWN ROUTINE
4460  016142  012737  016302  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
4461  016150  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
4462  016156  010046                          MOV     R0,-(SP)        ;;PUSH R0 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
```

```
                                        MOV     R4,-(SP)        ;;PUSH R4 ON STACK
                                        MOV     R5,-(SP)        ;;PUSH R5 ON STACK
                                        MOV     @SWR,-(SP)      ;;PUSH @SWR ON STACK
                                        MOV     SP,$SAVR6       ;;SAVE SP
                                        MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
                                        HALT
                                        BR      .-2             ;;HANG UP

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

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

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

                                .SBTTL  TRAP TABLE

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

                                ;       ROUTINE
                                ;       -------
```

```
MAINDEC-11-DVXWA-A     MACY11 27(732)  04-OCT-76  14:57  PAGE 106
DVXWA.P11       TRAP TABLE

                                                $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
016342
016342  015412                                  EM1:    .ASCIZ  <15><12>/CLOCK SR FUNCTION ERROR/
016344  013534
016346  013510
016350  013550
016352  013736
016354  014706
016356  014636
016360  015120
016362  015240
016364  005015  046103  041517
016372  020113  051123  043040
016400  047125  052103  047511
016406  020116  051105  047522
016414  000122
016416  005015  046103  041517                EM2:    .ASCIZ  <15><12>/CLOCK SR DATA ERROR/
016424  020113  051123  042040
016432  052101  020101  051105
016440  047522  000122
016444  005015  046103  041517                EM3:    .ASCIZ  <15><12>/CLOCK BR DATA ERROR/
016452  020113  051102  042040
016460  052101  020101  051105
016466  047522  000122
016472  005015  047503  047125                EM5:    .ASCIZ  <15><12>/COUNT REG. ERROR/
016500  020124  042522  027107
016506  042440  051122  051117
016514     000
016515     015  041412  047514                EM11:   .ASCIZ  <15><12>#CLOCK COUNT ERROR #
016522  045503  041440  052517
016530  052116  042440  051122
016536  051117  000040
016542  005015  046103  041517                EM12:   .ASCIZ  <15><12>#CLOCK COUNT FUNCTION ERROR#
016550  020113  047503  047125
016556  020124  052506  041516
016564  044524  047117  042440
016572  051122  051117     000
016577     015  041412  047514                EM16:   .ASCIZ  <15><12>#CLOCK INTERRUPT ERROR #
016604  045503  044440  052116
016612  051105  052522  052120
016620  042440  051122  051117
016626  000040
016630  005015  046103  041517                EM20:   .ASCIZ  <15><12>#CLOCK REPEATABILITY ERROR #
016636  020113  042522  042520
016644  052101  041101  046111
016652  052111  020131  051105
016660  047522  020122     000
016665     015  041412  047514                EM26:   .ASCIZ  <15><12>#CLOCK ADDRESSING ERROR#
016672  045503  040440  042104
016700  042522  051523  047111
016706  020107  051105  047522
016714  000122

016716  005015  051105  050122                DH1:    .ASCIZ  <15><12>#ERRPC  ASR     WAS     S/B#
016724  004503  051501  004522
```

```
 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     2NDCNT  1STNCT  3RDCNT#
 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                                  .EVEN
 4603
 4604  017114  001116  001376  001126  DT1:    .WORD   $ERRPC,ASR,$BDDAT,$GDDAT,0
 4605  017122  001124  000000
 4606  017126  001116  001400  001126  DT3:    .WORD   $ERRPC,ABR,$BDDAT,$GDDAT,0
 4607  017134  001124  000000
 4608  017140  001116  001376  000000  DT12:   .WORD   $ERRPC,ASR,0
 4609  017146  001116  001376  001126  DT20:   .WORD   $ERRPC,ASR,$BDDAT,$GDDAT,STMP0,0
 4610  017154  001124  001420  000000
 4611  017162  001116  001376  001126  DT22:   .WORD   $ERRPC,ASR,$BDDAT,STMP0,0
 4612  017170  001420  000000
 4613  017174  001116  001420  000000  DT26:   .WORD   $ERRPC,STMP0,0
 4614
 4615  017202  000000  000000          DF0:    .WORD   0,0
 4616
 4617          000001                          .END
```

```
ABASE = 170420          839*   926    967   1069   107?
ABR     001400         1070*  1199*  1200*  1255   12?    1741*  1743*  1744   1759*  1761   1780*  1782*  1783
                       1798*  1800   1909*  1926   195?*  1971   2050*  2054   2139*  2152   2180*  2211   2228*
                       2246*  2283*  2308*  2342*  2358   2393*  2409   2444*  2460   2435*  2511   2546*  2562
                       2601*  2617   2652*  2664   2693*  2712   2738*  2776*  2880*  2936*  2953   2974*  2994
                       3019*  3042   3079   3105*  3128   3165   3191*  3214   3251   3278*  3301   3338   3364*
                       3390   3430   3472*  3482   3505   3535*  3545   3568   3606*  3630   3815*  3821   3857*
                       4606
ACDW1 = 000000          926    969
ACDW2 = 000000          926
ACPUOP= 000000          926    941
ADDW0 = 000000          926
ADDW1 = 000000          926
ADDW10= 000000          926
ADDW11= 000000          926
ADDW12= 000000          926
ADDW13= 000000          926
ADDW14= 000000          926
ADDW15= 000000          926
ADDW2 = 000000          926
ADDW3 = 000000          926
ADDW4 = 000000          926
ADDW5 = 000000          926
ADDW6 = 000000          926
ADDW7 = 000000          926
ADDW8 = 000000          926
ADDW9 = 000000          926
ADEVCT= 000000          926    932
ADEVM = 000000          926    968
ADEW  = 000000          926    937
ADEWM = 000000          926    938
AFATAL= 000000          926    929
AMADR1= 000000          926    954
AMADR2= 000000          926    958
AMADR3= 000000          926    961
AMADR4= 000000          926    964
AMAMS1= 000000          926    948
AMAMS2= 000000          926    956
AMAMS3= 000000          926    959
AMAMS4= 000000          926    962
AMSCAD= 000000          926    934
AMSGLG= 000000          926    935
AMSGTY= 000000          926    928
AMTYP1= 000000          926    949
AMTYP2= 000000          926    957
AMTYP3= 000000          926    960
AMTYP4= 000000          926    963
APASS = 000000          926    931
APRIOR= 000200          841*   926    1075
APTCSU= 000040         4350   4455*
APTENV= 000001         4043   4343   4411   4453*
APTSIZ= 000200         1138   4452*
APTSP0= 000100         4345   4413   4454*
ASR     001376         1069*  1150*  1199   1219   1227   1296*  1297*  1299   1312*  1314   1341*  1342*  1344
                       1357*  1359   1386*  1387*  1389   1402*  1404   1431*  1432*  1434   1447*  1449   1476*
                       1477*  1479   1492*  1494   1521*  1522*  1524   1537*  1539   1566*  1567*  1569   1582*
```

```
                    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    2993*   2993*   2996*   3008*   3018*   3020*   3021*   3027*   3033
                    3037*   3041*   3044*   3065*   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*
                    3896    3900    4604    4608    4609    4611
ASHREG= 000000      926     939
ATESTN= 000000      926     930
AUNIT = 000000      926     933
AUSWR = 000000      926     940
AVECT1= 000440      840*    926     965     1071    1072    1073    1074
AVECT2= 000000      926     966
BIT0  = 000001      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      813*    823     2114
BIT01 = 000002      812*    822
BIT02 = 000004      811*    821
BIT03 = 000010      810*    820
BIT04 = 000020      809*    819
BIT05 = 000040      808*    818
BIT06 = 000100      807*    817
BIT07 = 000200      806*    816
BIT08 = 000400      805*    815     4147
BIT09 = 001000      804*    814     4054    4155
BIT1  = 000002      822*    1657    1658    1672    2190
BIT10 = 002000      803*    2026    2070    2979    3027    3065    3113    3151    3199    3237    3286    3324    3373
                    3414    3476    3539    4031
BIT11 = 004000      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
BIT12 = 010000      801*    2026    2070    2595    2858    2862    2887    2917    3900
BIT13 = 020000      800*    1342    1343    1357    2112    4038
BIT14 = 040000      799*    1297    1298    1312    2818    4133
BIT15 = 100000      798*    2862
BIT2  = 000004      821*    1612    1613    1627    1920    1965    2095    2146    2205    2352    2403    2454    2505
```

F09


```
                        2556    2611    2658    2706    2947    2988    3036    3073    3122    3159    3208    3245    3295
                        3332    3384    3424    3499    3562    3624
BIT3   = 000010          820#    1567    1568    1582    2940    2976
BIT4   = 000020          819#    1522    1523    1537    2140    2184    2248    2285
BIT5   = 000040          818#    1477    1478    1492    2140    2184    2243    2248    2285
BIT6   = 000100          817#    1432    1433    1447
BIT7   = 000200          816#
BIT8   = 000400          815#    2141    2186    2250    2287    2311    2654    2697    2779    3859    3895
BIT9   = 001000          814#    1925    1970    2078    2113    2151    2210    2357    2408    2459    2510    2561    2616
                        2663    2711    2819    2855    2856    2907    2908    2952    2993    3041    3078    3127    3164
                        3213    3250    3300    3337    3389    3429    3480    3504    3543    3567    3629    3818    3820
BPTVEC= 000014           830#
CKSWR  = 104406         3814    3855    3893    4027    4053    4131    4132    4531#
CR     = 000015          738#    4389    4399
CRLF   = 000200          739#    1170    4360    4399
DOISP  = 177570          745#     907    1126
DF0    = 017202          994    1002    1010    1018    1026    1034    1042    1050    1058    1066    4615#
DH1    = 016716          992    1000    4576#
DH12   = 017002         1032    1040    4588#
DH20   = 017027         1048    4591#
DH26   = 017070         1064    4597#
DH3    = 016742         1008    4580#
DH4    = 016766         1024    1056    4584#
DISPLA = 001142          907#   1126*   1134*   4030*   4177*
DISPRE = 000174          707#   1134
DR     = 001414         1076#   2027    2095    2190
DR2    = 001416         1077#   2026*   2070#
DSWR   = 177570          744#    906    1125
DT1    = 017114          993    1001    1025    4604#
DT12   = 017140         1033    1041    4608#
DT20   = 017146         1049    4609#
DT22   = 017162         1057    4611#
DT26   = 017174         1065    4613#
DT3    = 017126         1009    4606#
EMTVEC= 000030            633#   1110*   1111*
EM1    = 016364           991    4534#
EM11   = 016515          1055    4551#
EM12   = 016542          1031    4555#
EM16   = 016577          1039    4560#
EM2    = 016416           999    4539#
EM20   = 016630          1047    4565#
EM26   = 016665          1063    4570#
EM3    = 016444          1007    4543#
EM5    = 016472          1023    4547#
ENDP   = 012546          3603    3644    3648    3664#
ERCNT  = 001432          1083#   1184*   3763    4062*   4064#
ERRVEC= 000004            826#   1123    1124*   1135*   1215    1216*   1242*   1251    1252*   1278*   3677    3678*   3704*
                        3710*   4138    4139*   4141*   4144*
EXS    = 001440         1086#   1089*   1094*   1098*   1172    2024    2068    2093    2188    2238    2261    3602
GMS    = ****** U         706    1169    1176    1181    3611    3618    3689    3695    3700    3717    3723    3755    3761
                        3767    4523    4524    4525    4526    4527    4529    4531    4532    4533
GTSWR  = 104405          1164    4529#
HT     = 000011           736#   4358    4399
IOTST1 = 013322           716    3814#   3822    3824
IOTST2 = 013400           717    3855#   3863    3865
IOTST3 = 013446           718    3893#   3901    3903
```

```
IOTVEC= 000020           831#  1108*  1109*
LF    = 000012           737#  4393   4399
LOOP    002260          1188#  3706   3780
MDEVCT  001434          1084#  1183*  3671   3674*  3730*
PC    =%000007           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#
PR0   = 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*
RDCHR = 104407          4294    4532#
RDLIN = 104410          4533#
RESVEC= 000010           827#
ROTATE  001426          1081#  1186*  1187   3673*  3685   3731*  4066
RSTART  002070           715   1171#
R0    =%000000           748#  2344*  2346*  2395*  2397*  2446*  2448*  2497*  2499*  2548*  2550*  2603*  2605*
                        2698*  2700*  2740*  2741*  2862*  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    =%000001           749#  3369*  3377*  3410*  3417*  3995   3996*  3999*  4005*  4408   4445*  4463   4486*
R2    =%000002           750#  4464   4485*
R3    =%000003           751#  3942   3951*  3957*  3958*  3961*  3966*  3967*  3968   3977*  4290   4291*  4292
                        4295*  4296   4300   4302   4304*  4306*  4465   4484*
R4    =%000004           752#  3943   3945*  3946*  3947*  3948   3949*  3963   3965*  3973*  3976*  4466   4483*
R5    =%000005           753#  3944   3950*  3952*  3954*  3955*  3956*  3957   3975*  4467   4482*
R6    =%000006           754#   756   1102*  1103*  1104
R7    =%000007           755#   757
SP    =%000006           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*
                        3943*  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
START = 001472           714   1096#
STKLMT= 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*
TRTVEC= 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          2592#
TST42   006476          2598    2647#
TST43   006626          2689#
TST44   006770          2735#
TST45   007030          2766#
TST46   007150          2828#
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#
TYPOC = 104401          3692    3703    3720    3758    3764    4085    4109    4205    4524#
TYPON = 104403          4526#
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#
$APTHD  001000           856     872#
$ASTAT= ****** U        443.    4448
$ATYC   015720          440.    4406#
$ATY1   015674          4402#
$ATY3   015702          4348    4403#
$ATY4   015712          4046    4405#
$AUTOB  001134           903#   1166*   4199    4320
$BASE   001250           967#   1150    3726
$BOARD  001122           898#
$BOOAT  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*
```

```
                    2155    2211*   2214    2216    2358*   2361    2409*   2412    2460*   2463    2511*   2514    2562*
                    2565    2617*   2620    2664*   2667    2712*   2715    2861*   2890*   2915*   2917    2953*   2956
                    2994*   2997    3042*   3045    3048    3079*   3082    3083    3128*   3131    3134    3165*   3168
                    3169    3214*   3217    3220    3251*   3254    3255    3301*   3304    3307    3338*   3341    3342
                    3390*   3393    3396    3430*   3433    3434    3482*   3483    3505*   3508    3509    3545*   3546
                    3568*   3571    3585*   3630*   3633    4604    4606    4609    4611
$BELL   001164       918#   4033    4069
$BIN    014010      3998*   4001*   4002    4009#
$CDWI   001254       969#
$CHRPC  015670      4365*   4375*   4382    4391*   4396#
$CKSWR  014636      4191#   4531
$CMTAG  001100       886#   1101    1102    1110    1116    1117
$CM3  =  000000       916#
$CNTLG  015363      4202    4315#
$CNTLU  015356      4219    4314#
$CPUOP  001222       941#
$CRLF   001171       920#   4041    4069    4077    4096    4101    4105    4230    4314    4364    4399
$DEVCT  001204       932#   3691    3697*   3702    3711    3719    3728*
$DEVM   001252       968#
$DOAGN  013312      3749    3772    3778#
$ENDAD  013302       852    3774#
$ENDCT  013144      3751#
$ENULL  013316      3781#
$ENV    001214       937#   1160    4043    4343    4411    4435
$ENVM   001215       938#   1138    3667    4345    4350    4413
$EOP    013110      3741#
$EOPCT  013136      3748#   3752
$ERFLG  001103       869#   4028*   4069    4121    4151    4153    4159*   4181
$ERMAX  001115       895#   1118*   4153    4176*   4181
$ERROR  014012      1110    4026#
$ERRPC  001116       896#   4035*   4036*   4037    4069    4083    4604    4606    4608    4609    4611    4613
$ERRTB  001256       986#   4091
$ERRTY  014220      4040    4076#
$ERTTL  001112       893#   4034*   4069
$ESCAP  001162       917#   1117*   4057    4059    4069    4175*
$ETABL  001214       936#
$ETEND  001256       878     970#
$FATAL  001176       929#   4439*
$FFLG   016140      4402*   4405*   4433    4442*   4450#
$FILLC  001156       914#   4368    4399
$FILLS  001155       913#   4399
$GDADR  001120       897#
$GDDAT  001124       899#   1298*   1300    1313*   1343*   1345    1358*   1388*   1390    1403*   1433*   1435    1448*
                    1478*   1480    1493*   1523*   1525    1538*   1568*   1570    1583*   1613*   1615    1628*   1658*
                    1660    1673*   1703*   1705    1718*   1742*   1743    1746    1760*   1781*   1782    1785    1799*
                    1840*   1841*   1886*   1887*   1915*   1931    1960*   1976    2003*   2049*   2181*   2215*   2216
                    2228    2229    2720*   2862*   2891*   2916*   2937*   3032*   3046    3048    3083    3118*   3132
                    3134    3169    3204*   3218    3220    3255    3291*   3305    3307    3342    3380*   3394    3396
                    3434    3481*   3483    3509    3544*   3546    3558*   3587*   3620*   4604    4606    4609
$GET42  013272      3771#
$GTSWR  014706      4203#   4529
$HO   =  000001       697     698
$HIBTS  001000       873#
$ICNT   001104       890#   4166*   4167    4169*   4180
$ILLUP  016302      4460    4476    4493#
$INTAG  001135       904#   4231    4320
```

```
$ITEMB  001114          894#  4037*  4045   4069   4080
$LF     001172          921#  4069   4305   4314   4399
$LFLG   016137         4443*  4449#
$LPADR  001106          891#  1119*  1209*  4157*  4173*  4178   4180
$LPERR  001110          892#  1120*  1211*  2182*  4056   4157   4174*  4180
$MADR1  001226          954#
$MADR2  001232          958#
$MADR3  001236          961#
$MADR4  001242          964#
$MAIL   001174          874    878    927#   1137   1160   4043   4172   4343
$MAMS1  001224          948#
$MAMS2  001230          956#
$MAMS3  001234          959#
$ MMS4  001240          962#
$MBADR  001002          874#
$MFLG   016136         4403*  4409   4444*  4448#
$MNEW   015401          206   4318#
$MSGAD  001210          934#  4419*  4422
$MSGLG  001212          935#  4424*
$MSGTY  001174          928#  4417   4425*  4437   4441*
$MSWR   015370         4203   4316#
$MTYP1  001225          949#
$MTYP2  001231          957#
$MTYP3  001235          960#
$MTYP4  001241          963#
$MXCNT  014634         4170   4180#
$NULL   001154          912#  4370   4399
$NWTST= 000001         1204#  1245#  1285#  1287   1330#  1332   1375#  1377   1420#  1422   1465#  1467   1510#
                       1512   1555#  1557   1600#  1602   1645#  1647   1690#  1692   1736#  1775#  1820#  1822
                       1859#  1861   1902#  1947#  1993#  1995   2038#  2040   2072#  2107#  2133#  2167#  2169
                       2233#  2276#  2302#  2329#  2331   2380#  2382   2431#  2433   2482#  2484   2533#  2535
                       2584#  2586   2639#  2641   2686#  2732#  2763#  2805#  2848#  2873#  2901#  2929#  2968#
                       3012#  3098#  3184#  3271#  3357#  3450#  3452   3523#  3525   3596#
$OCNT   013732         3941*  3970*  3983#
$OMODE  013734         3936*  3940*  3945   3948*  3959*  3985#
$OVER   014620         4134   4150   4158   4168   4177#
$PASS   001202          931#  1137*  1185*  3683   3745*  3746*  3757   3781   4164   4181
$PASTM  001006          876#
$POWER  016310         4491   4496#
$PWRDN  016142         1114   4460#  4488
$PWRMG  016276         4491#
$PWRUP  016214         4470   4476#
$QUES   001170          919#  4069   4249   4298   4314   4399
$RDCHR  015120         4262#  4532
$RDDEC= ****** U       4534
$RDLIN  015240         4290#  4533
$RDOCT= ****** U       4534
$RDSZ = 000010         4283#
$RTNAD  013314         3780#
$R2A  = ****** U       4534
$SAVRE= ****** U       4534
$SAVR6  016306         4469*  4477   4478*  4479*  4495#
$SCOPE  014354         1108   4130#
$SETUP= 000117         1089#  1107   1108   1110   1112   1114   1116   1117   1119   1156   1157   3743   4027
                       4053   4061   4131   4186   4320
$STUP = 177777         1089#
```

```
$SVLAD  014564          4142    4171#
$SVPC = 000244           850#    855
$SWR  = 167400           679#    697    843#    916    917    918   1116   1117   1119   1120   1208   1249   1294
                        1339    1384   1429    1474   1519   1564   1609   1654   1699   1740   1779   1830   1869
                        1906    1951   2001    2047   2076   2111   2137   2177   2237   2280   2306   2338   2389
                        2440    2491   2542    2593   2648   2690   2736   2767   2809   2852   2877   2905   2933
                        2972    3016   3102    3188   3275   3361   3469   3532   3600   3738   3744   3773   3779
                        3781    4018   4019    4020   4021   4022   4031   4038   4050   4054   4069   4122   4123
                        4124    4125   4126    4133   4145   4147   4148   4151   4152   4153   4160   4161   4162
                        4174    4177   4180    4492
$SWREG  001216           939#   1140
$SWRMK= 000000          4126    4127   4149
$TESTN  001200           930#   4172*
$TIMES  001160           916#   1116*  1208*   1294*  1339*  1384*  1429*  1474*  1519*  1564*  1609*  1654*  1699*
                        1830*  1869*  1906*   1951*  2001*  2047*  2177*  2338*  2389*  2440*  2491*  2542*  2593*
                        2648*  2690*  2767*   2877*  2933*  3016*  3102*  3188*  3275*  3361*  3469*  3532*  3600*
                        3744*  4160*  4167    4170*  4180
$TKB    001146           909#   3608   3615    4184   4195   4212   4266   4272
$TKS    001144           908#   3613   4184    4193   4209   4233*  4264   4270
$TMPO   001120          1078#   1227*  1263*   3046*  3132*  3218*  3305*  3394*  4609   4611   4613
$TMP1   001122          1079#
$TMP3   001424          1080#   1918*  1919*   1920*  1921   1963*  1964*  1965*  1966   2144*  2145*  2146*  2147
                        2203*  2204*  2205*   2206   2350*  2351*  2352*  2353   2401*  2402*  2403*  2404   2452*
                        2453*  2454*  2455    2503*  2504*  2505*  2506   2554*  2555*  2556*  2557   2609*  2610*
                        2611*  2612   2656*   2657*  2658*  2659   2704*  2705*  2706*  2707   2945*  2946*  2947*
                        2948   2986*  2987*   2988*  2989   3034*  3035*  3036*  3037   3071*  3072*  3073*  3074
                        3120*  3121*  3122*   3123   3157*  3158*  3159*  3160   3206*  3207*  3208*  3209   3243*
                        3244*  3245*  3246    3293*  3294*  3295*  3296   3330*  3364*  3332*  3333   3382*  3383*
                        3384*  3385   3422*   3423*  3424*  3425   3497*  3498*  3499*  3500   3560*  3561*  3562*
                        3563   3622*  3623*   3624*  3625
STN   = 000064           697#    844#   1204    1208#  1245   1249#  1285   1294#  1330   1339#  1375   1384#  1420
                        1429#  1465   1474#   1510   1519#  1555   1564#  1600   1609#  1645   1654#  1690   1699#
                        1736   1740#  1775    1779#  1820   1830#  1859   1869#  1902   1906#  1947   1951#  1993
                        2001#  2022   2025    2028   2038   2047#  2069   2072   2076#  2092   2094   2096   2107
                        2111#  2133   2137#   2167   2177#  2200   2233   2237#  2260   2262   2265   2276   2290#
                        2302   2306#  2329    2338#  2380   2389#  2431   2440#  2482   2491#  2533   2542#  2584
                        2593#  2598   2639    2648#  2686   2690#  2732   2736#  2763   2767#  2805   2809#  2848
                        2852#  2873   2877#   2901   2905#  2929   2933#  2968   2972#  3012   3016#  3098   3102#
                        3184   3188#  3271    3275#  3357   3361#  3450   3469#  3523   3532#  3556   3582   3586
                        3596   3600#
STPB    001152           911#   4388*  4399
STPFLG  001157           915#   4337   4399
STPS    001150           910#   4386   4399
STRAP   016320          1112    450 #
$TRP  = 000011          4515#   4524#  4525#   4526#  4527#  4528#  4529   4530#  4531   4532#  4533#  4534#
STR990  016342          4512    4522#
STSTM   001004           875#
STSTNM  001102           888#   1210*  3743*   4030   4069   4121   4149   4171*  4172   4177   4181
STTYIN  015346          4291    4292   4309    4313#
STYPBN  013736          3995#   4527
STYPOS= ****** U        4527
STYPE   015412          4337#   4430   4515    4523
STYPEC  015624          4235    4367   4374    4381   4386#  4387
STYPEX  015672          4392    4394   4397#
STYPOC  013534          3939#   4524
STYPON  013550          3938    3941#  4526
```

M09


```
$TYPOS  013510          3934#   4525
$UNIT   001206           933#
$UNITM  001010           877#
$USWR   001220           940#
$VECT1  001244           965#   1149    3727
$VECT2  001246           966#
$XTSTR  014370          4136#
$SGET4= 000000          3773#
$OFILL  013733          3935*   3939*   3949    3984#
$40CAT= ****** U        4040    4133
.     = 017206           702#    706#    709#    713#    720#    722#    850     851#    853#    855#    856#    862     863#
                         865#    867#    885#    922    1092    1096    1105    1119    1120    1170#   1177#   3612#   3619#
                        3690#   3718#   3762#   3768#   3781    3782#   4069    4115#   4180    4181    4184    4313#   4314
                        4320    4399    4451#   4472    4494
.SWRH   000000           699#
.$ASTA= ****** U        4403    4406
.$X   = 001000           862#    867
```

```
ACTST    1201#  1202   1244
BUFLO    1734#  1735   1774
COMMEN      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
DIVCH    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   1683   1707   1712   1722   1728
         1749   1754   1764   1769   1788   1793   1803   1808   1847   1854   1891   1897   1935   1942   1980
         1987   2012   2019   2029   2034   2058   2064   2083   2089   2097   2102   2117   2124   2157   2162
         2192   2197   2218   2222   2255   2266   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   3088   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
```

DVKMA.P11 CROSS REFERENCE TABLE -- MACRO NAMES

```
                1652    1690    1697    1736    1738    1775    1777    1820    1828    1859    1867    1902    1904    1947    1949
                1993    1999    2038    2045    2072    2074    2107    2109    2133    2135    2167    2175    2233    2235    2276
                2278    2302    2304    2329    2336    2380    2387    2431    2438    2482    2489    2533    2540    2584    2591
                2639    2646    2686    2688    2732    2734    2763    2765    2805    2807    2848    2850    2873    2875    2901
                2903    2929    2931    2968    2970    3012    3014    3098    3100    3184    3186    3271    3273    3357    3359
                3450    3467    3523    3530    3596    3598    3736    3911    3988    4014    4071    4118    4183    4196    4254
                4283    4322    4401    4458    4474    4501
SWRSI             1#     838#   1121#
TEMPBB          4515#
TYPBIN            1#     838#   3769
TYPDEC            1#     838#
TYPHOM            1#     838#   1152
TYPNUM            1#     838#
TYPOCS            1#     676#    838#
TYPOCT            1#     838#   3691    3702    3719    3757    3763    4083    4107    4204
TYPTXT            1#     838#   1174    1178    3609    3616    3687    3693    3698    3715    3721    3753    3759    3765
ZIOTM           3784#   3788    3829
ZP21            1819#   1822    1861
ZP25            1992#   1995
ZP25A           2037#   2040
ZZ1             1281#   1287    1332    1377    1422    1467    1512    1557    1602    1647    1692   2167#    2169   2329#    2331
                2379#   2382   2430#   2433   2481#   2484   2532#   2535   2583#   2586   2638#    2641   3449#    3452   3522#
                3525
$$CMRE           879#
$$CMTM           879#
$$ESCA            1#     838#
$$NEWT            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    3038    3184    3271    3357    3450    3523    3555
$$SET           4515#   4524    4525    4526    4527    4529    4531    4532    4533
$$SETM          1137#
$$SKIP            1#     838#   2022    2025    2028    2069    2092    2094    2096    2200    2260    2262    2265    2536    3556
                3582    3586
.EQUAT            1#     676#    728
.HERDE            1#     675#    687
.KT11             1#
.SETTR          675#
.SETUP            1#     675#   1089
.SWRHI            1#     676#
.TRMTR          675#
.$ACT1            1#     678#    846
.$APTB            1#     923#
.$APTH            1#     678#    857
.$APTY            1#     678#   4395
.$ASTA            1#
.$CATC            1#     676#    700
.$CMTA            1#     676#    879
.$DB2D            1#
.$DB20            1#
.$DIV             1#
.$EOP             1#     677#   3734
.$EPRO            1#     677#   4012
.$ERRT            1#     677#   4069
.$MULT            1#
.$POWE            1#     676#   4456
```

```
.$ROND      1#
.$RCDE      1#
.$RDOC      1#    675#
.$READ      1#    677#   4181
.$RZAZ      1#
.$SAVE      1#
.$SS20      1#
.$SS20      1#
.$S[illegible]      1#    677#   4116
.$S[illegible]      1#
.$SUPR      1#
.$TRAP      1#    675#   4499
.$TYPB      1#    675#   3986
.$TYPD      1#    677#
.$TYPE      1#    677#   4320
.$TYPO      1#    676#   3909
.$40CA      1#
.1170       1#
```

```
ADCB    4001
ADD     1192    1194    1196    1200    1226    1262    2802    2841    3676    3705    3709    3937    3947    4091    4220
        4229    4356    4416    4428    4440
ASL     3673    4088    4089    4090    4243    4244    4245    4511
ASR     4423
BEQ     1139    1161    1173    1301    1315    1346    1360    1391    1405    1436    1450    1481    1495    1526    1540
        1571    1585    1616    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    4290
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    4291
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    2295    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    3895    3966    3967    4247
BISB    4030
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    3568    3863    3897
BNE     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    3865    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
        2882    2927    2935    2936    2937    2941    2966    2973    2974    3008    3018    3019    3095    3104    3105
```

DVKWA.P11 CROSS REFERENCE TABLE -- PERMANENT SYMBOLS

	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	1883	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	1298	1299	1314	1339	1343	1344	1359	1384	1388	1389	1404	1429	1433	1434
	1445	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	2181	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	2961	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	3188	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	4037	4102	4104	4108
	4128	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												
MOVB	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						

DVKWA.P11 CROSS REFERENCE TABLE - PERMANENT SYMBOLS

```
RESET   2006  2052  3773
ROL     3952  3954  3955  3956  3958  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  1286  1287
        1292  1293  1294  1295  1331  1332  1337  1338  1339  1340  1376  1377  1382  1383  1384
        1385  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  1969  1870  1903  1904  1905  1906  1907  1948  1949  1950  1951  1952  1994  1995
        1939  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  2280  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  2908  2909  2849  2950  2951  2852  2874  2875  2976  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
	4534														
.EQUIV	732	733	741	756	757	786	787	788	789	790	791	792	793	794	795
	814	815	816	817	818	819	820	821	822	823					
.EVEN	926	1170	1177	1182	3612	3619	3690	3696	3701	3718	3724	3756	3762	3768	3782
	4115	4451	4498	4602											
.IF	688	709	730	796	824	848	951	853	859	861	868	891	885	887	916
	917	918	922	923	925	948	956	959	962	965	966	967	968	969	970
	972	1089	1101	1106	1108	1110	1112	1114	1116	1117	1119	1137	1155	1156	1157
	1160	1169	1176	1181	1203	1204	1206	1208	1209	1210	1245	1247	1249	1285	1287
	1292	1294	1295	1330	1332	1337	1339	1340	1375	1377	1382	1384	1385	1420	1422
	1427	1429	1430	1465	1467	1472	1474	1475	1510	1512	1517	1519	1520	1555	1557
	1562	1564	1565	1600	1602	1607	1609	1610	1645	1647	1652	1654	1655	1690	1692
	1697	1699	1700	1736	1738	1740	1775	1777	1779	1820	1822	1828	1830	1831	1859
	1861	1867	1869	1870	1902	1904	1906	1907	1947	1949	1951	1952	1993	1995	1999
	2001	2002	2022	2025	2028	2038	2040	2045	2047	2048	2069	2072	2074	2076	2092
	2094	2096	2107	2109	2111	2132	2135	2137	2167	2169	2175	2177	2178	2200	2233
	2235	2237	2260	2262	2265	2276	2278	2280	2302	2304	2306	2329	2331	2336	2338
	2339	2340	2344	2380	2382	2387	2389	2390	2391	2395	2431	2433	2438	2440	2441
	2442	2446	2482	2484	2489	2491	2492	2493	2497	2533	2535	2540	2542	2543	2544
	2548	2584	2586	2591	2593	2594	2595	2598	2603	2639	2641	2646	2648	2649	2650
	2654	2686	2698	2690	2691	2732	2734	2736	2763	2765	2767	2768	2805	2807	2809
	2848	2850	2852	2873	2875	2877	2878	2901	2903	2905	2929	2931	2933	2934	2968
	2970	2972	3012	3014	3016	3017	3024	3031	3050	3062	3068	3098	3100	3102	3103
	3110	3117	3136	3148	3154	3184	3186	3188	3189	3196	3203	3222	3234	3240	3271
	3273	3275	3276	3283	3290	3309	3321	3327	3357	3359	3361	3362	3369	3377	3398
	3410	3417	3450	3452	3467	3469	3470	3523	3525	3530	3532	3533	3556	3582	3586
	3596	3598	3600	3601	3611	3618	3689	3695	3700	3717	3723	3736	3737	3738	3739
	3740	3742	3748	3751	3753	3755	3761	3767	3771	3773	3779	3781	3782	3911	3988
	4014	4017	4028	4031	4038	4040	4041	4043	4050	4054	4061	4069	4071	4086	4102
	4118	4121	4126	4132	4133	4145	4147	4148	4149	4151	4152	4153	4162	4164	4172
	4174	4179	4180	4181	4183	4185	4186	4187	4215	4254	4255	4293	4291	4292	4296
	4297	4313	4314	4320	4322	4343	4401	4403	4406	4433	4448	4458	4468	4469	4474
	4481	4482	4490	4492	4496	4501	4507	4511	4515	4524	4525	4526	4527	4529	4531
	4532	4533	4534												
.IFF	730	849	853	855	860	862	869	882	885	888	916	923	926	1106	1155
	1156	1204	1205	1206	1207	1208	1213	1245	1246	1247	1248	1249	1286	1287	1293
	1294	1331	1332	1338	1339	1376	1377	1383	1384	1421	1422	1428	1429	1466	1467
	1473	1474	1511	1512	1518	1519	1556	1557	1563	1564	1601	1602	1608	1609	1646
	1647	1653	1654	1691	1692	1698	1699	1737	1738	1739	1740	1776	1777	1778	1779
	1821	1822	1829	1830	1860	1861	1868	1869	1903	1904	1905	1906	1948	1949	1950
	1951	1994	1995	2008	2001	2023	2026	2029	2039	2040	2046	2047	2070	2073	2074
	2075	2076	2093	2095	2097	2108	2109	2110	2111	2134	2135	2136	2137	2168	2169
	2176	2177	2201	2234	2235	2236	2237	2261	2263	2266	2277	2278	2279	2280	2303
	2304	2305	2306	2330	2331	2337	2338	2344	2381	2382	2388	2389	2395	2432	2433
	2439	2440	2446	2483	2484	2490	2491	2497	2534	2535	2541	2542	2548	2585	2586
	2592	2593	2599	2603	2640	2641	2647	2648	2655	2687	2688	2689	2690	2733	2734
	2735	2736	2764	2765	2766	2767	2806	2807	2808	2809	2849	2850	2851	2852	2874
	2875	2876	2877	2902	2903	2904	2905	2930	2931	2932	2933	2969	2970	2971	2972
	3013	3014	3015	3016	3050	3099	3100	3101	3102	3136	3185	3186	3187	3188	3222
	3272	3273	3274	3275	3309	3358	3359	3360	3361	3400	3451	3452	3468	3469	3524
	3525	3531	3532	3557	3583	3587	3597	3598	3599	3600	3737	3739	3742	3749	3752
	3781	3912	3989	4015	4017	4031	4061	4072	4087	4116	4119	4146	4149	4150	4153
	4180	4181	4184	4187	4255	4257	4262	4283	4284	4293	4297	4314	4323	4402	4459
	4475	4492	4502	4508											
.IFT	1170	1177	1182	3612	3619	3690	3696	3701	3718	3724	3756	3762	3768	4041	4161

MAINDEC-11-DVKWA-A MACY11 27(732) 04-OCT-76 14:57 PAGE 128
DVKWA.P11 CROSS REFERENCE TABLE -- PERMANENT SYMBOLS

```
        4257  4262
.IFTF   1170  1177  1182  3612  3619  3690  3696  3701  3718  3724  3756  3762  3768  4040  4159
        4202  4255  4258
.IIF     687   692   697   698   706   922   926  1107  1110  1116  1117  1119  1120  1156  3692
        3703  3720  3738  3743  3744  3758  3764  3770  3781  3782  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  1728  1731  1736  1743  1750  1752
        1755  1757  1765  1767  1770  1772  1775  1779  1789  1792  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  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
.MEXIT   971
.NLIST     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    1598    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    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    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
.PAGE            879     972    1244    1280    1284    1328    1329    1373    1374    1418    1419    1463    1464    1508    1509
                1553    1554    1598    1599    1643    1644    1688    1689    1733
.RADIX          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    2805    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)
```

J10

Spooler runtime 15 Seconds, 67 KCS, 950 disk reads, 3 disk writes, 125 pages