MNCKW

MNCKW DIAGNOSTIC
MD-11-DVMNC-A

EP-DVMNC-A-DL-A

FICHE 1 OF 1

APR 1978
digital
MADE IN USA

EOF1CZKMBRSEQ411 00010000 780330 IDENTIFICATION 33HDR1DVMNCASEQ 00010000 780330
SEQ 0001

PRODUCT CODE: MAINDEC-11-DVMNC-A-D

PRODUCT NAME: MNCKW DIAGNOSTIC

DATE CREATED: MARCH 1978

MAINTAINER: DIAGNOSTIC ENGINEERING

TABLE OF CONTENTS

1.0 ABSTRACT

THIS PROGRAM ALLOWS THE USER TO CHECK-OUT OR DEBUG THE MNCKW PROGRAMMABLE REAL-TIME CLOCK. THE LOGIC TEST IS SELF CONTAINED AND NEEDS TO EXTERNAL MAINTENANCE HARDWARE OR OPERATOR INTERVENTION WITH ONLY ONE EXCEPTION: IF THE CUSTOMER HARDWARE CONNECTED TO THE MNCKW COULD INJECT SIGNALS ON ST2, ST1, OR SLAVE IN INPUTS, IT MUST BE DISCONNECTED.

EVEN THOUGH THE MNCKW IS A Q BUS OPTION, THIS PROGRAM WAS DESIGNED TO RUN ON ANY PDP-11 FAMILY COMPUTER. IF THE USER IS UNFAMILIAR 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 8K OF MEMORY (OR MORE) AND I/O FACILITIES (A SWITCH REGISTER OR TTY).

2. MNCKW UNDER TEST.

2.2 STORAGE

THIS PROGRAM OCCUPIES AND USES ONLY THE LOWER 8K OF MEMORY.

3.0 LOADING PROCEDURE

3.1 METHOD

STANDARD PROCEDURE FOR NORMAL BINARY TAPES SHOULD BE FOLLOWED.

1. ABSOLUTE LOADER MUST BE IN MEMORY.
2. PLACE BINARY TAPE IN READER.
3. TYPE ADDRESS #7500 (5# DETERMINE BY LOCATION OF LOADER).
4. TYPE "G" (PROGRAM WILL BE LOADED INTO MEMORY).

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

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

THIS PROGRAM IS SET TO TEST A MNCKW WITH A STANDARD ADDRESS AND VECTOR. IF ANY OF THESE ARE DIFFERENT ON THE MNCKW 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	$VEcTI:	000440	;;INTERRUPT VECTOR #1
176	$SWREG:	000000	;;MANUAL SWR.
1157	$TPFLG:	.BYTE 0	;;"TERMINAL AVAILABLE" ;; FLAG (BIT<0:7>=0=YES)

NOTE

SINCE 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
250 DWARF STARTING ADDRESS

4.3 PROGRAM AND/OR OPERATOR ACTION

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

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 MNCKW'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 ;THERE ARE 4 (OCTAL) UNITS.

THE GOOD UNITS ARE ( L TO R) 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 MNCKW IT DETECTS ON THE SYSTEM. TO INHIBIT THIS FEATURE, SET BIT 15 OF LOCATION "$ENVM". ALSO, TO TEST AN INDIVIDUAL MNCKW IN A GROUP, SET THIS BIT AND REFER TO SECTION 3.2 FOR CHANGING THE BASE ADDRESS OF THE MNCKW 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 "$TYPE"+10 WHEN RUNNING WITH NO TERmINAL INDICATES AN ERROR HAS OCCURRED. TO FIND OUT THE NUMBER OF THE ERROR, EXAMINE LOCATION "$TSTNM". 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 "$TSTMN" AND FOUND A 5(101) WE GO TO LOCATION "$ERRTB" AND LOOK THROUGH THE ERROR POINTER TABLE UNTIL WE FOUND ITEM 5. THE INFORMATION WOULD LOOK LIKE:

```
;ITEM 5

        EM5         ;CLOCK SR DATA ERROR
        DH5         ;ERRPC ASR WAS 5/8
        DT5         ;$ERRPC,ASR,$BDDAT,$GDDAT
        DF0         ;ALL NUMBERS ARE IN OCTAL FORM
```

TO FIND OUT THE INFORMATION SPECIFIED BY DT5 ($ERRPC,BSR,$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 MNCKW 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 SIGNALS 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 CANNOT 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(PS) TO 200, LOADING THE PC WITH THE STARTING ADDRESS INSTEAD OF USING THE "G" COMMAND, AND THEN USING THE "P" COMMAND. BEFORE USING THE "I" COMMAND, THE PSW(PS) 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 MNCKW RESPONDS TO THE ADDRESS THE PROGRAM THINKS ITS AT. IF THE MNCKW DOES NOT RESPOND, A BUS ERROR OCCURS. ALSO BUS ERRORS CAN OCCUR DURING THE TIME THE PROGRAM SIZES TO SEE HOW MANY KWV11'S ARE ON YOUR SYSTEM.

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

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

8.0 MISCELLANEOUS

8.1 POWER FAIL

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

8.2 XXDP, ACT, APT

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

8.3 EXECUTION TIME

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

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 LOATION R.
/	REOPEN THE LAST LOCATION.
$N/ 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 COMMANDS 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 TES! MUST HAVE ALREADY BEEN RUN ON THE MNCKW.

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 JUMFER MUST BE INSTALLED.

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.

SEQ 0016

8.8 PRODUCTION STARTING ADDRESS

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

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

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

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.

8.10 DWARF STARTING ADDRESS

MORE COMPLETE TESTING OF THE CLOCKS I/O SIGNALS CAN BE MADE IF A DWARF MODULE IS CONNECTED TO THE CLOCK. IF YOU DO THIS, START THE DWARF'S STARTING ADDRESS.
A SERIES OF INSTRUCTIONS WILL BE TYPED OUT FOR YOU TO FOLLOW.

```
                                .NLIST  MC,MD,CND
                                .LIST   ME
                                .ENABL  ABS
                                .ENABL  AMA
                                .MCALL  .HEADER,.SETUP,.SETTRAP,.TRMTRP,.$TRAP,.$RDOCT,.$TYPBIN
                                .MCALL  TYPOCS,.$POWER,.$CATCH,.$TYPOCT,.EQUAT,.$CMTAG,.SWRHI
                                .MCALL  .$EVP,.$ERROR,.$ERRTYP,.$TYPDEC,.$SCOPE,.$READ,.$TYPE
                                .MCALL  .$ACT11,.$APTHDR,.$APTYPE
167400                          $SWR=   167400



                        .TITLE  MAINDEC-11-DVMNC-A
                        ;*COPYRIGHT (C) 1977
                        ;*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-C2), SEPT 14, 1976.
                        ;*
000001                  $TN=1

                        .SBTTL  OPERATIONAL SWITCH SETTINGS
                        ;*
                        ;*      SWITCH                  USE
                        ;*      ------          --------------------
                        ;*        15            HALT ON ERROR
                        ;*        14            LOOP ON TEST
                        ;*        13            INHIBIT ERROR TYPEOUTS
                        ;*        11            INHIBIT ITERATIONS
                        ;*        10            BELL ON ERROR
                        ;*         9            LOOP ON ERROR
                        ;*         8            LOOP ON TEST IN SWR<7:0>

                                .SBTTL  TRAP CATCHER

000000                          .=0
                        ;*ALL UNUSED LOCATIONS FROM 4-776 CONTAIN A ".+2"
                        ;*AND "JSR PC,@0" SEQUENCE TO CATCH ILLEGAL INTERRUPTS.
                        ;*AND INTERRUPTS TO THE WRONG VECTOR.
                        ;*LOCATION 0 CONTAINS A 0 TO CATCH IMPROPERLY LOADED
                        ;*VECTORS.
000004                  .=4
000004  020032  000200          .WORD   IOTRD,200       ;HANDLE BUSS ERROR.
000174                  .=174
000174  000000         DISPREG:        .WORD   0               ;;SOFTWARE DISPLAY REGISTER.
000176  000000         SWREG:  .WORD   0               ;;SOFTWARE SWITCH REGISTER.
000100                  .=100
```

```
                                                ;IF "B EVENT"ON Q-BUS IS
                                                ;CONNECTED,WE NEED A WAY OF
                                                ;IGNORING ITS INTERRUPTS.

                    000200          .=200
000200  000137  001534              JMP     @#START
000204  000137  002162              JMP     @#RSTART

                    000230          .=230
000230  000137  001514              JMP     @#WSTART            ;WESTFIELD STARTING ADDRESS
                    000240          .=240
000240  000137  001474              JMP     @#TSTSTR            ;ALL TESTER TESTS
                    000250          .=250
000250  000137  001454              JMP     @#DWARFT            ;TEST MNCKW WITH DWARF MODULE.
                                                                ;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
                    000004      R4=     %4              ;;GENERAL REGISTER
                    000005      R5=     %5              ;;GENERAL REGISTER
                    000006      R6=     %6              ;;GENERAL REGISTER
                    000007      R7=     %7              ;;GENERAL REGISTER
                    000006      SP=     %6              ;;STACK POINTER
                    000007      PC=     %7              ;;PROGRAM COUNTER

                                ;*PRIORITY LEVEL DEFINITIONS
                    000000      PR0=    0               ;;PRIORITY LEVEL 0
                    000040      PR1=    40              ;;PRIORITY LEVEL 1
                    000100      PR2=    100             ;;PRIORITY LEVEL 2
                    000140      PR3=    140             ;;PRIORITY LEVEL 3
```

```
                        
                        PR4=    200             ;;PRIORITY LEVEL 4
                        PR5=    240             ;;PRIORITY LEVEL 5
                        PR6=    300             ;;PRIORITY LEVEL 6
                        PR7=    340             ;;PRIORITY LEVEL 7
                        
                        ;*"SWITCH REGISTER" SWITCH DEFINITIONS
100000                  SW15=   100000
040000                  SW14=   40000
020000                  SW13=   20000
010000                  SW12=   10000
004000                  SW11=   4000
002000                  SW10=   2000
001000                  SW09=   1000
000400                  SW08=   400
000200                  SW07=   200
000100                  SW06=   100
000040                  SW05=   40
000020                  SW04=   20
000010                  SW03=   10
000004                  SW02=   4
000002                  SW01=   2
000001                  SW00=   1
                        .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)
100000                  BIT15=  100000
040000                  BIT14=  40000
020000                  BIT13=  20000
010000                  BIT12=  10000
004000                  BIT11=  4000
002000                  BIT10=  2000
001000                  BIT09=  1000
000400                  BIT08=  400
000200                  BIT07=  200
000100                  BIT06=  100
000040                  BIT05=  40
000020                  BIT04=  20
000010                  BIT03=  10
000004                  BIT02=  4
000002                  BIT01=  2
000001                  BIT00=  1
                        .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 OTHER ERRORS
000010                  RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
000014                  TBITVEC=14              ;;"T" BIT
000014                  TRTVEC= 14              ;;TRACE TRAP
000014                  BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
000020                  IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
000024                  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                          AVECT1= 440
000200                          APRIOR= 200

167400                          $SWR=   167400
000001                          $TN=    1

                        .SBTTL  ACT11 HOOKS

                        ;;**************************************************************
                        ;HOOKS REQUIRED BY ACT11
000254                          $SVPC=.                 ;SAVE PC
000046                          .=46
014664                  000046  $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOp
000052                          .=52
000000                  000052  .WORD   0               ;;2)SET LOC.52 TO ZERO
000254                          .=$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
000200                  000024  200     ;;FOR APT START UP
000044                          .=44    ;;POINT TO APT INDIRECT ADDRESS PNTR.
001000                  000044  $APTHDR ;;POINT TO APT HEADER BLOCK
001000                          .=.$X   ;;RESET LOCATION COUNTER
                        ;;**************************************************************
                        ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
                        ;INTERFACE SPEC.
```

```
    217  001000
    218  001000  000000  $APTHD:
    219  001002  001174  $HIBTS: .WORD   0          ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
    220  001004  000002  $MBADR: .WORD   $MAIL      ;;ADDRESS OF APT MAILBOX (BITS 0-15)
    221  001006  000170  $TSTM:  .WORD   2          ;;RUN TIM OF LONGEST TEST
    222  001010  000170  $PASTM: .WORD   120.       ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
    223  001012  000031  $UNITM: .WORD   120.       ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
                         .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                  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
```

```
                                $UNIT:  .WORD   AUNIT   ;;I/O UNIT NUMBER
                                $MSGAD: .WORD   AMSGAD  ;;MESSAGE ADDRESS
                                $MSGLG: .WORD   AMSGLG  ;;MESSAGE LENGTH
                                $ETABLE:                ;;APT ENVIRONMENT TABLE
                                $ENV:   .BYTE   AENV    ;;ENVIRONMENT BYTE
                                $ENVM:  .BYTE   AENVM   ;;ENVIRONMENT MODE BITS
                                $SWREG: .WORD   ASWREG  ;;APT SWITCH REGISTER
                                $USWR:  .WORD   AUSWR   ;;USER SWITCHES
                                $CPUOP: .WORD   ACPUOP  ;;CPU TYPE,OPTIONS
                                ;*                      BITS 15-11=CPU TYPE
                                ;*                              11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
                                ;*                              11/70=06,PDQ=07,Q=10
                                ;*                      BIT 10=REAL TIME CLOCK
                                ;*                      BIT  9=FLOATING POINT PROCESSOR
                                ;*                      BIT  8=MEMORY MANAGEMENT
                                $MAMS1: .BYTE   AMAMS1  ;;HIGH ADDRESS,M.S. BYTE
                                $MTYP1: .BYTE   AMTYP1  ;;MEM. TYPE,BLK#1
                                ;*                      MEM.TYPE BYTE   --  (HIGH BYTE)
                                ;*                              900 NSEC CORE=001
                                ;*                              300 NSEC BIPOLAR=002
                                ;*                              500 NSEC MOS=003
                                $MADR1: .WORD   AMADR1  ;;HIGH ADDRESS,BLK#1
                                ;*                        MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
                                $MAMS2: .BYTE   AMAMS2  ;;HIGH ADDRESS,M.S. BYTE
                                $MTYP2: .BYTE   AMTYP2  ;;MEM.TYPE,BLK#2
                                $MADR2: .WORD   AMADR2  ;;MEM.LAST ADDRESS,BLK#2
                                $MAMS3: .BYTE   AMAMS3  ;;HIGH ADDRESS,M.S.BYTE
                                $MTYP3: .BYTE   AMTYP3  ;;MEM.TYPE,BLK#3
                                $MADR3: .WORD   AMADR3  ;;MEM.LAST ADDRESS,BLK#3
                                $MAMS4: .BYTE   AMAMS4  ;;HIGH ADDRESS,M.S.BYTE
                                $MTYP4: .BYTE   AMTYP4  ;;MEM.TYPE,BLK#4
                                $MADR4: .WORD   AMADR4  ;;MEM.LAST ADDRESS,BLK#4
                                $VECT1: .WORD   AVECT1  ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
                                $VECT2: .WORD   AVECT2  ;;INTERRUPT VECTOR#2BUS PRIORITY#2
                                $BASE:  .WORD   ABASE   ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
                                $DEVM:  .WORD   ADEVM   ;;DEVICE MAP
                                $CDW1:  .WORD   ACDW1   ;;CONTROLLER DEScRIPTION WORD#1
                                $ETEND:
                                .MEXIT
```

```
                                .SBTTL  ERROR POINTER TABLE

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

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


001256                          $ERRTB:


                                ;ITEM   1

001256  020272                          EM1                     ;CLOCK   SR FUNCTION ERROR
001260  020615                          DH1                     ;ERRPC   ASR      WAS      S/B
001262  021142                          DT1                     ;$ERRPC,ASR,$BDDAT,$GDDAT
001264  021236                          DF0                     ;ALL NUMBERS ARE IN OCTAL FORM


                                ;ITEM   2

001266  020324                          EM2                     ;CLOCK   SR DATA ERROR
001270  020615                          DH1                     ;ERRPC   ASR      WAS      S/B
001272  021142                          DT1                     ;$ERRPC,ASR,$BDDAT,$GDDAT
001274  021236                          DF0                     ;ALL NUMBERS ARE IN OCTAL FORM


                                ;ITEM   3

001276  020352                          EM3                     ;CLOCK  BR DATA ERROR
001300  020641                          DH3                     ;ERRPC   ABR      WAS
001302  021154                          DT3                     ;$ERRPC,ABR,$BDDAT,$GDDAT
001304  021236                          DF0                     ;ALL NUMBERS ARE IN OCTAL FORM


                                ;ITEM   4

001306  020400                          EM4                     ;INTERRUPT ERROR.
001310  020665                          DH4A                    ;ERRPC   TO      ROM ADDR.
001312  021166                          DT4                     ;$ERRPC,         TRTO,TRFRO
001314  021236                          DF0                     ;ALL NUMBERS ARE IN OCTAL FORM


                                ;ITEM   5

001316  020421                          EM5                     ;CLOCK   COUNT REG ERROR
001320  020615                          DH1                     ;ERRPC   ASR    WAS    S/B
001322  021142                          DT1                     ;$ERRPC, ACR, $BDDAT, $GDDAT
```

```
001324  021236                          DF0                     ;ALL NUMBERS ARE IN OCTAL FORM


                                ;ITEM   6

001326  020463                          EM12                    ;CLOCK   COUNT FUNCTION ERROR
001330  020721                          DH12                    ;ERRPC   ASR
001332  021176                          DT12                    ;ERRPC, ASR
001334  021236                          DF0                     ;ALL NUMBERS ARE IN OCTAL FORM


                                ;ITEM   7

001336  020512                          EM16                    ;CLOCK   INTERRUPT ERROR
001340  020721                          DH12                    ;ERRPC   ASR
001342  021176                          DT12                    ;$ERRPC, ASR
001344  021236                          DF0                     ;ALL NUMBERS ARE IN OCTAL FORM


                                ;ITEM   10

001346  020543                          EM20                    ;CLOCK   REPEATABILITY ERROR
001350  020736                          DH20                    ;ERROR   ASR   2ND CNT 1ST CNT  3RD CNT
001352  021204                          DT20                    ;$ERRPC, ASR, $BDDAT, $GDDAT, $TMPO
001354  021236                          DF0                     ;ALL NUMBERS ARE IN OCTAL FORM


                                ;ITEM   11

001356  020444                          EM11                    ;CLOCK   COUNT ERROR
001360  020615                          DH1                     ;ERRPC   ASR   WAS   S/B
001362  021216                          DT22                    ;$ERRPC, ASR, $BDDAT, $TMPO
001364  021236                          DF0                     ;ALL NUMBERS ARE IN OCTAL FORM


                                ;ITEM   12

001366  020572                          EM26                    ;CLOCK ADDRESSING ERROR
001370  020771                          DH26                    ;ERRPC   CLOCK ADDR.
001372  021230                          DT26                    ;$ERRPC,$TMPO
001374  021236                          DF0                     ;ALL NUMBERS ARE IN OCTAL FORM


001376  170420          ASR:    .WORD   ABASE
001400  170422          ABR:    .WORD   ABASE+2
001402  000440          VECT1:  .WORD   AVECT1
001404  000442          VECTP:  .WORD   AVECT1+2
001406  000444          VECT2:  .WORD   AVECT1+4        ;VECTOR ADDR. OF ST2 INTRS.
001410  000446          VECT2P: .WORD   AVECT1+6
001412  000200          PRIOR:  .WORD   APRIOR

001414  167774          DR:     .WORD   167774
001416  167772          DR2:    .WORD   167772
001420  170430          TSCLC:  .WORD   170430          ;ADR. OF TESTOR CLOCK
```

```
001422  170432                  TSCLD:  .WORD   170432          ;BUFFER PRESET REG.
001424  000000                  $TMP0:  .WORD   0               ;TEMP STORAGE.
001426  000000                  $TMP1:  .WORD   0               ;TMP STORAGE.
001430  000000                  $TMP3:  .WORD   0
001432  000000                  ROTATE: .WORD   0                       ;POINT TO DEVICE UNDER TEST.
001434  000000                  UTEST:  .WORD   0                       ;KEEPS TRACK OF GOOD UNITS.
001436  000000                  ERCNT:  .WORD   0                       ;COUNTS ERRORS.
001440  000000                  MDEVCT: .WORD   0                       ;COUNTS DEVICES TESTED.
001442  000000                  TSTCNT: .WORD   0                       ;MAX DEVICES TO BE TESTED.
001444  000000                  EXS:    .WORD   0                       ;=0, NORMAL; =1 SPECIAL TESTOR START, BY L+S @ 2
001446  000000                  LCNT:   .WORD   0                       ;TOTAL UNITS TESTED.
001450  000000                  DWARF:  .WORD   0                       ;INDICATE IF DWARF MODULE PRESENT (=1,YES)
001452  000000                  ASK:    .WORD   0                       ;=1 WHEN QUESTION ASKED IN RUN.

001454                          DWARFT=.
001454  005237  001450                  INC     DWARF           ;INDICATE DWARF TESTS.
001460  005237  001444                  INC     EXS             ;PLUS REST OF TESTS.
001464  012737  000004  001442          MOV     #4,TSTCNT       ;MAX TO BE TESTED.
001472  000427                  BR      1$
001474                          TSTSTR=.
001474  005237  001444                  INC     EXS                     ;SET FOR TESTOR.
001500  005037  001450                  CLR     DWARF
001504  012737  000020  001442          MOV     #16.,TSTCNT             ;ALLOW 16 UNITS
001512  000417                  BR      1$
001514                          WSTART=.
001514  012737  000020  001442          MOV     #16.,TSTCNT     ;TEST UP TO 16 UNITS.
001522  005037  001450                  CLR     DWARF
001526  005037  001444                  CLR     EXS
001532  000407                  BR      1$
001534                          START=.
001534  012737  000004  001442          MOV     #4,TSTCNT       ;TEST UP TO FOUR UNITS.
001542  005037  001450                  CLR     DWARF
001546  005037  001444                  CLR     EXS
001552                          1$:
                                .SBTTL  INITIALIZE THE COMMON TAGS
                                ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
001552  012706  001100                  MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
001556  005026                  CLR     (R6)+           ;;CLEAR MEMORY LOCATION
001560  022706  001140                  CMP     #SWR,R6 ;;DONE?
001564  001374                  BNE     .-6             ;;LOOP BACK IF NO
001566  012706  001100                  MOV     #STACK,SP       ;;SETUP THE STACK POINTER
                                ;;INITIALIZE A FEW VECTORS
001572  012737  016066  000020          MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
001600  012737  000340  000022          MOV     #340,@#IOTVEC+2 ;;LEVEL 7
001606  012737  015524  000030          MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
001614  012737  000340  000032          MOV     #340,@#EMTVEC+2 ;;LEVEL 7
001622  012737  020212  000034          MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
001630  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
001636  012737  017654  000024          MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
001644  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;LEVEL 7
001652  005037  001160                  CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
001656  005037  001162                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
001662  112737  000001  001115          MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
```

```
                                                MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
                                                MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
                                ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
                                ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
                                                MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
                                                MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
                                                MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
                                                MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
                                                CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
                                                BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                                        ;;AND  THE HARDWARE SWR IS NOT = -1
                                                BR      65$             ;;BRANCH IF NO TIMEOUT
                                64$:            MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
                                                RTI
                                65$:            MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
                                                MOV     #DISPREG,DISPLAY
                                66$:            MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR

                                                CLR     $PASS           ;;CLEAR PASS COUNT
                                                BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
                                                BEQ     67$             ;;YES,USE NON-APT SWITCH
                                                MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
                                67$:


                                                MOV     #340,-(SP)              ;SET CPU PRIORITY ON RETERN.
                                                MOV     #68$,-(SP)              ;SHOW RETURN ADDRESS.
                                                RTI                             ;CAUSE A RETURN(PUTS STATUS IN STATUS REG.).
                                68$:

                                                CLR     $DEVCT          ;ZERO DEVICE COUNT.
                                                MOV     #IOTRD,@#ERRVEC ;FIX TRAP CATCHER.
                                                MOV     #340,@#ERRVEC+2
                                                MOV     $VECTI,VECTI    ;NOW FIX VECTOR ADDR.
                                                MOV     $BASE,ASR       ;FIX ADDRESS OF CSR.

                                                TST     @#42            ;RUNNIGN UNDER APT,XXDP?
                                                BNE     RSTART
                                                INC     #-1             ;ONLY TYPE ON 1ST START UP
                                                BNE     RSTART
                                                TYPE,   WRNM1
                                .SBTTL  TYPE PROGRAM NAME
                                ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
                                                INC     #-1             ;;FIRST TIME?
                                                BNE     69$             ;;BRANCH IF NO
                                                TYPE    ,70$            ;;TYPE ASCIZ STRING
                                .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
                                                TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
                                                BNE     71$             ;;BRANCH IF YES
                                                CMPB    $ENV,#1         ;;ARE WE RUNNING UNDER APT?
                                                BEQ     71$             ;;BRANCH IF YES
                                                CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
                                                BNE     72$             ;;BRANCH IF NO
                                                GTSWR                   ;;GET SOFT-SWR SETTINGS
```

```
                                        BR      72$
002144  000403
002146  112737  000001  001134  71$:    MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
                                72$:
002154                                  BR      69$             ;;GET OVER THE ASCIZ
002154  000402
                                ;;70$:  .ASCIZ  <CRLF>##<CRLF>
                                69$:
002162
                                RSTART:
002162                                  CLR     ASK
002162  005037  001452                   TST     EXS             ;TESTOR MODE ENABLED??
002166  005737  001444                   BEQ     1$              ;NO DON'T TYPE NEXT MESSAGE.
002172  001417                          TYPE    ,65$            ;;TYPE ASCIZ STRING
002174  104401  002202                   BR      64$             ;;GET OVER THE ASCIZ
002200  000414
                                ;;65$:  .ASCIZ  <15><12>#TESTOR MODE ENABLED--#
                                64$:
002232
                                1$:
002232                                  TYPE    ,67$            ;;TYPE ASCIZ STRING
002232  104401  002240                   BR      66$             ;;GET OVER THE ASCIZ
002236  000411
                                ;;67$:  .ASCIZ  <15><12>#TEST RUNNING...#
                                66$:
002262
002262  005037  001440                   CLR     MDEVCT          ;TESTING FIRST UNIt.
002266  005037  001436                   CLR     ERCNT           ;NO ERRORS.
002272  005037  001202                   CLR     $PASS           ;NO PASSES.
002276  012737  000001  001432           MOV     #1,ROTATE       ;POINT TO FIRST UNIT.
002304  013737  001432  001434           MOV     ROTATE,UTEST
002312                          LOOP:                           ;COME HERE FOR NEXT UNIT,OR END PASS.

002312  042737  170000  001402           BIC     #170000,VECT1   ;CLEAR OUT PRIORITY BITS.
002320  013737  001402  001404           MOV     VECT1,VECTP     ;NOW FIX VECTOR +2 ADDR.
002326  062737  000002  001404           ADD     #2,VECTP
002334  013737  001402  001406           MOV     VECT1,VECT2     ;LETS FIX ST2 VECTOR ADDR.
002342  062737  000004  001406           ADD     #4,VECT2        ;ITS 4 GREATER THEN THE 1ST.
002350  013737  001406  001410           MOV     VECT2,VECT2P    ;VECTOR +2 ADDR.
002356  062737  000002  001410           ADD     #2,VECT2P

002364  013737  001376  001400           MOV     ASR,ABR         ;FIX ADDR OF PRESET REG=
002372  062737  000002  001400           ADD     #2,ABR          ;CSR + 2


                        ;;**************************************************************
                        ;*TEST 1         *TEST THE ADDRESSABILITY OF CLOCK CSR
                        ;;**************************************************************
002400  000240          TST1:   NOP
002402  012737  000050  001160           MOV     #50,$TIMES      ;;DO 50 ITERATIONS
002410  012737  002440  001106           MOV     #1$,$LPADR      ;;SET SCOPE LOOP ADDRESS
002416  112737  000001  001102           MOVB    #1,$TSTNM       ;SET TEST #1.
002424  112737  000001  001200           MOVB    #1,$TESTN       ;DON'T FORGET APT!
002432  012737  002440  001110           MOV     #1$,$LPERR


002440  013746  000004          1$:     MOV     @#ERRVEC,-(SP)  ;SAVE CONTENTS OF ADDRS 6.
002444  012737  002460  000004           MOV     #2$,@#ERRVEC    ;SET TIME-OUT TRAP VECTOR TO hANDLER IN CASE.
                                                                ;WE TIME-OUT WHEN ADDRESSING THE KW11.
```

```
002452  005777  176720                  TST     @ASR                    ;ADDRESS THE CLOCK!
                                                                        ;IF CLOCK DOEs NOT RETURN
                                                                        ;"BUS SSYN" THEN WE'LL TIME-OUT

002456  000406                          BR      3$                      ;THE CLOCK WAS THERE! EXIT SUB-TESt.
002460                          2$:
002460  062706  000004                  ADD     #4,SP                           ;/ADD #4 TO sTACK POINTER.
002464  013737  001376  001424          MOV     ASR,$TMPO               ;FOR ERROR TYPEOUT.


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

002472  104012                          ERROR   12                      ;REPORT ERROR=CLOCK CSR FAILED TO rETURN
                                                                        ;"BUS SSYN" WHEN ADDRESSED.
                                                                        ;NOTE: IF PROGRAM HAS INCORRECT
                                                                        ;ADDRESS THEN WE MIG NOT JE
                                                                        ;TALKING TO THE CLOCK. MAKE SURE
                                                                        ;OF CLOCK ADDRESS.


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

002474  012637  000004          3$:     MOV     (SP)+,@#ERRVEC


                                ;;*************************************************************
                                ;*TEST 2         *TEST THE ADDRESSABILITY OF CLOCK BUFFER REG.
                                ;;*************************************************************
002500  000004                  TST2:   SCOPE


002502  013746  000004          1$:     MOV     @#ERRVEC,-(SP)          ;SAVE CONTENTS OF ADDRS 6.
002506  012737  002522  000004          MOV     #2$,@#ERRVEC            ;SET TIME-OUT TRAP VECTOR TO HANDLER IN CASE.
                                                                        ;WE TIME-OUT WHEN ADDRESSING THE KW11.
002514  005777  176660                  TST     @ABR                    ;ADDRESS THE CLOCK!
                                                                        ;IF CLOCK DOES NOT RETURN
                                                                        ;"BUS SSYN" THEN WE'LL TIME-OUT

002520  000406                          BR      3$                      ;THE CLOCK WAS THERE! EXIT SUB-TEST.
002522                          2$:
002522  062706  000004                  ADD     #4,SP                           ;/ADD #4 TO STACK POINTER.
002526  013737  001400  001424          MOV     ABR,$TMpO               ;FOR ERROR TYPEOUT.


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

002534  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.
```

G03


```
                                ;;$$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
002536  012637  000004          3$:     MOV     (SP)+,@#ERRVEC
```

```
                                        ;/*
                                        ;;**********************************************************************
                                        ;*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
                                        ;*
                                        ;;**********************************************************************
002542  000004                        TST3:   SCOPE
002544  012737  000100  001160                MOV     #100,$TIMES     ;;DO 100 ITERATIONS

002552  005077  176620                        CLR     @ASR            ;/CLEAR THE STATUS REGISTER.
002556  052777  040000  176612                BIS     #BIT14,@ASR     ;/SET BIT 14.
002564  012737  040000  001124                MOV     #BIT14,$GDDAT   ;/SET FOR ERROR TYPEOUT S/B.
002572  017737  176600  001126                MOV     @ASR,$BDDAT     ;/READ THE STATUS REGISTER.
002600  023737  001124  001126                CMP     $GDDAT,$BDDAT   ;/DID BIT 14 AND ONLY BIT 14 SET?
002606  001402                                BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.

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

002610  104002                                ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER
                                                                        ;/BIT 14 FAILED TO BIT SET.

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

002612  000412                                BR      2$              ;/BR TO END SUBTEST.

002614  042777  040000  176554  1$:           BIC     #BIT14,@ASR     ;/TRY CLEARING BIT 14.
002622  005037  001124                        CLR     $GDDAT          ;/CLEAR S/B FOR TYPEOUT IF ANY.
002626  017737  176544  001126                MOV     @ASR,$BDDAT     ;/NOW READ IT BACK.
002634  001401                                BEQ     2$              ;/IF ZERO - NO ERROR!

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

002636  104002                                ERROR   2               ;/ERROR - CLOCK A STATUS REGISTER.
                                                                        ;/BIT 14 FAILED TO CLEAR.

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

002640                                  2$:
```

```
                                        ;/*
                                ;;**********************************************************************
                                ;*TEST 4         *TEST THAT CLOCK A STATUS REGISTER BIT 13 CAN BE SET AND CLEARED
                                ;*
                                ;*CLOCK STATUS REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
                                ;*F/FS OR GATES
                                ;*
                                ;;**********************************************************************
002640  000004                  TST4:   SCOPE
002642  012737  000100  001160          MOV     #100,$TIMES     ;;DO 100 ITERATIONS

002650  005077  176522          1$:     CLR     @ASR            ;/CLEAR THE STATUS REGISTER.
002654  052777  020000  176514          BIS     #BIT13,@ASR     ;/SET BIT 13.
002662  012737  020000  001124          MOV     #BIT13,$GDDAT   ;/SET FOR ERROR TYPEOUT S/B.
002670  017737  176502  001126          MOV     @ASR,$BDDAT     ;/READ THE STATUS REGISTER.
002676  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;/DID BIT 13 AND ONLY BIT 13 SET?
002704  001402                        BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.

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

002706  104002                        ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER
                                                                ;/BIT 13 FAILED TO BIT SET.

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

002710  000412                        BR      2$              ;/BR TO END SUBTEST.

002712  042777  020000  176456  1$:     BIC     #BIT13,@ASR     ;/TRY CLEARING BIT 13.
002720  005037  001124                  CLR     $GDDAT          ;/CLEAR S/B FOR TYPEOUT IF ANY.
002724  017737  176446  001126          MOV     @ASR,$BDDAT     ;/NOW READ IT BACK.
002732  001401                        BEQ     2$              ;/IF ZERO - NO ERROR!

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

002734  104002                        ERROR   2               ;/ERROR - CLOCK A STATUS REGISTER.
                                                                ;/BIT 13 FAILED TO CLEAR.

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

002736                          2$:
```

```
                                        ;/*
                                        ;;**********************************************************************
                                        ;*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
                                        ;*
                                        ;;**********************************************************************
002736  000004                        1$TS:   SCOPE
002740  012737  000100  001160                MOV     #100.,$TIMES    ;;DO 100 ITERATIONS

002746  005077  176424                        CLR     @ASR            ;/CLEAR THE STATUS REGISTER.
002752  052777  004000  176416                BIS     #BIT11,@ASR     ;/SET BIT 11.
002760  012737  004000  001124                MOV     #BIT11,$GDDAT   ;/SET FOR ERROR TYPEOUT S/B.
002766  017737  176404  001126                MOV     @ASR,$BDDAT     ;/READ THE STATUS REGISTER.
002774  023737  001124  001126                CMP     $GDDAT,$BDDAT   ;/DID BIT 11 AND ONLY BIT 11 SET?
003002  001402                                BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.

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

003004  104002                                ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER
                                                                        ;/BIT 11 FAILED TO BIT SET.

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

003006  000412                                BR      2$              ;/BR TO END SUBTEST.

003010  042777  004000  176360  1$:     BIC     #BIT11,@ASR     ;/TRY CLEARING BIT 11.
003016  005037  001124                CLR     $GDDAT          ;/CLEAR S/B FOR TYPEOUT IF ANY.
003022  017737  176350  001126                MOV     @ASR,$BDDAT     ;/NOW READ IT BACK.
003030  001401                                BEQ     2$              ;/IF ZERO - NO ERROR!

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

003032  104002                                ERROR   2               ;/ERROR - CLOCK A STATUS REGISTER.
                                                                        ;/BIT 11 FAILED TO CLEAR.

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

003034                                  2$:
```

```
                                        ;/*
                                        ;;**********************************************************************
                                        ;*TEST 6         *TEST THAT CLOCK A STATUS REGISTER BIT 6 CAN BE SET AND CLEARED
                                        ;*
                                        ;*CLOCK STATUS REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
                                        ;*F/FS OR GATES
                                        ;*
                                        ;;**********************************************************************
003034  000004                    TST6:   SCOPE
003036  012737  000100  001160            MOV     #100,$TIMES     ;;DO 100 ITERATIONS

003044  005077  176326                    CLR     @ASR            ;/CLEAR THE STATUS REGISTER.
003050  052777  000100  176320            BIS     #BIT6,@ASR      ;/SET BIT 6.
003056  012737  000100  001124            MOV     #BIT6,$GDDAT    ;/SET FOR ERROR TYPEOUT S/B.
003064  017737  176306  001126            MOV     @ASR,$BDDAT     ;/READ THE STATUS REGISTER.
003072  023737  001124  001126            CMP     $GDDAT,$BDDAT   ;/DID BIT 6 AND ONLY BIT 6 SET?
003100  001402                            BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.

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

003102  104002                            ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER
                                                                ;/BIT 6 FAILED TO BIT SET.

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

003104  000412                            BR      2$              ;/BR TO END SUBTEST.

003106  042777  000100  176262  1$:       BIC     #BIT6,@ASR      ;/TRY CLEARING BIT 6.
003114  005037  001124                    CLR     $GDDAT          ;/CLEAR S/B FOR TYPEOUT IF ANY.
003120  017737  176252  001126            MOV     @ASR,$BDDAT     ;/NOW READ IT BACK.
003126  001401                            BEQ     2$              ;/IF ZERO - NO ERROR!

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

003130  104002                            ERROR   2               ;/ERROR - CLOCK A STATUS REGISTER.
                                                                ;/BIT 6 FAILED TO CLEAR.

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

003132                          2$:
```

```
                                        ;/*
                                        ;;**********************************************************************
                                        ;*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
                                        ;*
                                        ;;**********************************************************************
003132  000004                          TST7:   SCOPE
003134  012737  000100  001160                  MOV     #100,$TIMES     ;;DO 100 ITERATIONS

003142  005077  176230                        CLR     @ASR            ;/CLEAR THE STATUS REGISTER.
003146  052777  000040  176222                BIS     #BIT5,@ASR      ;/SET BIT 5.
003154  012737  000040  001124                MOV     #BIT5,$GDDAT    ;/SET FOR ERROR TYPEOUT S/B.
003162  017737  176210  001126                MOV     @ASR,$BDDAT     ;/READ THE STATUS REGISTER.
003170  023737  001124  001126                CMP     $GDDAT,$BDDAT   ;/DID BIT 5 AND ONLY BIT 5 SET?
003176  001402                                BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.

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

003200  104002                                ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER
                                                                        ;/BIT 5 FAILED TO BIT SET.

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

003202  000412                                BR      2$              ;/BR TO END SUBTEST.

003204  042777  000040  176164  1$:           BIC     #BIT5,@ASR      ;/TRY CLEARING BIT 5.
003212  005037  001124                        CLR     $GDDAT          ;/CLEAR S/B FOR TYPEOUT IF ANY.
003216  017737  176154  001126                MOV     @ASR,$BDDAT     ;/NOW READ IT BACK.
003224  001401                                BEQ     2$              ;/IF ZERO - NO ERROR!

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

003226  104002                                ERROR   2               ;/ERROR - CLOCK A STATUS REGISTER.
                                                                        ;/BIT 5 FAILED TO cLEAR.

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

003230                                  2$:
```

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

                                        ;;*******************************************************************
003230  000004                          TST10:  SCOPE
003232  012737  000100  001160                  MOV     #100,$TIMES     ;;DO 100 ITERATIONS

003240  005077  176132                          CLR     @ASR            ;/CLEAR THE STATUS REGISTER.
003244  052777  000020  176124                  BIS     #BIT4,@ASR      ;/SET BIT 4.
003252  012737  000020  001124                  MOV     #BIT4,$GDDAT    ;/SET FOR ERROR TYPEOUT S/B.
003260  017737  176112  001126                  MOV     @ASR,$BDDAT     ;/READ THE STATUS REGISTER.
003266  023737  001124  001126                  CMP     $GDDAT,$BDDAT   ;/DID BIT 4 AND ONLY BIT 4 SET?
003274  001402                                  BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.

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

003276  104002                                  ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER
                                                                        ;/BIT 4 FAILED TO BIT SET.

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

003300  000412                                  BR      2$              ;/BR TO END SUBTEST.

003302  042777  000020  176066          1$:     BIC     #BIT4,@ASR      ;/TRY CLEARING BIT 4.
003310  005037  001124                          CLR     $GDDAT          ;/CLEAR S/B FOR TYPEOUT IF ANY.
003314  017737  176056  001126                  MOV     @ASR,$BDDAT     ;/NOW READ IT BACK.
003322  001401                                  BEQ     2$              ;/IF ZERO - NO ERROR!


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

003324  104002                                  ERROR   2               ;/ERROR - CLOCK A STATUS REGISTER.
                                                                        ;/BIT 4 FAILED TO CLEAR.


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

003326                                  2$:
```

```
                                        ;/*
                                        ;;**********************************************************************
                                        ;*TEST 11        *TEST THAT CLOCK A STATUS REGISTER BIT 3 CAN BE SET AND CLEARED
                                        ;*
                                        ;*CLOCK STATUS REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
                                        ;*F/FS OR GATES
                                        ;*
                                        ;;**********************************************************************
003326  000004                          TST11:  SCOPE
003330  012737  000100  001160                  MOV     #100,$TIMES     ;;DO 100 ITERATIONS

003336  005077  176034                          CLR     @ASR            ;/CLEAR THE STATUS REGISTER.
003342  052777  000010  176026                  BIS     #BIT3,@ASR      ;/SET BIT 3.
003350  012737  000010  001124                  MOV     #BIT3,$GDDAT    ;/SET FOR ERROR TYPEOUT S/B.
003356  017737  176014  001126                  MOV     @ASR,$BDDAT     ;/READ THE STATUS REGISTER.
003364  023737  001124  001126                  CMP     $GDDAT,$BDDAT   ;/DID BIT 3 AND ONLY BIT 3 SET?
003372  001402                                  BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.

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

003374  104002                                  ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER
                                                                        ;/BIT 3 FAILED TO BIT SET.

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

003376  000412                                  BR      2$              ;/BR TO END SUBTEST.

003400  042777  000010  175770  1$:     BIC     #BIT3,@ASR      ;/TRY CLEARING BIT 3.
003406  005037  001124                  CLR     $GDDAT          ;/CLEAR S/B FOR TYPEOUT IF ANY.
003412  017737  175760  001126          MOV     @ASR,$BDDAT     ;/NOW READ IT BACK.
003420  001401                          BEQ     2$              ;/IF ZERO - NO ERROR!

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

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

003422  104002                                  ERROR   2               ;/ERROR - CLOCK A STATUS REGISTER.
                                                                        ;/BIT 3 FAILED TO CLEAR.

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

003424                                  2$:
```

```
                                        ;/*
                                ;;**********************************************************************
                                ;*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
                                ;*
                                ;;**********************************************************************
003424  000004                  ST12:   SCOPE
003426  012737  000100  001160          MOV     #100,$TIMES     ;;DO 100 ITERATIONS

003434  005077  175736          CLR     @ASR            ;/CLEAR THE STATUS REGISTER.
003440  052777  000004  175730  BIS     #BIT2,@ASR      ;/SET BIT 2.
003446  012737  000004  001124  MOV     #BIT2,$GDDAT    ;/SET FOR ERROR TYPEOUT S/B.
003454  017737  175716  001126  MOV     @ASR,$BDDAT     ;/READ THE STATUS REGISTER.
003462  023737  001124  001126  CMP     $GDDAT,$BDDAT   ;/DID BIT 2 AND ONLY BIT 2 SET?
003470  001402                  BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.

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

003472  104002                  ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER
                                                        ;/BIT 2 FAILED TO BIT SET.

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

003474  000412                  BR      2$              ;/BR TO END SUBTEST.

003476  042777  000004  175672  1$:     BIC     #BIT2,@ASR      ;/TRY CLEARING BIT 2.
003504  005037  001124          CLR     $GDDAT          ;/CLEAR S/B FOR TYPEOUT IF ANY.
003510  017737  175662  001126  MOV     @ASR,$BDDAT     ;/NOW READ IT BACK.
003516  001401                  BEQ     2$              ;/IF ZERO - NO ERROR!

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

003520  104002                  ERROR   2               ;/ERROR - CLOCK A STATUS REGISTER.
                                                        ;/BIT 2 FAILED TO CLEAR.

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

003522                          2$:
```

```
                                        ;/*
                                ;;**********************************************************************
                                ;*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
                                ;*
                                ;;**********************************************************************
003522  000004                  ;ST13:  SCOPE
003524  012737  000100  001160          MOV     #100,$TIMES     ;;DO 100 ITERATIONS

003532  005077  175640          CLR     @ASR            ;/CLEAR THE STATUS REGISTER.
003536  052777  000002  175632  BIS     #BIT1,@ASR      ;/SET BIT 1.
003544  012737  000002  001124  MOV     #BIT1,$GDDAT    ;/SET FOR ERROR TYPEOUT S/B.
003552  017737  175620  001126  MOV     @ASR,$BDDAT     ;/READ THE STATUS REGISTER.
003560  023737  001124  001126  CMP     $GDDAT,$BDDAT   ;/DID BIT 1 AND ONLY BIT 1 SET?
003566  001402                  BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.

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

003570  104002                  ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER
                                                        ;/BIT 1 FAILED TO BIT SET.

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

003572  000412                  BR      2$              ;/BR TO END SUBTEST.

003574  042777  000002  175574  1$:     BIC     #BIT1,@ASR      ;/TRY CLEARING BIT 1.
003602  005037  001124          CLR     $GDDAT          ;/CLEAR S/B FOR TYPEOUT IF ANY.
003606  017737  175564  001126  MOV     @ASR,$BDDAT     ;/NOW READ IT BACK.
003614  001401                  BEQ     2$              ;/IF ZERO - NO ERROR!

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

003616  104002                  ERROR   2               ;/ERROR - CLOCK A STATUS REGISTER.
                                                        ;/BIT 1 FAILED TO CLEAR.

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

003620                          2$:
```

```
                                        ;/*
                                ;;**********************************************************************
                                ;*TEST 14        *TEST THAT CLOCK A STATUS REGISTER BIT 0 CAN BE SET AND CLEARED
                                ;*
                                ;*CLOCK STATUS REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
                                ;*F/FS OR GATES
                                ;*
                                ;;**********************************************************************
003620  000004                TST14:  SCOPE
003622  012737  000100  001160          MOV     #100,$TIMES     ;;DO 100 ITERATIONS

003630  005077  175542          CLR     @ASR            ;/CLEAR THE STATUS REGISTER.
003634  052777  000001  175534  BIS     #BIT0,@ASR      ;/SET BIT 0.
003642  012737  000001  001124  MOV     #BIT0,$GDDAT    ;/SET FOR ERROR TYPEOUT S/B.
003650  017737  175522  001126  MOV     @ASR,$BDDAT     ;/READ THE STATUS REGISTER.
003656  023737  001124  001126  CMP     $GDDAT,$BDDAT   ;/DID BIT 0 AND ONLY BIT 0 SET?
003664  001402                  BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.

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

003666  104002                          ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER
                                                                ;/BIT 0 FAILED TO BIT SET.

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

003670  000412                          BR      2$              ;/BR TO END SUBTEST.

003672  042777  000001  175476  1$:     BIC     #BIT0,@ASR      ;/TRY CLEARING BIT 0.
003700  005037  001124          CLR     $GDDAT                  ;/CLEAR S/B FOR TYPEOUT IF ANY.
003704  017737  175466  001126  MOV     @ASR,$BDDAT     ;/NOW READ IT BACK.
003712  001401                  BEQ     2$              ;/IF ZERO - NO ERROR!

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

003714  104002                          ERROR   2               ;/ERROR - CLOCK A STATUS REGISTER.
                                                                ;/BIT 0 FAILED TO CLEAR.

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

003716                          2$:

                                        .RADIX  8


                                ;;**********************************************************************
                                ;*TEST 15        *TEST THAT PATERN 125252 WILL SET AND CLEAR IN BUFFER REG.
                                ;;**********************************************************************
003716  000004                TST15:  SCOPE

003720  005077  175454          CLR     @ABR            ;/CLEAR THE BUFFER REG.
003724  012737  125252  001124  MOV     #125252,$GDDAT  ;/RECORD PATTERN: 125252 .
```

```
1107  003732  013777  001124  175440          MOV     $GDDAT,@ABR     ;/SET PATTERN IN BUFFER REG.
1108  003740  017737  175434  001126          MOV     @ABR,$BDDAT     ;/READ THE BUFFER REG.
1109
1110  003746  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;/DID THE PATTERN SET OK?
1111  003754  001402                          BEQ     1$              ;/YES-TRY CLEARING IT.
1112
1113                                  ;;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

1116  003756  104003                          ERROR   3               ;/ERROR PATTERN 125252 FAILED TO
1117                                                                  ;/SET PROPERLY IN BUFFER REG.
1118
                                    ;;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

1121  003760  000412                          BR      2$              ;/GOTO SCOPE LOOP.
1122
1123  003762  042777  125252  175410  1$:     BIC     #125252,@ABR    ;/TRY CLEARING PATTERN.
1124  003770  005037  001124                  CLR     $GDDAT          ;/EXPECT ZERO BACK.
1125  003774  017737  175400  001126          MOV     @ABR,$BDDAT     ;/READ BUFFER REG.,WAS IT ZERO?
1126  004002  001401                          BEQ     2$              ;/YES-NEXT TEST.
1127
1128
                                    ;;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

1131  004004  104003                          ERROR   3               ;/BUFFER REG. COULD NOT BE LOADED
1132                                                                  ;/TO A ZERO.
1133
                                    ;;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

1136  004006                         2$:
1137
1138
1139                                ;;**************************************************************
1140                                ;*TEST 16        *TEST THAT PATERN 052525 WILL SET AND CLEAR IN BUFFER REG.
1141                                ;;**************************************************************
1142  004006  000004                TST16:  SCOPE
1143
1144  004010  005077  175364                  CLR     @ABR            ;/CLEAR THE BUFFER REG.
1145  004014  012737  052525  001124          MOV     #052525,$GDDAT  ;/RECORD PATTERN: 052525
1146  004022  013777  001124  175350          MOV     $GDDAT,@ABR     ;/SET PATTERN IN BUFFER REG.
1147  004030  017737  175344  001126          MOV     @ABR,$BDDAT     ;/READ THE BUFFER REG.
1148
1149  004036  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;/DID THE PATTERN SET OK?
1150  004044  001402                          BEQ     1$              ;/YES-TRY CLEARING IT.
1151
1152
                                    ;;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

1155  004046  104003                          ERROR   3               ;/ERROR PATTERN 052525 FAILED TO
1156                                                                  ;/SET PROPERLY IN BUFFER REG.
1157
                                    ;;$$$$$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

1160  004050  000412                          BR      2$              ;/GOTO SCOPE LOOP.
```

```
004052 042777 052525 175320 1$:     BIC     #052525,@ABR       ;/TRY CLEARING PATTERN.
004060 005037 001124                CLR     $GDDAT             ;/EXPECT ZERO BACK.
004064 017737 175310 001126         MOV     @ABR,$BDDAT        ;/READ BUFFER REG.,WAS IT ZERO?
004072 001401                       BEQ     2$                 ;/YES-NEXT TEST.

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

004074 104003                       ERROR   3                  ;/BUFFER REG. COULD NOT BE LOADED
                                                               ;/TO A ZERO.

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

004076                      2$:



                                    .SBTTL  *
                                    .SBTTL  * PHASE 2 ADVANCED BASIC LOGIC TESTS
                                    .SBTTL  *


                            ;;*************************************************************
                            ;*TEST 17        *TEST THE LOW BYTE OPERATION OF CLOCK'S STATUS REGISTER

                            ;*
                            ;*WE CAN SUCCESSFULLY WRITE EVERY BIT IN STATUS REG A
                            ;*NOW LETS CHECK THE BYTE OPERATION OF THIS REGISTER.
                            ;*

                            ;;*************************************************************
004076 000004               TST17:  SCOPE
004100 012737 000050 001160         MOV     #50,$TIMES         ;;DO 50 ITERATIONS

004106 005077 175264                CLR     @ASR               ;MAKE SURE THE STATUS REGISTER IS CLEAR.
004112 112777 127677 175256         MOVB    #127677,@ASR       ;TRY WRITING ALL BITS IN THE
                                                               ;STATUS REGISTER. LOGIC SHOULD PREVENT IT
                                                               ;FROM BEING WRITTEN INTO BECAUSE
                                                               ;WE ARE USING A DATOB INSTRUCTION.

004120 017777 175252 175000         MOV     @ASR,@$BDDAT       ;NOW EXAMINE THE
                                                               ;STATUS REGISTER.
004126 013737 001126 001124         MOV     $BDDAT,$GDDAT      ;FIX $GDDAT FOR ERROR TYPEOUT IF
004134 105037 001125                CLRB    $GDDAT+1           ;ANY RROR HAS OCCURRED, UPPER BYTE CLEARED.

004140 105737 001127                TSTB    $BDDAT+1           ;ARE ANY BITS IN THE UPPER BYTE
                                                               ;OF THE STATUS REGISTER SET?
004144 001401                       BEQ     1$                 ;BRANCH NEXT TEST IF UPPER BYTE=0.


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

004146 104001                       ERROR   1                  ;ERROR - WROTE INTO UPPER BYTE OF
```

```
                                                        ;CLOCK'S STATUS WHEN
                                                        ;DOING A DATOB TO THE LOW BYTE.


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

004150                             1$:

                                   ;;*******************************************************************
                                   ;*TEST 20        *TEST THE HIGH BYTE OPERATION OF A'S STATUS REGISTER

                                   ;*
                                   ;*WE CAN SUCCESSFULLY WRITE EVERY BIT IN STATUS REG A
                                   ;*NOW LETS CHECK THE BYTE OPERATION OF THIS REGISTER.
                                   ;*

                                   ;;*******************************************************************
004150  000004                     TST20:  SCOPE
004152  012737  000050  001160             MOV     #50,$TIMES          ;;DO 50 ITERATIONS

004160  005077  175212                     CLR     @ASR                ;CLEAR THE STATUS REGISTER.

004164  005237  001376                     INC     ASR                 ;ADD #1 TO THE STATUS REGISTER'S ADDRESS
                                                                       ;SO THAT WE WILL BE WRITING INTO
                                                                       ;THE HIGH BYTE.

004170  112777  177213  175200             MOVB    #177213,@ASR        ;TRY WRITING ALL BITS IN THE STATUS
                                                                       ;REGISTER. LOGIC SHOULD PREVENT THE LOW
                                                                       ;BYTE OF THE STATUS REGISTER FROM
                                                                       ;BEING WRITTEN INTO BECAUSE WE ARE USING
                                                                       ;A DATOB INSTRUCTION WITH A00 SET.

004176  005337  001376                     DEC     ASR                 ;FIX ADDRESS OF THE STATUS REGISTER ADDR.
                                                                       ;SO WE CAN LOOK AT THE WHOLE WORD.
004202  017737  175170  001126             MOV     @ASR,$BDDAT         ;READ BACK WHAT THE STATUS REG. CONTAINS
004210  013737  001126  001124             MOV     $BDDAT,$GDDAT       ;FIX $GDDAT FOR ERROR TYPEOUT IF AN ERROR
004216  105037  001124                     CLRB    $GDDAT              ;OCCURRED, LOWER BYTE CLEARED.
004222  105737  001126                     TSTB    $BDDAT              ;IS LOWER BYTE CLEAR?
004226  001401                             BEQ     1$                  ;BR IF YES TO NEXT SUBTEST.


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

004230  104001                             ERROR   1                   ;ERROR - WROTE INTO LOWER BYTE
                                                                       ;OF CLOCKS STATUS REGISTER WHEN
                                                                       ;DOING A DATOB TO THE HIGH BYTE.

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

004232                             1$:

                                   ;;*******************************************************************
                                   ;*TEST 21        *TEST CLOCK'S COUNT REGISTER WITH 125252 PATTERN
                                   ;;*******************************************************************
```

```
                                TST21:  SCOPE
1269  004232  000004
1270  004234  012737  000100  001160        MOV     #100,$TIMES     ;;DO 100 ITERATIONS
1271
1272  004242  005077  175130                CLR     @ASR            ;SELECT MODE 0.
1273  004246  012777  125252  175124        MOV     #125252,@ABR    ;LOAD THE BUFFER REGISTER WITH
1274                                                                ;PATTERN 125252. IT WILL BE
1275                                                                ;TRANSFERRED TO THE COUNT REGISTER
1276                                                                ;SINCE THIS IS MODE 0.
1277  004254  052777  000001  175114        BIS     #BIT0,@ASR      ;SET GO BIT(ALLOWS BUFFER-COUNT REG XFER).
1278
1279  004262  012737  125252  001124        MOV     #125252,$GDDAT  ;SET EXPECTED TO PATTERN IN CASE OF
1280                                                                ;NEED OF ERROR TYPEOUT.
1281                                                                ;/-RDCLK-
1282  004270  017746  175102                MOV     @ASR,-(6)       ;/SAVE CSR
1283  004274  052777  004007  175074        BIS     #4007,@ASR      ;/SET MODE 3,DIS INTR OSC NO RATE
1284                                                                ;/THIS MUST BE DONE IN
1285                                                                ;/ORDER TO XFERR COUNTER
1286                                                                ;/TO BUFFER ON ST2.
1287  004302  052777  001000  175066        BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
1288  004310  012746  000010                MOV     #8.,-(SP)       ;/NOW GENERATE
1289  004314  052777  000400  175054  64$:  BIS     #BIT8,@ASR
1290  004322  005316                        DEC     (SP)            ;/EIGHT ST1 PULSES
1291  004324  001373                        BNE     64$
1292  004326  005726                        TST     (SP)+           ;/RESET STACK
1293  004330  017737  175044  001126        MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
1294                                                                ;/PREVIOUS COUNTER
1295  004336  012677  175034                MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
1296  004342  005737  001126                TST     $BDDAT          ;/RESTORE CSR
1297
1298  004346  023737  001124  001126        CMP     $GDDAT,$BDDAT   ;DID ALL THE BITS AND NO OTHER BITS
1299                                                                ;COME THROUGH?
1300  004354  001401                        BEQ     1$              ;BR IF YES TO NEXT TEST.
1301
1302

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

1305  004356  104005                        ERROR   5               ;DATA ERROR CLOCK - PATTERN "125252"
1306                                                                ;FAILED TO TRANSFER PROPERLY BETWEEN
1307                                                                ;BUFFER AND COUNT REGISTERS.
1308
1309

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

1312  004360                    1$:

1313
1314                            ;;**********************************************************************
1315                            ;*TEST 22        *TEST CLOCKS COUNTER REGISTER WITH 052525 PATTERN
1316                            ;;**********************************************************************
1317  004360  000004            TST22:  SCOPE
1318  004362  012737  000050  001160        MOV     #50,$TIMES      ;;DO 50 ITERATIONS
1319
1320  004370  005077  175002                CLR     @ASR            ;SELECT MODE 0.
1321  004374  012777  052525  174776        MOV     #052525,@ABR    ;LOAD THE BUFFER REGISTER WITH
1322                                                                ;PATTERN 052525. IT WILL BE
```

```
                                                        ;TRANSFERRED TO THE COUNT REGISTER
                                                        ;SINCE THIS IS MODE 0.
004402  052777  000001  174766          BIS     #BIT0,@#SR      ;SET BC BIT(ALLOWS BUFFER-COUNT REG XFER).

004410  012737  052525  001124          MOV     #052525,$GDDAT  ;SET EXPECTED TO PATTERN IN CASE OF
                                                        ;NEED OF ERROR TYPEOUT.
                                                        ;/-RDCLK-
004416  017746  174754                  MOV     @#SR,-(6)       ;/SAVE CSR
004422  052777  004007  174746          BIS     #4007,@#SR      ;/SET MODE 3,DIS INTR OSC NO RATE
                                                        ;/THIS MUST BE DONE IN
                                                        ;/ORDER TO xFERR COUNTER
                                                        ;/TO BUFFER ON ST2.
004430  052777  001000  174740          BIS     #BIT9,@#SR      ;/GENERATE ON ST2 PULSE
004436  012746  000010                  MOV     #8.,-(SP)       ;/NOW GENERATE
004442  052777  000400  174726  64$:    BIS     #BIT8,@#SR
004450  005316                          DEC     (SP)            ;/EIGHT ST1 PULSES
004452  001373                          BNE     64$
004454  005726                          TST     (SP)+           ;/RESET STACK
004456  017737  174716  001126          MOV     @#BR,$BDDAT     ;/READ THE PRESET BUFFER,
                                                        ;/PREVIOUS COUNTER
004464  012677  174706                  MOV     (6)+,@#SR       ;/CONTENTS ARE IN $BDDAT.
004470  005737  001126                  TST     $BDDAT          ;/RESTORE CSR

004474  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;DID ALL THE BITS AND NO OTHER BITS
                                                        ;COME THROUGH?
004502  001401                          BEQ     1$              ;BR IF YES TO NEXT TEST.


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

004504  104005                          ERROR   5               ;DATA ERROR CLOCK - PATTERN "052525"
                                                        ;FAILED TO TRANSFER PROPERLY BETWEEN
                                                        ;BUFFER AND COUNT REGISTERS.


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

004506                          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.
                                ;*
                                ;;**********************************************************************
004506  000004                  TST23:  SCOPE
004510  012737  000005  001160          MOV     #5,$TIMES       ;;DO 5 ITERATIONS

004516  005037  001124                  CLR     $GDDAT          ;EXPECTED DATA IS ZERO.
004522  012777  176377  174646          MOV     #176377,@#SR    ;SET ALL BITS IN THE STATUS REG.

004530  000005                          RESET                   ;SYSTEM INITIALIZE.
```

```
1377
1378  004532  017737  174640  001126          MOV     @#SR,$BDDAT     ;READ THE STATUS REG., ALL BITS SHOULD
1379                                                                    ;HAVE BEEN CLEARED BY INIT.
1380  004540  001402                          BEQ     1$              ;BR IF YES TO nEXT TEST.
1381
1382
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

1385
1386  004542  104002                          ERROR   2               ;ERROR - SYSTEM INIT FAILED TO CLEAR
1387                                                                  ;STATUS REGISTER CLOCK A.
1388
1389
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

1392  004544  000416                          BR      TST24           ;;
1393  004546                          1$:
1394  004546  005737  001444                  TST     EXS             ;TEST EXTERNAL SIGS?
1395  004552  001413                          BEQ     TST24           ;;
1396  004554  005737  001450                  TST     DWArF           ;;
1397  004560  001010                          BNE     TST24           ;;
1398  004562  052777  016002  174626          BIS     #BIT11!BIT12!BIT10!BIT1,@#DR2    ;ENABLE ST1,ST2 TO LATCH.
1399  004570  032777  000006  174616          BIT     #6,@#DR         ;ST1,ST2, OVERFLOW SET?
1400  004576  001401                          BEQ     TST24           ;;
1401
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

1404  004600  104006                          ERROR   6               ;INIT FAILED TO CLEAR
1405                                                                  ;ST1,ST2, AND/OR OVERFLOW
1406
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

1409
1410                                    ;;**********************************************************************
1411                                    ;*TEST 24        *TEST THAT INIT CLEARS BUFFER REGISTER
1412                                    ;*
1413                                    ;*THIS TEST IS DESIGNED TO SEE IF "INIT H"
1414                                    ;*CLEARS THE BUFFER REGISTER. WE ALREADY
1415                                    ;*KNOW IT CLEARS THE STATUS REG.
1416                                    ;*
1417                                    ;;**********************************************************************
1418  004602  000004                  TST24:  SCOPE
1419  004604  012737  000005  001160          MOV     #5,$TIMES       ;;DO 5 ITERATIONS
1420
1421  004612  005037  001124                  CLR     $GDDAT          ;CLEAR EXPECTED DATA.
1422  004616  012777  177777  174554          MOV     #177777,@#ABR   ;SET ALL BITS IN THE BUFFER REGISTER.
1423
1424  004624  000005                          RESET                   ;ISSUE SYSTEM INITIALIZE.
1425
1426  004626  017737  174546  001126          MOV     @#ABR,$BDDAT    ;READ THE BUFFER REGISTER, ALL BITS
1427                                                                  ;SHOULD HAVE BEEN CLEARED BY INIT.
1428  004634  001401                          BEQ     1$              ;BR IF YES TO NEXT SUBTEST.
1429
1430
```

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

1433  004636  104003                            ERROR   3               ;ERROR - SYSTEM INIT FAILED
1434                                                                    ;TO CLEAR BUFFER REGISTER A.
1435
1436
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

1439  004640                            1$:
1440  004640  005737  001444                   TST     EXS             ;TEST EXTERNAL SIGS?
1441  004644  001406                           BEQ     TST25           ;;
1442  004646  005737  001450                   TST     DWARF
1443  004652  001003                           BNE     TST25           ;;
1444  004654  052777  016000  174534           BIS     #BIT11!BIT12!BIT10,@DR2 ;ENABLE THEM
1445
1446                                    ;;**************************************************************
1447                                    ;*TEST 25        *TEST THE SETTING OF MAINTENANCE ST2 IN CLOCK BIT 15 TO SET
1448                                    ;;**************************************************************
1449  004662  000004                    TST25:  SCOPE
1450
1451  004664  005077  174506                   CLR     @ASR            ;2 CLEAR THE CSR
1452  004670  012777  000001  174500           MOV     #1,@ASR         ;SET THE GO BIT(ENABLES ST2 TO SET FLG).
1453  004676  052777  001000  174472           BIS     #BIT9,@ASR      ;SET MAINTENANCE ST2.
1454
1455  004704  005777  174466                   TST     @ASR            ;DID BIT15 (ST2 FLAG) SET?
1456  004710  100402                           BMI     1$              ;BR IF YES - NEXT TEST
1457
1458
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

1461  004712  104001                            ERROR   1               ;ERROR - MAINTENANCE ST2 (BIT9)
1462                                                                    ;DID NOT SET BIT15 (ST2 FLAG).
1463
1464
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

1467  004714  000424                           BR      TST26           ;;
1468  004716  005737  001444           1$:     TST     EXS             ;TEST EXTERNAL SIGNALS?
1469  004722  001411                           BEQ     2$
1470  004724  005737  001450                   TST     DWARF
1471  004730  001016                           BNE     TST26           ;;
1472  004732  032777  000004  174454           BIT     #BIT2,@DR       ;DID EXTERNAL ST2 GET SET?
1473  004740  001002                           BNE     2$
1474
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

1477  004742  104006                            ERROR   6               ;ST2 OUT NOT DETECTED
1478                                                                    ;BY TESTOR
1479
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

1482  004744  000410                           BR      TST26           ;;
1483
1484  004746  042777  100000  174422  2$:     BIC     #BIT15,@ASR     ;NOW TRY CLEARING ST2 FLAG.
```

```
004754  032777  100000  174414                  BIT     #BIT15,@ASR             ;DID IT CLEAR??
004762  001401                                  BEQ     TST26                   ;;

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

004764  104001                                  ERROR   1                       ;ST2 FLAG FAILED TO CLEAR.

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


                                ;;*************************************************************
                                ;*TEST 26          *TEST THAT ST1 FLAG SETS ON MAINTENANCE ST1
                                ;;*************************************************************
004766  000004                  TST26:  SCOPE

004770  005077  174402                  CLR     @ASR                    ;2 CLEAR THE CSR
004774  052777  000400  174374          BIS     #BIT8,@ASR              ;2 SET MAIN. ST1
005002  032777  002000  174366          BIT     #BIT10,@ASR             ;2 DID ST1 FLAG SET?
005010  001001                          BNE     1$                      ;2 YES - CONTINUE

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

005012  104001                          ERROR   1                       ;2 ST1 FLAG FAILED TO SET

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

005014  042777  002000  174354  1$:     BIC     #BIT10,@ASR             ;2 NOW TRY CLEARING IT.
005022  032777  002000  174346          BIT     #BIT10,@ASR             ;2 DID IT CLEAR?
005030  001401                          BEQ     TST27                   ;;

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

005032  104001                          ERROR   1                       ;2 ST1 FLAG FAILED
                                                                        ;2 TO CLEAR

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


                                ;;*************************************************************
                                ;*TEST 27          *TEST THAT BIT00 IN CLOCK STATUS REG. WILL SET WHEN BIT13 AND MAIN. ST2
                                ;;*************************************************************
005034  000004                  TST27:  SCOPE

005036  012777  020000  174332          MOV     #BIT13,@ASR             ;SET "ST2 ENB COUNTER" IN CLK  STATUS REG.
005044  052777  001000  174324          BIS     #BIT9,@ASR              ;GENERATE A MAINTENANCE ST2.
005052  032777  000001  174316          BIT     #BIT00,@ASR             ;DID BIT00 (GO) SET?
005060  001001                          BNE     1$                      ;BR IF YES - NEXT TEST.


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

005062  104001                          ERROR   1                       ;ERROR - BIT00 OF CLOCK'S STATUS REGISTER
                                                                        ;FAILED TO sET WHEN BIT13 WAS SET
```

```
                                                          ;AND A MAINTENANCE ST2 GENERATED.


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

005064  032777  020000  174304  1$:     BIT     #BIT13,@ASR     ;DID BIT 13 CLEAR ITSELF??
005072  001401                          BEQ     2$              ;YES GO AHEAD.

                                  ;;$$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
005074  104001                          ERROR   1               ;BIT 13 FAILED TO CLEAR.

                                  ;;$$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
005076  005077  174274          2$:     CLR     @ASR            ;LEAVE SUBTEST WITH CLOCK CLEAR.

                                        .SBTTL  *
                                        .SBTTL  *PHASE 3 COUNT TESTS
                                        .SBTTL  *

                                  ;;*********************************************************************
                                  ;*TEST 30       *TEST TO SEE IF THE COUNTER WILL INCREMENT
                                  ;;*********************************************************************
005102  000004                  TST30:  SCOPE

005104  005077  174266                  CLR     @ASR            ;CLEAR THE CSR.
005110  005077  174264                  CLR     @ABR            ;CLEAR THE BUFFER
005114  052777  000061  174254          BIS     #BIT5!BIT4!BIT0,@ASR ;SET RATE:ST1, GO.
005122  052777  000400  174246          BIS     #BIT8,@ASR      ;GENERATE A MAINTENANCE ST1.
                                                                ;DID THE CLOCK COUNT?
                                                                ;/-RDCLK-
005130  017746  174242                  MOV     @ASR,-(6)       ;/SAVE CSR
005134  052777  004007  174234          BIS     #4007,@ASR      ;/SET MODE 3,DIS INTR OSC NO RATE
                                                                ;/THIS MUST BE DONE IN
                                                                ;/ORDER TO XFERR COUNTER
                                                                ;/TO BUFFER ON ST2.
005142  052777  001000  174226          BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
005150  012746  000010                  MOV     #8.,-(SP)       ;/NOW GENERATE
005154  052777  000400  174214  64$:    BIS     #BIT8,@ASR
005162  005316                          DEC     (SP)            ;/EIGHT ST1 PULSES
005164  001373                          BNE     64$
005166  005726                          TST     (SP)+           ;/RESET STACK
005170  017737  174204  001126          MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
                                                                ;/PREVIOUS COUNTER
005176  012677  174174                  MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
005202  005737  001126                  TST     $BDDAT          ;/RESTORE CSR
005206  001001                          BNE     1$              ;YES, NEXT TEST.

                                  ;;$$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
005210  104006                          ERROR   6               ;CLOCK FAILED TO INCREMENT.
```

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

005212                                          1$:

                                                ;;*************************************************************************
                                                ;*TEST 31         *SEE IF CLOCK WILL COUNT UP FROM A ZERO BASE, RATE:ST1
                                                ;*
                                                ;*        NOTE:    IN THIS TEST, LOOP ON ERROR WILL CAUSE A LOOP
                                                ;*                 ON THE FAILING COUT PATTERN:
                                                ;*                 WHILE LOOP ON TEST WILL START THE TEST
                                                ;*                 AND THE CLOCK FROM ZERO TO THE FAILING COUNT.
                                                ;*
                                                ;;*************************************************************************
005212  000004                                  TST31:  SCOPE
005214  012737  000010  001160                          MOV     #10,$TIMES         ;;DO 10 ITERATIONS

005222  005077  174150                                  CLR     @ASR               ;CLEAR THE CSR.
005226  005077  174146                                  CLR     @ABR               ;CLEAR THE BUFFER REG
005232  012737  000000  001124                          MOV     #0,$GDDAT          ;CLEAR EXPECTED.
005240  012737  005404  001110                          MOV     #2$,$LPERR

005246  012777  000061  174122          1$:     MOV     #BIT5!BIT4!BIT0,@ASR ;START CLOCK, RATE:ST1.

005254  052777  000400  174114                          BIS     #BIT8,@ASR         ;GENERATE A MAINTENANCE ST1
                                                                                   ;CLOCK SHOULD COUNT ONCE.
005262  005737  001444                                  TST     EXS                ;TEST EXTERNAL SIGNALS?
005266  001411                                          BEQ     10$                ;NO
005270  005737  001450                                  TST     DWARF
005274  001053                                          BNE     TST32              ;;
005276  032777  000002  174110                          BIT     #BIT1,@DR          ;YES - DID ST1 GET SET?
005304  001002                                          BNE     10$                ;YES

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

005306  104006                                          ERROR   6                  ;ST1 OUT NOT DETECTED
                                                                                   ;BY TES!OR

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

005310  000445                                          BR      TST32              ;;
005312                                          10$:
                                                                                   ;/-RDCLK1-
005312  017746  174060                                  MOV     @ASR,-(SP)         ;/SAVE CSR CONTENTS.
005316  052777  000005  174052                          BIS     #5,@ASR            ;/SET TO mODE 2,GO
005324  042777  100000  174044                          BIC     #BIT15,@ASR        ;/CLR ST FLAG.
005332  052777  001000  174036                          BIS     #BIT9,@ASR         ;/GENERATE ST2 PULSE.
005340  017737  174034  001126                          MOV     @ABR,$BDDAT        ;/READ COUNT REG.
005346  052677  174024                                  BIS     (SP)+,@ASR         ;/RESTORE CSR.
005352  005737  001126                                  TST     $BDDAT             ;/PREVIOUS CONTENTS OF COUNT REG
                                                                                   ;/IN $BDDAT.
005356  005237  001124                                  INC     $GDDAT             ;COUNT=OLD COUNT+1
005362  013737  001124  001424                          MOV     $GDDAT,$TMP0       ;FOR ERROR TYPEOUT.
005370  023737  001124  001126                          CMP     $GDDAT,$BDDAT      ;COUNT READ=COUNT EXP'ED?
005376  001402                                          BEQ     2$                 ;YES - SEE IF WE'RE THROUGH.
```

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

005400  104011                                  ERROR   11                      ;CLOCK FAILED TO COUNT UP PROPERLY.

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

005402  000410                                  BR      3$                      ;GOTO SCOPE LOOP.

005404  005077  173766                  2$:     CLR     @ASR
005410  013777  001124  173762                  MOV     $GDDAT,@ABR
005416  005737  001124                          TST     $GDDAT                  ;ALL DONE?
005422  001311                                  BNE     1$                      ;NO DO NEXT INCREMENT.
005424                                  3$:

                                        ;;*****************************************************************
                                        ;*TEST 32        *TEST THAT OVERFLOW (CSR BIT07) WILL SET ON OVERFLOW
                                        ;;*****************************************************************
005424  000004                          TST32:  SCOPE

005426  005737  001444                          TST     EXS                     ;TESTOR MODE ENABLED??
005432  001410                                  BEQ     2$                      ;NO-THEN SKIP NEXT SECTION OF CODE.
005434  005737  001450                          TST     DWARF
005440  001042                                  BNE     TST33                   ;;

005442  052777  020002  173746                  BIS     #BIT13!BIT1,@DR2
005450  012700  000010                          MOV     #8.,R0                  ;SET TIME OUT NUMBER.
005454                                  2$:
005454  005077  173716                          CLR     @ASR                    ;CLEAR THE CSR
005460  012777  177777  173712                  MOV     #-1,@ABR                ;SET PRESET BUFFER TO ALL ONES.

005466  052777  000061  173702                  BIS     #BIT5!BIT4!BIT0,@ASR ;START CLOCK, RATE ST1.

005474  052777  000400  173674                  BIS     #BIT8,@ASR              ;COUNT CLOCK ONCE, OVERFLOW
                                                                                ;SHOULD OCCUR.
005502  105777  173670                          TSTB    @ASR                    ;DID OVERFLOW SET?
005506  100402                                  BMI     1$                      ;YES - THEN NEXT TEST

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

005510  104006                                  ERROR   6                       ;ERROR - OVERFLOW, CSR BIT07
                                                                                ;FAILED TO SET ON OVERFLOW
005512  000415                                  BR      TST33                   ;;
005514  005737  001444                  1$:     TST     EXS                     ;TEST EXTERNAL SIGNALS?
005520  001412                                  BEQ     TST33                   ;;
005522  005737  001450                          TST     DWARF
005526  001007                                  BNE     TST33                   ;;
005530  032777  000010  173656                  BIT     #BIT3,@DR
005536  001003                                  BNE     TST33                   ;;
005540  005300                                  DEC     R0                      ;DID WE ALLOW ENOUGH TIME??
005542  001364                                  BNE     1$                      ;NO-THEN WAIT.

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

```
005544 104006                    ERROR   6                ;OVERFLOW OUT NOT DETECTED
                                                          ;BY TESTOR

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


                          ;;*************************************************************
                          ;*TESt 33          *TEST THAT OVERFLOW WILL CLEAR THE GO BIT
                          ;;*************************************************************
005546 000004             TST33:  SCOPE

005550 005077 173622             CLR     @ASR             ;CLEAR THE CSR.

005554 012777 177777 173616      MOV     #-1,@ABR         ;PRESET CLOCK TO -1.

005562 052777 000061 173606      BIS     #BIT5!BIT4!BIT0,@ASR ;START CLOCK, RATE:ST1

005570 052777 000400 173600      BIS     #BIT8,@ASR       ;COUNT ONCE, OVERFLOW
                                                          ;SHOULD OCCUR CLEARING
                                                          ;ENABLE (CSR BIT00)

005576 032777 000001 173572      BIT     #BIT0,@ASR       ;DID THE ENABLE CLEAR?
005604 001401                    BEQ     1$               ;YES - NEXT TEST.


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

005606 104006                    ERROR   6                ;ERROR - OVERFLOW FAILED
                                                          ;TO CLEAR ENABLE (CSR BIT00)

005610                    1$:


                          ;;*************************************************************
                          ;*TEST 34          *TEST THAT GO BIT DOES NOT CLEAR ON OVERFLOW, IF MODE 1
                          ;;*************************************************************
005610 000004             TST34:  SCOPE

005612 005077 173560             CLR     @ASR             ;CLEAR THE CSR.
005616 012777 177777 173554      MOV     #-1,@ABR         ;PRESET BUFFER=ONE COUNT FROM OVERFLOW.
005624 052777 000063 173544      BIS     #63,@ASR         ;MODE 1, RATE:ST1, GO.

005632 052777 000400 173536      BIS     #BIT8,@ASR       ;GENERATE MAINTENANCE ST1.

005640 032777 000001 173530      BIT     #BIT0,@ASR       ;DID ENABLE (GO BIT) CLEAR?
005646 001001                    BNE     1$               ;NO (GOOD) NEXT TEST.

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

005650 104006                    ERROR   6                ;GO BIT CLEARED ON OVERFLOW
                                                          ;WHEN MODE 1 WAS SELECTED

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

```
                                                ;CLEAR THE CLOCK.
005652  005077  173520          1$:   CLR     @ASR
                                ;;*****************************************************************
                                ;*TEST 35        *TEST THE ABILITY OF CLOCK TO COUNT AT 1MHZ RATE
                                ;*
                                ;*THIS TEST IS DESIGNED TO TEST THE CLOCK'S ABILITY
                                ;*TO COUNT AT 1MHZ RATE.
                                ;*
                                ;;*****************************************************************
005656  000004                  TST35:  SCOPE
005660  012737  000005  001160          MOV     #5,$TIMES       ;;DO 5 ITERATIONS
005666  005077  173504          CLR     @ASR            ;/CLEAR CLOCK
005672  005077  173502          CLR     @ABR            ;/CLEAR PRESET BUFFER
005676  012777  000011  173472  MOV     #BIT0!10,@ASR   ;/START CLOCK, MODE0, RATE:1MHZ
005704  005000                  CLR     R0              ;/NOW WE'LL DO A LITTLE DELAY. THIS DELAY
005706  005200          1$:     INC     R0              ;/WILL AMOUNT TO APPROXIMATELY
005710  001376                  BNE     1$              ;/369 MS.
                                                        ;/-RDCLK-
005712  017746  173460          MOV     @ASR,-(6)       ;/SAVE CSR
005716  052777  004007  173452  BIS     #4007,@ASR      ;/SET MODE 3,DIS INTR OSC NO RATE
                                                        ;/THIS MUST BE DONE IN
                                                        ;/ORDER TO XFERR COUNTER
                                                        ;/TO BUFFER ON ST2.
005724  052777  001000  173444  BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
005732  012746  000010          MOV     #8.,-(SP)       ;/NOW GENERATE
005736  052777  000400  173432  64$:    BIS     #BIT8,@ASR
005744  005316                  DEC     (SP)            ;/EIGHT ST1 PULSES
005746  001373                  BNE     64$
005750  005726                  TST     (SP)+           ;/RESET STACK
005752  017737  173422  001126  MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
                                                        ;/PREVIOUS COUNTER
005760  012677  173412          MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
005764  005737  001126          TST     $BDDAT          ;/RESTORE CSR
005770  001004                  BNE     2$              ;/YES - NEXT TEST.
005772  105766  177776          TSTB    -2(6)           ;/AT HIGH RATE MAY HAVE HAD OVERFLOW
                                                        ;/NOTE: CSR HAD BEEN PUT ON STACK.
005776  100401                  BMI     2$              ;/NEXT TEST IF OVERFLOW.

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

006000  104006          ERROR   6                       ;/CLOCK FAILED TO COUNT AT
                                                        ;/RATE:1MHZ
```

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

006002  005077  173370           2$:     CLR     @ASR                    ;/CLEAR THE CLOCK.



                                  ;;*************************************************************
                                  ;*TEST 36        *TEST THE ABILITY OF CLOCK TO COUNT AT 100KHZ RATE

                                  ;*
                                  ;*THIS TEST IS DESIGNED TO TEST THE CLOCK'S ABILITY
                                  ;*TO COUNT AT 100KHZ RATE.
                                  ;*
                                  ;;*************************************************************
006006  000004                   TST36:  SCOPE
006010  012737  000005  001160           MOV     #5,$TIMES               ;;DO 5 ITERATIONS


006016  005077  173354                   CLR     @ASR                    ;/CLEAR CLOCK
006022  005077  173352                   CLR     @ABR                    ;/CLEAR PRESET BUFFER
006026  012777  000021  173342           MOV     #BIT0!20,@ASR           ;/START CLOCK, MODE0, RATE:100KHZ
006034  005000                           CLR     R0                      ;/NOW WE'LL DO A LITTLE DELAY. THIS DELAY

006036  005200                   1$:     INC     R0                      ;/WILL AMOUNT TO APPROXIMATELY
006040  001376                           BNE     1$                      ;/369 MS.

                                                                         ;/-RDCLK-
006042  017746  173330                   MOV     @ASR,-(6)               ;/SAVE CSR
006046  052777  004007  173322           BIS     #4007,@ASR              ;/SET MODE 3,DIS INTR OSC NO RATE
                                                                         ;/THIS MUST BE DONE IN
                                                                         ;/ORDER TO XFERR COUNTER
                                                                         ;/TO BUFFER ON ST2.
006054  052777  001000  173314           BIS     #BIT9,@ASR              ;/GENERATE ON ST2 PULSE
006062  012746  000010                   MOV     #8.,-(SP)               ;/NOW GENERATE
006066  052777  000400  173302   64$:    BIS     #BIT8,@ASR
006074  005316                           DEC     (SP)                    ;/EIGHT ST1 PULSES
006076  001373                           BNE     64$
006100  005726                           TST     (SP)+                   ;/RESET STACK
006102  017737  173272  001126           MOV     @ABR,$BDDAT             ;/READ THE PRESET BUFFER,
                                                                         ;/PREVIOUS COUNTER
006110  012677  173262                   MOV     (6)+,@ASR               ;/CONTENTS ARE IN $BDDAT.
006114  005737  001126                   TST     $BDDAT                  ;/RESTORE CSR
006120  001004                           BNE     2$                      ;/YES - NEXT TEST.
006122  105766  177776                   TSTB    -2(6)                   ;/AT HIGH RATE MAY HAVE HAD OVERFLOW
                                                                         ;/NOTE: CSR HAD BEEN PUT ON STACK.
006126  100401                           BMI     2$                      ;/NEXT TEST IF OVERFLOW.


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

006130  104006                           ERROR   6                       ;/CLOCK FAILED TO COUNT AT
                                                                         ;/RATE:100KHZ
```

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

1865
1866  006132  005077  173240            2$:     CLR     @ASR            ;/CLEAR THE CLOCK.
1867
1868
1869
1870                                    ;;**********************************************************************
1871                                    ;*TEST 37        *TEST THE ABILITY OF CLOCK TO COUNT AT 10KHZ RATE
1872
1873                                    ;*
1874                                    ;*THIS TEST IS DESIGNED TO TEST THE CLOCK'S ABILITY
1875                                    ;*TO COUNT AT 10KHZ RATE.
1876                                    ;*
1877                                    ;;**********************************************************************
1878  006136  000004                    $ST37:  SCOPE
1879  006140  012737  000005  001160            MOV     #5,$TIMES       ;;DO 5 ITERATIONS
1880
1881
1882  006146  005077  173224            CLR     @ASR            ;/CLEAR CLOCK
1883  006152  005077  173222            CLR     @ABR            ;/CLEAR PRESET BUFFER
1884  006156  012777  000031  173212    MOV     #BIT0!30,@ASR   ;/START CLOCK, MODE0, RATE:10KHZ
1885  006164  005000                    CLR     R0              ;/NOW WE'LL DO A LITTLE DELAY. THIS DELAY
1886
1887  006166  005200            1$:     INC     R0              ;/WILL AMOUNT TO APPROXIMATELY
1888  006170  001376                    BNE     1$              ;/369 MS.
1889
1890                                                            ;/-RDCLK-
1891  006172  017746  173200            MOV     @ASR,-(6)       ;/SAVE CSR
1892  006176  052777  004007  173172    BIS     #4007,@ASR      ;/SET MODE 3,DIS INTR OSC NO RATE
1893                                                            ;/THIS MUST BE DONE IN
1894                                                            ;/ORDER TO XFERR COUNTER
1895                                                            ;/TO BUFFER ON ST2.
1896  006204  052777  001000  173164    BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
1897  006212  012746  000010            MOV     #8.,-(SP)       ;/NOW GENERATE
1898  006216  052777  000400  173152  64$:     BIS     #BIT8,@ASR
1899  006224  005316                    DEC     (SP)            ;/EIGHT ST1 PULSES
1900  006226  001373                    BNE     64$
1901  006230  005726                    TST     (SP)+           ;/RESET STACK
1902  006232  017737  173142  001126    MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
1903                                                            ;/PREVIOUS COUNTER
1904  006240  012677  173132            MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
1905  006244  005737  001126            TST     $BDDAT          ;/RESTORE CSR
1906  006250  001004                    BNE     2$              ;/YES - NEXT TEST.
1907  006252  105766  177776            TSTB    -2(6)           ;/AT HIGH RATE MAY HAVE HAD OVERFLOW
1908                                                            ;/NOTE: CSR HAD BEEN PUT ON STACK.
1909  006256  100401                    BMI     2$              ;/NEXT TEST IF OVERFLOW.
1910
1911

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

1914  006260  104006                    ERROR   6               ;/CLOCK FAILED TO COUNT AT
1915                                                            ;/RATE:10KHZ
1916
```

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

006262  005077  173110            2$:     CLR     @ASR            ;/CLEAR THE CLOCK.



                                  ;;***************************************************************
                                  ;*TEST 40        *TEST THE ABILITY OF CLOCK TO COUNT AT 1KHZ RATE

                                  ;*
                                  ;*THIS TEST IS DESIGNED TO TEST THE CLOCK'S ABILITY
                                  ;*TO COUNT AT 1KHZ RATE.
                                  ;*
                                  ;;***************************************************************
006266  000004                    TST40:  SCOPE
006270  012737  000005  001160            MOV     #5,$TIMES       ;;DO 5 ITERATIONS


006276  005077  173074                    CLR     @ASR            ;/CLEAR CLOCK
006302  005077  173072                    CLR     @ABR            ;/CLEAR PRESET BUFFER
006306  012777  000041  173062            MOV     #BIT0!40,@ASR   ;/START CLOCK, MODE0, RATE:1KHZ
006314  005000                            CLR     R0              ;/NOW WE'LL DO A LITTLE DELAY. THIS DELAY

006316  005200                    1$:     INC     R0              ;/WILL AMOUNT TO APPROXIMATELY
006320  001376                            BNE     1$              ;/369 MS.

                                                                  ;/-RDCLK-
006322  017746  173050                    MOV     @ASR,-(6)       ;/SAVE CSR
006326  052777  004007  173042            BIS     #4007,@ASR      ;/SET MODE 3,DIS INTR OSC NO RATE
                                                                  ;/THIS MUST BE DONE IN
                                                                  ;/ORDER TO xFERR COUNTER
                                                                  ;/TO BUFFER ON ST2.
006334  052777  001000  173034            BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
006342  012746  000010                    MOV     #8.,-(SP)       ;/NOW GENERATE
006346  052777  000400  173022    64$:    BIS     #BIT8,@ASR
006354  005316                            DEC     (SP)            ;/EIGHT ST1 PULSES
006356  001373                            BNE     64$
006360  005726                            TST     (SP)+           ;/RESET STACK
006362  017737  173012  001126            MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
                                                                  ;/PREVIOUS COUNTER
006370  012677  173002                    MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
006374  005737  001126                    TST     $BDDAT          ;/RESTORE CSR
006400  001004                            BNE     2$              ;/YES - NEXT TEST.
006402  105766  177776                    TSTB    -2(6)           ;/AT HIGH RATE MAY HAVE HAD OVERFLOW
                                                                  ;/NOTE: CSR HAD BEEN PUT ON STACK.
006406  100401                            BMI     2$              ;/NEXT TEST IF OVERFLOW.


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

006410  104006                            ERROR   6               ;/CLOCK FAILED TO COUNT AT
                                                                  ;/RATE:1KHZ
```

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

006412  005077  172760          2$:     CLR     @ASR                    ;/CLEAR THE CLOCK.



                                        ;;***************************************************************
                                        ;*TEST 41        *TEST THE ABILITY OF CLOCK TO COUNT AT 100HZ RATE

                                        ;*
                                        ;*THIS TEST IS DESIGNED TO TEST THE CLOCK'S ABILITY
                                        ;*TO COUNT AT 100HZ RATE.
                                        ;*
                                        ;;***************************************************************
006416  000004                  TST41:  SCOPE
006420  012737  000005  001160          MOV     #5,$TIMES               ;;DO 5 ITERATIONS


006426  005077  172744                  CLR     @ASR                    ;/CLEAR CLOCK
006432  005077  172742                  CLR     @ABR                    ;/CLEAR PRESET BUFFER
006436  012777  000051  172732          MOV     #BIT0!50,@ASR           ;/START CLOCK, MODE0, RATE:100HZ
006444  005000                          CLR     R0                      ;/NOW WE'LL DO A LITTLE DELAY. THIS DELAY

006446  005200                  1$:     INC     R0                      ;/WILL AMOUNT TO APPROXIMATELY
006450  001376                          BNE     1$                      ;/369 MS.

                                                                        ;/-RDCLK-
006452  017746  172720                  MOV     @ASR,-(6)               ;/SAVE CSR
006456  052777  004007  172712          BIS     #4007,@ASR              ;/SET MODE 3,DIS INTR OSC NO RATE
                                                                        ;/THIS MUST BE DONE IN
                                                                        ;/ORDER TO XFERR COUNTER
                                                                        ;/TO BUFFER ON ST2.
006464  052777  001000  172704          BIS     #BIT9,@ASR              ;/GENERATE ON ST2 PULSE
006472  012746  000010                  MOV     #8.,-(SP)               ;/NOW GENERATE
006476  052777  000400  172672  64$:    BIS     #BIT8,@ASR
006504  005316                          DEC     (SP)                    ;/EIGHT ST1 PULSES
006506  001373                          BNE     64$
006510  005726                          TST     (SP)+                   ;/RESET STACK
006512  017737  172662  001126          MOV     @ABR,$BDDAT             ;/READ THE PRESET BUFFER.
                                                                        ;/PREVIOUS COUNTER
006520  012677  172652                  MOV     (6)+,@ASR               ;/CONTENTS ARE IN $BDDAT.
006524  005737  001126                  TST     $BDDAT                  ;/RESTORE CSR
006530  001004                          BNE     2$                      ;/YES - NEXT TEST.
006532  105766  177776                  TSTB    -2(6)                   ;/AT HIGH RATE MAY HAVE HAD OVERFLOW
                                                                        ;/NOTE: CSR HAD BEEN PUT ON STACK.
006536  100401                          BMI     2$                      ;/NEXT TEST IF OVERFLOW.


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

006540  104006                          ERROR   6                       ;/CLOCK FAILED TO COUNT AT
                                                                        ;/RATE:100HZ
```

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

2027
2028  006542  005077  172630            2$:     CLR     @ASR            ;/CLEAR THE CLOCK.
2029
2030
2031
2032                                    ;;*****************************************************************
2033                                    ;*TEST 42        *TEST THE ABILITY OF CLOCK TO COUNT AT LINEFREQ RATE
2034                                    ;*
2035                                    ;*
2036                                    ;*THIS TEST IS DESIGNED TO TEST THE CLOCK'S ABILITY
2037                                    ;*TO COUNT AT LINEFREQ RATE.
2038                                    ;*
2039                                    ;*****************************************************************
2040  006546  000004                    TST42:  SCOPE
2041  006550  012737  000005  001160            MOV     #5,$TIMES       ;;DO 5 ITERATIONS
2042
2043  006556  032777  010000  172354            BIT     #BIT12,@SWR     ;SHALL WE TEST
2044                                                                    ;LINE FREQ?
2045                                                                    ;SW12=1
2046  006564  001450                            BEQ     TST43           ;;
2047
2048  006566  005077  172604                    CLR     @ASR            ;/CLEAR CLOCK
2049  006572  005077  172602                    CLR     @ABR            ;/CLEAR PRESET BUFFER
2050  006576  012777  000071  172572            MOV     #BIT0!70,@ASR   ;/START CLOCK, MODE0, RATE:LINEFREQ
2051  006604  005000                            CLR     R0              ;/NOW WE'LL DO A LITTLE DELAY. THIS DELAY
2052
2053  006606  005200                    1$:     INC     R0              ;/WILL AMOUNT TO APPROXIMATELY
2054  006610  001376                            BNE     1$              ;/369 MS.
2055
2056                                                                    ;/-RDCLK-
2057  006612  017746  172560                    MOV     @ASR,-(6)       ;/SAVE CSR
2058  006616  052777  004007  172552            BIS     #4007,@ASR      ;/SET MODE 3,DIS INTR OSC NO RATE
2059                                                                    ;/THIS MUST BE DONE IN
2060                                                                    ;/ORDER TO xFERR COUNTER
2061                                                                    ;/TO BUFFER ON ST2.
2062  006624  052777  001000  172544            BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
2063  006632  012746  000010                    MOV     #8.,-(SP)       ;/NOW GENERATE
2064  006636  052777  000400  172532  64$:     BIS     #BIT8,@ASR
2065  006644  005316                            DEC     (SP)            ;/EIGHT ST1 PULSES
2066  006646  001373                            BNE     64$
2067  006650  005726                            TST     (SP)+           ;/RESET STACK
2068  006652  017737  172522  001126            MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
2069                                                                    ;/PREVIOUS COUNTER
2070  006660  012677  172512                    MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
2071  006664  005737  001126                    TST     $BDDAT          ;/RESTORE CSR
2072  006670  001004                            BNE     2$              ;/YES - NEXT TEST.
2073  006672  105766  177776                    TSTB    -2(6)           ;/AT HIGH RATE MAY HAVE HAD OVERFLOW
2074                                                                    ;/NOTE: CSR HAD BEEN PUT ON STACK.
2075  006676  100401                            BMI     2$              ;/NEXT TEST IF OVERFLOW.
2076
2077

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

```
006700  104006                          ERROR   6               ;/CLOCK FAILED TO COUNT AT
                                                                ;/RATE:LINEFREQ
                                ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

006702  005077  172470          2$:     CLR     @ASR            ;/CLEAR THE CLOCK.



                                ;;*****************************************************************
                                ;*TEST 43        *TEST THAT COUNTER DOESN'T COUNT WHEN "SLAVE IN" RATE IS SELECTED
                                ;;*****************************************************************
006706  000004                  TST43:  SCOPE
006710  012737  000020  001160          MOV     #20,$TIMES      ;;DO 20 ITERATIONS

006716  005077  172454                  CLR     @ASR            ;CLEAR CSR.
006722  005077  172452                  CLR     @ABR            ;CLEAR PRESET BUFFER.

006726  052777  000001  172442          BIS     #BIT0,@ASR      ;SET GO BIT, RATE:"SLAVE IN"

006734  052777  000400  172434          BIS     #BIT8,@ASR      ;GENERATE A MAINTENANCE ST1.
006742  005000                          CLR     R0

006744  005200                  1$:     INC     R0              ;NOW DO A SHORT DELAY.
006746  001376                          BNE     1$

                                                                ;/-RDCLK-
006750  017746  172422                  MOV     @ASR,-(6)       ;/SAVE CSR
006754  052777  004007  172414          BIS     #4007,@ASR      ;/SET MODE 3,DIS INTR OSC NO RATE
                                                                ;/THIS MUST BE DONE IN
                                                                ;/ORDER TO XFERR COUNTER
                                                                ;/TO BUFFER ON ST2.
006762  052777  001000  172406          BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
006770  012746  000010                  MOV     #8.,-(SP)       ;/NOW GENERATE
006774  052777  000400  172374  64$:    BIS     #BIT8,@ASR
007002  005316                          DEC     (SP)            ;/EIGHT ST1 PULSES
007004  001373                          BNE     64$
007006  005726                          TST     (SP)+           ;/RESET STACK
007010  017737  172364  001126          MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER.
                                                                ;/PREVIOUS COUNTER
007016  012677  172354                  MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
007022  005737  001126                  TST     $BDDAT          ;/RESTORE CSR
007026  001003                          BNE     2$              ;IF ANY COUNT, ERROR.
007030  105777  172342                  TSTB    @ASR            ;CHECK FOR OVERFLOW.
007034  100003                          BPL     3$              ;IF NO OVERFLOW, NEXT TEST

007036  005037  001124          2$:     CLR     $GDDAT          ;CLEAR FOR TYPEOUT, EXP'D ZERO.

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

007042  104011                          ERROR   11              ;CLOCK COUNTED WHEN RATE:"SLAVE IN"
                                                                ;WAS SELECTED.
```

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

007044  005077  172326              3$:     CLR     @ASR            ;CLEAR CLOCK'S CSR.


                                    ;;*************************************************************
                                    ;*TEST 44       *TEST THAT THE CLOCK WILL COUNT IN MODE 1
                                    ;;*************************************************************
007050  000004                      TST44:  SCOPE

007052  005077  172320                      CLR     @ASR            ;CLEAR THE CSR.
007056  012777  177777  172314              MOV     #-1,@ABR        ;PRESET BUFFER REG.
007064  052777  000013  172304              BIS     #13,@ASR        ;START CLOCK,RATE:1 MHZ,MODE 1.
007072  005000                              CLR     R0
007074  105200                      1$:     INCB    R0              ;NOW DO A SHORT DELAY SO  HAT THE
007076  001376                              BNE     1$              ;CLOCK CAN COUNT TO OVERFLOW.

007100  105777  172272                      TSTB    @ASR            ;OVERFLOW SHOULD HAVE SET.
007104  100401                              BMI     2$              ;GOTO NEXT TEST IF SO.


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

007106  104006                              ERROR   6               ;CLOCK FAILED TO COUNT UP AND SET
                                                                    ;OVERFLOW IN MODE 1.

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

007110                              2$:


                                            .SBTTL  *
                                            .SBTTL  *PHASE 4 CLOCK INTERRUPT TEST.
                                            .SBTTL  *


                                    ;;*************************************************************
                                    ;*TEST 45       *TEST THAT THE CLOCK WILL INTERRUPT ON OVERFLOW
                                    ;;*************************************************************
007110  000004                      TST45:  SCOPE
007112  012737  000020  001160              MOV     #20,$TIMES      ;;DO 20 ITERATIONS


007120  012746  000340                      MOV     #340,-(SP)      ;PUT PRIORITY ON STACK.
007124  012746  007132                      MOV     #64$,-(SP)      ;PUT RETURN ADDRESS ON STACK
007130  000002                              RTI                     ;DO AN RTI, PUTS PRIORITY IN CPU.
007132                              64$:

007132  005077  172240                      CLR     @ASR            ;CLEAR CLOCK'S CSR.
007136  012777  177777  172234              MOV     #-1,@ABR        ;SET PRESET BUFFER TO ALL ONES.

007144  012777  000161  172224              MOV     #161,@ASR       ;START CLOCK, RATE:ST1.
007152  052777  000400  172216              BIS     #BIT8,@ASR      ;GENERATE A MAINTENANCE ST1.
```

```
2187  007160  012777  007220  172214          MOV     #1$,@VECT1      ;SET INTERRUPT ADDR.
2188
2189  007166  012746  000000                  MOV     #0,-(SP)        ;PUT PRIORITY ON STACK.
2190  007172  012746  007200                  MOV     #65$,-(SP)      ;PUT RETURN ADDRESS ON STACK
2191  007176  000002                          RTI                     ;DO AN RTI, PUTS PRIORITY IN CPU.
2192  007200                          65$:
2193
2194  007200  000240                          NOP                     ;STALL TIME
2195
2196  007202  012746  000340                  MOV     #340,-(SP)      ;PUT PRIORITY ON STACK.
2197  007206  012746  007214                  MOV     #66$,-(SP)      ;PUT RETURN ADDRESS ON STACK
2198  007212  000002                          RTI                     ;DO AN RTI, PUTS PRIORITY IN CPU.
2199  007214                          66$:
2200
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

2203  007214  104007                          ERROR   7               ;CLOCK FAILED TO INTERRUPT.
2204
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

2207  007216  000402                          BR      2$
2208  007220                          1$:
2209  007220  062706  000004                  ADD     #4,SP                   ;/ADD #4 TO STACK POINTER.
2210  007224  005077  172146          2$:     CLR     @ASR            ;CLEAR THE CLOCK.
2211  007230  013777  001404  172144          MOV     VECTP,@VECT1    ;2 RESTORE VECTOR.
2212
2213                                    ;;**************************************************************
2214                                    ;*TEST 46       *TEST THAT ST2 WILL CAUSE AN INTERRUPT
2215                                    ;;**************************************************************
2216  007236  000004                  ST46:   SCOPE
2217
2218
2219  007240  012746  000340                  MOV     #340,-(SP)      ;PUT PRIORITY ON STACK.
2220  007244  012746  007252                  MOV     #64$,-(SP)      ;PUT RETURN ADDRESS ON STACK
2221  007250  000002                          RTI                     ;DO AN RTI, PUTS PRIORITY IN CPU.
2222  007252                          64$:
2223
2224  007252  005077  172120                  CLR     @ASR            ;CLEAR CLOCKS CSR.
2225  007256  012777  007332  172122          MOV     #1$,@VECT2      ;SET UP INTERRUPT VECTOR.
2226  007264  012777  040001  172104          MOV     #BIT14!BIT0,@ASR;SET "INT2",AND GO BIT.
2227  007272  052777  001000  172076          BIS     #BIT9,@ASR      ;GENERATE A MAINTENANCE ST2.
2228
2229  007300  012746  000000                  MOV     #0,-(SP)        ;PUT PRIORITY ON STACK.
2230  007304  012746  007312                  MOV     #65$,-(SP)      ;PUT RETURN ADDRESS ON STACK
2231  007310  000002                          RTI                     ;DO AN RTI, PUTS PRIORITY IN CPU.
2232  007312                          65$:
2233
2234  007312  000240                          NOP                     ;STALL TIME
2235
2236  007314  012746  000340                  MOV     #340,-(SP)      ;PUT PRIORITY ON STACK.
2237  007320  012746  007326                  MOV     #66$,-(SP)      ;PUT RETURN ADDRESS ON STACK
2238  007324  000002                          RTI                     ;DO AN RTI, PUTS PRIORITY IN CPU.
2239  007326                          66$:
2240
```

```
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
007326  104007                                  ERROR   7                   ;CLOCK FAILED TO INTERRUPT ON ST2.

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

007330  000402                                  BR      2$
007332                                  1$:
007332  062706  000004                          ADD     #4,SP                        ;/ADD #4 TO STACK POINTER.
007336  005077  172034                  2$:     CLR     @ASR                ;CLEAR CLOCK'S CSR.
007342  013777  001410  172036                  MOV     VECT2P,@VECT2       ;2 RESTORE VECTOR.

                                        ;;*****************************************************************
                                        ;*TEST 47               *TEST THAT ST1 WILL CAUSE AN INTERRUPT
                                        ;;*****************************************************************
007350  000004                          TST47:  SCOPE


007352  012746  000340                          MOV     #340,-(SP)          ;PUT PRIORITY ON STACK.
007356  012746  007364                          MOV     #64$,-(SP)          ;PUT RETURN ADDRESS ON STACK
007362  000002                                  RTI                         ;DO AN RTI, PUTS PRIORITY IN CPU.
007364                                  64$:
007364  005077  172006                          CLR     @ASR                ;2 CLEAR CSR
007370  012777  007436  172010                  MOV     #1$,@VECT2          ;2 SET UP INTR. VECTOR.
007376  052777  040400  171772                  BIS     #BIT14!BIT8,@ASR    ;2 INTR ENABLE AND ST1.

007404  012746  000000                          MOV     #0,-(SP)            ;PUT PRIORITY ON STACK.
007410  012746  007416                          MOV     #65$,-(SP)          ;PUT RETURN ADDRESS ON STACK
007414  000002                                  RTI                         ;DO AN RTI, PUTS PRIORITY IN CPU.
007416                                  65$:
007416  000240                                  NOP                         ;2 INTR. FROM HERE.

007420  012746  000340                          MOV     #340,-(SP)          ;PUT PRIORITY ON STACK.
007424  012746  007432                          MOV     #66$,-(SP)          ;PUT RETURN ADDRESS ON STACK
007430  000002                                  RTI                         ;DO AN RTI, PUTS PRIORITY IN CPU.
007432                                  66$:

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

007432  104007                                  ERROR   7                   ;2 CLOCK FAILED TO INTR. ON ST1.
007434  000402                                  BR      2$
007436                                  1$:
007436  062706  000004                          ADD     #4,SP                        ;/ADD #4 TO STACK POINTER.
007442  005077  171730                  2$:     CLR     @ASR                ;2 CLEAR CSR.
007446  013777  001410  171732                  MOV     VECT2P,@VECT2       ;2 RESTORE INTR. VECTOR.

                                                .SBTTL  *
                                                .SBTTL  *PHASE 5 ADVANCED TES!ING
                                                .SBTTL  *
```

```
                                ;;*********************************************************************
                                ;*TEST 50        *TEST THAT THE "FOR" BIT WILL SET ON 2 ST2'S
                                ;;*********************************************************************
007454  000004                  TST50:  SCOPE

007456  005077  171714                  CLR     @ASR            ;START WITH CSR CLEAR.
007462  005277  171710                  INC     @ASR            ;SET GO BIT.
007466  052777  001000  171702          BIS #BIT9,@ASR          ; GENERATE THE 1ST ST2 PULSE.
007474  052777  001000  171674          BIS     #BIT9,@ASR      ;GENERATE 2ND ST2 PULSE.
                                                                ;THIS SHOULD CAUSE FOR BIT TO SET.
007502  032777  010000  171666          BIT     #BIT12,@ASR     ;DID FOR BIT SET?
007510  001007                          BNE     1$              ;YES-THEN NEXT TEST.

007512  017737  171660  001126          MOV     @ASR,$BDDAT     ;RECORD CSR.
007520  012737  110001  001124          MOV     #BIT15!BIT12!BIT0,$GDDAT        ;RECORD S/B.

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

007526  104001                          ERROR   1               ;ERROR-"FOR" BIT FAILED TO SET ON
                                                                ;ON TWO SUCCESIVE ST2 PULSES.

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

007530                          1$:
                                ;;*********************************************************************
                                ;*TEST 51        *TEST THAT THE "FOR" BIT WILL SET ON 2 ST1'S
                                ;;*********************************************************************
007530  000004                  TST51:  SCOPE

007532  005077  171640                  CLR     @ASR            ;START WITH THE CSR CLEAR.
007536  005277  171634                  INC     @ASR            ;SET GO BIT.
007542  052777  000400  171626          BIS     #BIT08,@ASR     ;GENERATE AN ST1.
007550  052777  000400  171620          BIS     #BIT08,@ASR     ;GENERATE ANOTHER ST1.
                                                                ;AT THIS POINT THE "FOR" BIT SHOULD HAVE SET
007556  032777  010000  171612          BIT     #BIT12,@ASR     ;DID THE FOR BIT SET?
007564  001007                          BNE     TST52           ;.
007566  017737  171604  001126          MOV     @ASR,$BDDAT     ;RECORD CSR.
007574  012737  012001  001124          MOV     #BIT10!BIT12!BIT0,$GDDAT        ;RECORD S/B.

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

007602  104001                          ERROR   1               ;ERROR- "FOR" BIT FAILED TO SET ON
                                                                ;TWO SUCCESIVE ST1 PULSES.

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


                                ;;*********************************************************************
                                ;*TEST 52        *TEST THAT FOR BIT WILL SET ON TWO OVERFLOWS
                                ;;*********************************************************************
007604  000004                  TST52:  SCOPE
007606  012737  000002  001160          MOV     #2,$TIMES       ;;DO 2 ITERATIONS
```

```
007614 005077 171556              CLR     @ASR              ;START WITH CSR CLEAR.
007620 012777 177777 171552       MOV     #-1,@ABF          ;PRELOAD BUFFER REG.
007626 052777 000013 171542       BIS     #13,@ASR          ;START CLOCK,MODE 1,1MHZ.
007634 005000                     CLR     R0                ;NOW DO A SHORT DELAY.
007636 105200              1$:    INCB    R0                ;DURING THIS DELAY, THE CLOCK WILL
007640 001376                     BNE     1$                ;HAVE OVERFLOWED TWICE-
                                                            ;THIS SHOULD CAUSE THE FOR BIT TO SET.

007642 032777 010000 171526       BIT     #BIT12,@ASR       ;DID FOR SET?
007650 001007                     BNE     2$                ;YES-NEXT TEST.

007652 017737 171520 001126       MOV     @ASR,$BDDAT       ;NO-RECORD THE CSR.
007660 012737 010213 001124       MOV     #010213,$GDDAT    ;RECORD S/B.

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

007666 104001                     ERROR   1                 ;ERROR FOR BIT FAILED TO SET ON
                                                            ; TWO SUCCESSIVE OVERFLOWS.

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

007670                     2$:
                           ;;*************************************************************
                           ;*TEST 53        *TEST THAT FOR BIT WILL CLEAR IF GO BIT IS SET
                           ;;*************************************************************
007670 000004              TST53: SCOPE

007672 012777 000001 171476       MOV     #BIT0,@ASR        ;CLEAR CSR,SET GO BIT.
007700 052777 001000 171470       BIS     #BIT9,@ASR        ;SET 1ST ST2 PULSE.
007706 052777 001000 171462       BIS     #BIT9,@ASR        ;GENERATE 2ND ST2 PULSE.
                                                            ;FOR BIT SETS HERE.
007714 042777 100001 171454       BIC     #BIT0!BIT15,@ASR         ;CLEAR GO BITAND ST2 FLAG.

007722 052777 000001 171446       BIS     #BIT0,@ASR        ;SET THE "GO" BIT AGAIN -
                                                            ;SHOULD CLEAR FOR BIT.

007730 017737 171442 001126       MOV     @ASR,$BDDAT       ;READ THE CSR.

007736 012737 100001 001124       MOV     #100001,$GDDAT    ;RECORD WHAT CSR S/B.
007744 032737 010000 001126       BIT     #BIT12,$BDDAT     ;DID FOR BIT CLEAR?
007752 001401                     BEQ     1$                ;YES NEXT TEST.

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

007754 104001                     ERROR   1                 ;FOR BIT FALED TO CLEAR
                                                            ;WHEN "GO" BIT WAS SET.

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

007756 005077 171414       1$:    CLR     @ASR              ;CLEAR THE CSR.

                           ;;*************************************************************
                           ;*TEST 54        *TEST THAT WE CAN DISABLE THE INTERNAL OSC
```

```
                                ;;**********************************************************************
000004                  TST54:  SCOPE
012737  000005  001160          MOV     #5,$TIMES       ;;DO 5 ITERATIONS
                                ;
005077  171400                  CLR     @ASR            ;CLEAR THE CSR
005077  171376                  CLR     @ABR            ;CLEAR THE PRESET BUFFER
005037  001124                  CLR     $GDDAT          ;CLEAR EXPED.
                                ;
012777  004000  171362          MOV     #BIT11,@ASR     ;DISABLE THE INTERNAL OSC.
052777  000011  171354          BIS     #BIT3!BIT0,@ASR ;START CLOCK:RATE 1MHZ.
005000                          CLR     R0
105200                  1$:     INCB    R0              ;DELAY A SHORT TIME.
001376                          BNE     1$
                                ;/-RDCLK-
017746  171342                  MOV     @ASR,-(6)       ;/SAVE CSR
052777  004007  171334          BIS     #4007,@ASR      ;/SET MODE 3,DIS INTR OSC NO RATE
                                ;/THIS MUST BE DONe IN
                                ;/ORDER TO XFERR COUNTER
                                ;/TO BUFFER ON ST2.
052777  001000  171326          BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
012746  000010                  MOV     #8.,-(SP)       ;/NOW GENERATE
052777  000400  171314  64$:     BIS     #BIT8,@ASR
005316                          DEC     (SP)            ;/EIGHT ST1 PULSES
001373                          BNE     64$
005726                          TST     (SP)+           ;/RESET STACK
017737  171304  001126          MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
                                ;/PREVIOUS COUNTER
012677  171274                  MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
005737  001126                  TST     $BDDAT          ;/RESTORE CSR
001401                          BEQ     2$              ;NO - GOOd - NEXT TEST.

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

104011                          ERROR   11              ;CLOCK DISABLE INTERNAL
                                                        ;OSC. DID NOT WORK.

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

005077  171260          2$:     CLR     @ASR            ;CLEAR THE CSR.

                                ;;**********************************************************************
                                ;*TEST 55        *TEST THAT CLOCK CAN BE COUNTED USING MAINTENANCE OSC
                                ;;**********************************************************************
000004                  TST55:  SCOPE

005077  171252                  CLR     @ASR            ;CLEAR THE CSR.
005077  171250                  CLR     @ABR            ;CLEAR THE PRESET BUFFER.
052777  004000  171240          BIS     #BIT11,@ASR     ;DISABLE THE INTERNAL OSC.
052777  000011  171232          BIS     #BIT3!BIT0,@ASR ;START CLOCK, 1MHZ, GO.
012700  177754                  MOV     #-20.,R0        ;SET TO COUNT 20 TIMES

052777  000400  171220  1$:     BIS     #BIT8,@ASR      ;2 GENERATE 1 MAINTENANCE OSC.
                                                        ;NOTE: AT 1MHZ, IT TAKES 10
                                                        ;MAINT. OSC TO EQUAL 1 COUNT
```

```
2453  010156  005200                          INC     R0              ;DO 20 MAINTENANCE OSC.
2454  010160  001373                          BNE     1$
2455
2456                                                                  ;/-RDCLK-
2457  010162  017746  171210                  MOV     @#ASR,-(6)      ;/SAVE CSR
2458  010166  052777  004007  171202          BIS     #4007,@#ASR     ;/SET MODE 3,DIS INTR OSC NO RATE
2459                                                                  ;/THIS MUST BE DONE IN
2460                                                                  ;/ORDER TO XFERR COUNTER
2461                                                                  ;/TO BUFFER ON ST2.
2462  010174  052777  001000  171174          BIS     #BIT9,@#ASR     ;/GENERATE ON ST2 PULSE
2463  010202  012746  000010                  MOV     #8.,-(SP)       ;/NOW GENERATE
2464  010206  052777  000400  171162  64$:    BIS     #BIT8,@#ASR
2465  010214  005316                          DEC     (SP)            ;/EIGHT ST1 PULSES
2466  010216  001373                          BNE     64$
2467  010220  005726                          TST     (SP)+           ;/RESET STACK
2468  010222  017737  171152  001126          MOV     @#ABR,$BDDAT    ;/READ THE PRESET BUFFER,
2469                                                                  ;/PREVIOUS COUNTER
2470  010230  012677  171142                  MOV     (6)+,@#ASR      ;/CONTENTS ARE IN $BDDAT.
2471  010234  005737  001126                  TST     $BDDAT          ;/RESTORE CSR
2472  010240  001001                          BNE     2$              ;YES - NEXT TEST.
2473
2474

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

2477  010242  104011                          ERROR   11              ;ERROR COULD NOT COUNT USING
2478                                                                  ;MAINTENANCE OSC.
2479

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

2482  010244  005077  171126          2$:     CLR     @#ASR           ;CLEAR THE CSR.
2483
2484
2485
2486
2487                                    ;;*****************************************************************
2488                                    ;*TEST 56        *TEST THE CLOCK'S 1MHZ DIVIDER
2489                                    ;*
2490                                    ;*IN THIS TEST WE WILL CHECH OUT PART OF THE DIVIDER CHAIN LOGIC.
2491                                    ;*THERE ARE SEVERAL TESTS THAT ARE USED TO DO THIS,THIS TEST CHECKS
2492                                    ;*THAT 100,000 MAIN OSC PULSES GIVES US 10000. COUNTS AT 1MHZ RATE.
2493                                    ;*****************************************************************
2494  010250  000004                    TST56:  SCOPE
2495  010252  012737  000005  001160          MOV     #5,$TIMES       ;;DO 5 ITERATIONS
2496
2497                                                                  ;/-DIVCH-
2498  010260  012700  000012                  MOV     #10.,R0
2499  010264  005077  171106                  CLR     @#ASR
2500  010270  005077  171104                  CLR     @#ABR
2501  010274  052777  004000  171074          BIS     #BIT11,@#ASR            ;/DISABLE INTERNAL OSC.
2502  010302  052777  000017  171066          BIS     #7!10,@#ASR             ;/SET GO,RATE: 1MHZ.,MODE 3.
2503
2504  010310  012701  023420          1$:     MOV     #10000.,R1              ;/DO THAT MANY TIMES.
2505  010314  052777  000400  171054  2$:     BIS     #BIT8,@#ASR             ;/GENERATE AN OSC PULSE.
2506  010322  005301                          DEC     R1
```

```
2507  010324  001373                          BNE     2$
2508  010326  005300                          DEC     R0
2509  010330  001367                          BNE     1$
2510  010332  052777  001000  171036          BIS     #BIT9,@ASR      ;/ST2
2511  010340  012700  000010                  MOV     #8.,R0
2512  010344  052777  000400  171024  3$:     BIS     #BIT8,@ASR
2513  010352  005300                          DEC     R0
2514  010354  001373                          BNE     3$
2515
2516  010356  017737  171016  001126          MOV     @ABR,$BDDAT             ;/READ COUNT.
2517  010364  012737  023420  001424          MOV     #10000.,$TMP0           ;/EXPECT THESE MANY COUNTS.
2518  010372  023737  001424  001126          CMP     $TMP0,$BDDAT            ;/DID WE GET THEM??
2519  010400  001401                          BEQ     TST57           ;;
2520  010402  104011                          ERROR   11                      ;/ERROR  100,000. OSC PULSES
2521                                                                          ;/DID NOT GENERATE 10000.
2522                                                                          ;/COUNTS AT RATE .MHZ
2523
2524
2525
2526                                  ;;**********************************************************************
2527                                  ;*TEST 57        *TEST THE CLOCK'S 100KHZ DIVIDER
2528                                  ;*
2529                                  ;*IN THIS TEST WE WILL CHECH OUT PART OF THE DIVIDER CHAIN LOGIC.
2530                                  ;*THERE ARE SEVERAL TESTS THAT ARE USED TO DO THIS,THIS TEST CHECKS
2531                                  ;*THAT 100,000 MAIN OSC PULSES GIVES US 1000. COUNTS AT 100KHZ RATE.
2532                                  ;;**********************************************************************
2533  010404  000004                  TST57:  SCOPE
2534  010406  012737  000005  001160          MOV     #5,$TIMES       ;;DO 5 ITERATIONS
2535
2536                                                                  ;/-DIVCH-
2537  010414  012700  000012                  MOV     #10.,R0
2538  010420  005077  170752                  CLR     @ASR
2539  010424  005077  170750                  CLR     @ABR
2540  010430  052777  004000  170740          BIS     #BIT11,@ASR             ;/DISABLE INTERNAL OSC.
2541  010436  052777  000027  170732          BIS     #7!20,@ASR              ;/SET GO,RATE: 100KHZ.,MODE 3.
2542
2543  010444  012701  023420          1$:     MOV     #10000.,R1              ;/DO THAT MANY TIMES.
2544  010450  052777  000400  170720  2$:     BIS     #BIT8,@ASR              ;/GENERATE AN OSC PULSE.
2545  010456  005301                          DEC     R1
2546  010460  001373                          BNE     2$
2547  010462  005300                          DEC     R0
2548  010464  001367                          BNE     1$
2549  010466  052777  001000  170702          BIS     #BIT9,@ASR      ;/ST2
2550  010474  012700  000010                  MOV     #8.,R0
2551  010500  052777  000400  170670  3$:     BIS     #BIT8,@ASR
2552  010506  005300                          DEC     R0
2553  010510  001373                          BNE     3$
2554
2555  010512  017737  170662  001126          MOV     @ABR,$BDDAT             ;/READ COUNT.
2556  010520  012737  001750  001424          MOV     #1000.,$TMP0            ;/EXPECT THESE MANY COUNTS.
2557  010526  023737  001424  001126          CMP     $TMP0,$BDDAT            ;/DID WE GET THEM??
2558  010534  001401                          BEQ     TST60           ;;
2559  010536  104011                          ERROR   11                      ;/ERROR  100,000. OSC PULSES
2560                                                                          ;/DID NOT GENERATE 1000.
```

```
                                                        ;/COUNTS AT RATE 100KHZ
                                        ;;**************************************************************
                                        ;*TEST 60        *TEST THE CLOCK'S 10KHZ DIVIDER
                                        ;*
                                        ;*IN THIS TEST WE WILL CHECH OUT PART OF THE DIVIDER CHAIN LOGIC.
                                        ;*THERE ARE SEVERAL TESTS THAT ARE USED TO DO tHIS,THIS TEST CHECKS
                                        ;*THAT 100,000 MAIN OSC PULSES GIVES US 100. COUNTS AT 10KHZ RATE.
                                        ;;**************************************************************
010540  000004                        TST60:  SCOPE
010542  012737  000005  001160                MOV     #5,$TIMES       ;;DO 5 ITERATIONS
                                                                        ;/-DIVCH-
010550  012700  000012                        MOV     #10.,R0
010554  005077  170616                        CLR     @ASR
010560  005077  170614                        CLR     @ABR
010564  052777  004000  170604                BIS     #BIT11,@ASR             ;/DISABLE INTERNAL OSC.
010572  052777  000037  170576                BIS     #7!30,@ASR              ;/SET GO,RATE: 10KHZ.,MODE 3.

010600  012701  023420                1$:     MOV     #10000.,R1              ;/DO THAT MANY TIMES.
010604  052777  000400  170564        2$:     BIS     #BIT8,@ASR              ;/GENERATE AN OSC PULSE.
010612  005301                                DEC     R1
010614  001373                                BNE     2$
010616  005300                                DEC     R0
010620  001367                                BNE     1$
010622  052777  001000  170546                BIS     #BIT9,@ASR      ;/ST2
010630  012700  000010                        MOV     #8.,R0
010634  052777  000400  170534        3$:     BIS     #BIT8,@ASR
010642  005300                                DEC     R0
010644  001373                                BNE     3$

010646  017737  170526  001126                MOV     @ABR,$BDDAT             ;/READ COUNT.
010654  012737  000144  001424                MOV     #100.,$TMP0             ;/EXPECT THESE MANY COUNTS.
010662  023737  001424  001126                CMP     $TMP0,$BDDAT            ;/DID WE GET THEM??
010670  001401                                BEQ     TST61           ;;
010672  104011                                ERROR   11                      ;/ERROR  100,000. OSC PULSES
                                                                        ;/DID NOT GENERATE 100.
                                                                        ;/COUNTS AT RATE 10KHZ


                                        ;;**************************************************************
                                        ;*TEST 61        *TESt THE CLOCK'S 1KHZ DIVIDER
                                        ;*
                                        ;*IN THIS TEST WE WILL CHECH OUT PART OF THE DIVIDER CHAIN LOGIC.
                                        ;*THERE ARE SEVERAL TESiS THAT ARE USED TO DO THIS,THIS TEST CHECKS
                                        ;*THAT 100,000 MAIN OSC PULSES GIVES US 10. COUNTS AT 1KHZ RATE.
                                        ;;**************************************************************
010674  000004                        TST61:  SCOPE
010676  012737  000005  001160                MOV     #5,$TIMES       ;;DO 5 ITERATIONS
```

```
                                                        ;/-DIVCH-
010704  012700  000012                MOV     #10.,R0
010710  005077  170462                CLR     @ASR
010714  005077  170460                CLR     @ABR
010720  052777  004000  170450        BIS     #BIT11,@ASR             ;/DISABLE INTERNAL OSC.
010726  052777  000047  170442        BIS     #7!40,@ASR              ;/SET GO,RATE: 1KHZ.,MODE 3.

010734  012701  023420          1$:   MOV     #10000.,R1              ;/DO THAT MANY TIMES.
010740  052777  000400  170430  2$:   BIS     #BIT8,@ASR              ;/GENERATE AN OSC PULSE.
010746  005301                        DEC     R1
010750  001373                        BNE     2$
010752  005300                        DEC     R0
010754  001367                        BNE     1$
010756  052777  001000  170412        BIS     #BIT9,@ASR      ;/ST2
010764  012700  000010                MOV     #8.,R0
010770  052777  000400  170400  3$:   BIS     #BIT8,@ASR
010776  005300                        DEC     R0
011000  001373                        BNE     3$

011002  017737  170372  001126        MOV     @ABR,$BDDAT             ;/READ COUNT.
011010  012737  000012  001424        MOV     #10.,$TMP0              ;/EXPECT THESE MANY COUNTS.
011016  023737  001424  001126        CMP     $TMP0,$BDDAT            ;/DID WE GET THEM??
011024  001401                        BEQ     TST62           ;;
011026  104011                        ERROR   11                      ;/ERROR  100,000. OSC PULSES
                                                                        ;/DID NOT GENERATE 10.
                                                                        ;/COUNTS AT RATE 1KHZ


                                ;;*********************************************************************
                                ;*TEST 62        *TEST THE CLOCK'S 100HZ DIVIDER
                                ;*
                                ;*IN THIS TEST WE WILL CHECH OUT PART OF THE DIVIDER CHAIN LOGIC.
                                ;*THERE ARE SEVERAL TESTS THAT ARE USED TO DO THIS,THIS TEST CHECKS
                                ;*THAT 100,000 MAIN OSC PULSES GIVES US 1 COUNTS AT 100HZ RATE.
                                ;;*********************************************************************
011030  000004                TST62:  SCOPE
011032  012737  000005  001160        MOV     #5,$TIMES               ;;DO 5 ITERATIONS

                                                        ;/-DIVCH-
011040  012700  000012                MOV     #10.,R0
011044  005077  170326                CLR     @ASR
011050  005077  170324                CLR     @ABR
011054  052777  004000  170314        BIS     #BIT11,@ASR             ;/DISABLE INTERNAL OSC.
011062  052777  000057  170306        BIS     #7!50,@ASR              ;/SET GO,RATE: 100HZ.,MODE 3.

011070  012701  023420          1$:   MOV     #10000.,R1              ;/DO THAT MANY TIMES.
011074  052777  000400  170274  2$:   BIS     #BIT8,@ASR              ;/GENERATE AN OSC PULSE.
011102  005301                        DEC     R1
011104  001373                        BNE     2$
011106  005300                        DEC     R0
011110  001367                        BNE     1$
011112  052777  001000  170256        BIS     #BIT9,@ASR      ;/ST2
011120  012700  000010                MOV     #8.,R0
```

```
2669  011124  052777  000400  170244  3$:     BIS     #BIT8,@ASR
2670  011132  005300                          DEC     R0
2671  011134  001373                          BNE     3$
2672
2673  011136  017737  170236  001126          MOV     @ABR,$BDDAT             ;/READ COUNT.
2674  011144  012737  000001  001424          MOV     #1,$TMP0                ;/EXPECT THESE MANY COUNTS.
2675  011152  023737  001424  001126          CMP     $TMP0,$BDDAT            ;/DID WE GET THEM??
2676  011160  001401                          BEQ     TST63           ;;
2677  011162  104011                          ERROR   11                      ;/ERROR  100,000. OSC PULSES
2678                                                                          ;/DID NOT GENERATE 1
2679                                                                          ;/COUNTS AT RATE 100HZ
2680
2681
2682
2683                                    ;;**********************************************************************
2684                                    ;*TEST 63        *TEST THE CLOCK'S MODE 2 OPERATION
2685                                            ;*
2686                                            ;*IN THIS TEST WE'LL CHECK MODE 2 OPERATION
2687                                            ;*MODE 2: EXTERNAL EVENTS TIMING MODE
2688                                            ;*SETTING THE GO BIT CAUSES THE COUNTER TO BEGIN COUNTING FROM
2689                                            ;*ZERO AND TO FREE-RUN UNTIL THE GO BIT IS WRITTEN
2690                                            ;*TO A ZERO THE COUNTER WILL CONTINUE COUNTING AFTER
2691                                            ;*OVERFLOW. AN EXTERNAL PULSE FROM SCHMITZ TRIGGER 2
2692                                            ;*(WHEN ST2 GO ENABLE IS A "0") CAUSES DATA TO
2693                                            ;*TRANSFER FROM THE COUNTER TO THE BUFFER/PRESET REG.
2694                                            ;*WHILE THE COUNTER CONTINUES TO RUN.
2695                                            ;*
2696                                            ;*TO TEST THIS MODE, WE'LL DISABLE THE INTERNAL OSC AND USE
2697                                            ;*MAINTENANCE OSC PULSES AS WELL AS A MAINTENANCE
2698                                            ;*ST2.
2699                                            ;*
2700                                    ;;**********************************************************************
2701  011164  000004                  TST63:  SCOPE
2702  011166  012737  000020  001160          MOV     #20,$TIMES              ;;DO 20 ITERATIONS
2703
2704  011174  005077  170176                  CLR     @ASR                    ;CLEAR THE CSR.
2705  011200  005077  170174                  CLR     @ABR                    ;CLEAR THE PRESET REG.
2706  011204  012777  000060  170164          MOV     #60,@ASR                ;SET RATE ST1
2707  011212  052777  000005  170156          BIS     #5,@ASR                 ;START CLOCK MODE 2.
2708
2709  011220  012700  177776          1$:     MOV     #-2.,R0                 ;SET TO GIVE 2 ST1 PULSES.
2710  011224  052777  000400  170144  2$:     BIS     #BIT8,@ASR              ; GENERATE AN ST1 PULSE
2711  011232  005200                          INC     R0
2712  011234  001373                          BNE     2$                      ;IF NOT DONE 2 TIMES, LOOP.
2713
2714  011236                          3$:
2715  011236  052777  001000  170132          BIS     #BIT9,@ASR              ;HERE'S THE BIGGIE! AN ST2 HAS BEEN GENERATED
2716  011244  012737  000002  001124          MOV     #2,$GDDAT               ;THE PRESET BUFFER SHOULD BE 2.
2717  011252  017737  170122  001126          MOV     @ABR,$BDDAT             ;READ THE PRESET BUFFER.
2718  011260  023737  001126  001124          CMP     $BDDAT,$GDDAT           ;DID A COUNTER TO PRESET BUFFER OCCUR?
2719  011266  001402                          BEQ     4$                      ;YES - NEXT SUBTEST.
2720
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

```
011270 104005                           ERROR   5               ;A COUNTER TO PRESET BUFFER DID NOT
                                                                ;HAPPEN PROPERLY.

                                ::$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

011272 000427                           BR      5$

011274                          4$:
                                                                ;/-RDCLK1-
011274 017746 170076                    MOV     @ASR,-(SP)      ;/SAVE CSR CONTENTS.
011300 052777 000005 170070             BIS     #5,@ASR         ;/SET TO MODE 2,GO
011306 042777 100000 170062             BIC     #BIT15,@ASR     ;/CLR ST FLAG.
011314 052777 001000 170054             BIS     #BIT9,@ASR      ;/GENERATE ST2 PULSE.
011322 017737 170052 001126             MOV     @ABR,$BDDAT     ;/READ COUNT REG.
011330 052677 170042                    BIS     (SP)+,@ASR      ;/RESTORE CSR.
011334 005737 001126                    TST     $BDDAT          ;/PREVIOUS CONTENTS OF COUNT REG
                                                                ;/IN $BDDAT.
011340 023737 001124 001126             CMP     $GDDAT,$BDDAT   ;WAS THE COUNTER ACCIDENTLY ZEROED?
011346 001401                           BEQ     5$              ;NO - NEXT TEST.

                                ::$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

011350 104005                           ERROR   5               ;THE COUNT REGISTER SHOULD NOT
                                                                ;HAVE BEEN EFFECTED BY THE ST2
                                                                ;IN MODE 2.

                                ::$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

011352                          5$:
011352 042777 000070 170016             BIC     #70,@ASR        ;CLEAR THE RATE BITS.
011360 005037 001124                    CLR     $GDDAT
011364 052777 000400 170004             BIS     #BIT8,@ASR      ;SET ST1 SIM.
                                                                ;/-RDCLK1-
011372 017746 170000                    MOV     @ASR,-(SP)      ;/SAVE CSR CONTENTS.
011376 052777 000005 167772             BIS     #5,@ASR         ;/SET TO MODE 2,GO
011404 042777 100000 167764             BIC     #BIT15,@ASR     ;/CLR ST FLAG.
011412 052777 001000 167756             BIS     #BIT9,@ASR      ;/GENERATE ST2 PULSE.
011420 017737 167754 001126             MOV     @ABR,$BDDAT     ;/READ COUNT REG.
011426 052677 167744                    BIS     (SP)+,@ASR      ;/RESTORE CSR.
011432 005737 001126                    TST     $BDDAT          ;/PREVIOUS CONTENTS OF COUNT REG
                                                                ;/IN $BDDAT.
011436 001401                           BEQ     TST64           ;;

                                ::$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

011440 104001                           ERROR   1               ;CLOCK COUNT REG SHOULD BE ZERO

                                ::$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$



                                ;;***********************************************************************
                                ;*TEST 64        *TEST THE CLOCK'S MODE 3 OPERATION
                                        ;*
```

J06


```
                                        ;*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.
                                        ;*
                                ;;*****************************************************************
011442  000004                  TST64:  SCOPE
011444  012737  000020  001160          MOV     #20,$TIMES          ;;DO 20 ITERATIONS

011452  005077  167720                  CLR     @ASR                ;CLEAR THE CSR.
011456  005077  167716                  CLR     @ABR                ;CLEAR THE BUFFER REG.
011462  012777  000060  167706          MOV     #60,@ASR            ;SET RATE: ST1
011470  052777  000007  167700          BIS     #7,@ASR             ;START CLOCK: MODE 3

011476  012700  177776          1$:     MOV     #-2.,R0             ;SET TO GIVE 2 ST1 PULSES
011502  052777  000400  167666  2$:     BIS     #BIT8,@ASR          ;GENERATE AN ST1 PULSE
011510  005200                          INC     R0
011512  001373                          BNE     2$                  ;IF NOT DONE 2 TIMES, LOOP.

011514                          3$:
011514  052777  001000  167654          BIS     #BIT9,@ASR          ;HERE'S THE BIGGIE! AN ST2 HAS BEEN GENERATED
011522  012737  000002  001124          MOV     #2,$GDDAT           ;THE PRESET BUFFER SHOULD BE 2.
011530  017737  167644  001126          MOV     @ABR,$BDDAT         ;READ THE PRESET BUFFER.
011536  023737  001126  001124          CMP     $BDDAT,$GDDAT       ;DID A COUNTER TO PRESET BUFFER OCCUR?
011544  001402                          BEQ     4$                  ;YES - NEXT SUBTEST.

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

011546  104005                          ERROR   5                   ;A COUNTER TO PRESET BUFFER DID NOT
                                                                    ;HAPPEN PROPERLY.

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

011550  000440                          BR      TST65               ;;

011552  005037  001124          4$:     CLR     $GDDAT              ;EXPECT ZERO BACK FROM COUNT REG.
                                                                    ;/-RDCLK1-
011556  017746  167614                  MOV     @ASR,-(SP)          ;/SAVE CSR CONTENTS.
011562  052777  000005  167606          BIS     #5,@ASR             ;/SET TO MODE 2,GO
011570  042777  100000  167600          BIC     #BIT15,@ASR         ;/CLR ST FLAG.
011576  052777  001000  167572          BIS     #BIT9,@ASR          ;/GENERATE ST2 PULSE.
011604  017737  167570  001126          MOV     @ABR,$BDDAT         ;/READ COUNT REG.
011612  052677  167560                  BIS     (SP)+,@ASR          ;/RESTORE CSR.
011616  005737  001126                  TST     $BDDAT              ;/PREVIOUS CONTENTS OF COUNT REG
                                                                    ;/IN $BDDAT.
011622  001402                          BEQ     5$                  ;IF SO - NEXT TEST.

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

011624  104005                          ERROR   5                   ;THE CLOCK FORGOT TO ZERO THE COUNT
                                                                    ;REG. AFTER AN ST2 OCCURRED ON
                                                                    ;A MODE 3 COUNT.

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

```
2831  011626  000411                          BR      TST65           ;;
2832  011630                          5$:
2833  011630  005077  167542                  CLR     @ASR            ;NOW TRY CLEARING THE CSR.
2834  011634  017737  167536  001126          MOV     @ASR,$BDDAT     ;READ THE CSR - DID IT CLEAR?
2835  011642  001403                          BEQ     TST6$           ;;
2836  011644  005037  001124                  CLR     $GDDAT          ;NO - RECORD S/B.
2837

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

2840  011650  104002                          ERROR   2               ;CSR FAILED TO CLEAR
2841

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

2844
2845                                    ;;**************************************************************
2846                                    ;*TEST 65        *DWARF TEST OF OVERFLOW OUT,ST2 IN AND OL.,AND ST1 IN
2847                                    ;*
2848                                    ;* IN THIS TEST WE WILL TEST OVERFLOW OUT,ST2 IN ST2 OUT, AND ST1
2849                                    ;* IN.   TO DO THIS WE CONNECTED THEM TOGETHER WITH THE DWARF MODULE.
2850                                    ;*  NOTE: THIS TEST ONLY RAN IF DWARF MODE SELECTED.
2851                                    ;*
2852                                    ;;**************************************************************
2853  011652  000004                  TST65:  SCOPE
2854  011654  012737  000010  001160          MOV     #10,$TIMES      ;;DO 10 ITERATIONS
2855  011662  012737  012076  001110          MOV     #1$,$LPERR
2856  011670  012737  012076  001106          MOV     #1$,$LPADR
2857
2858  011676  005737  001450                  TST     DWARF           ;DWARF MODE??
2859  011702  001527                          BEQ     TST66           ;;
2860                                                                  ;TO RUN DWARF MODE L&S AT 250.
2861  011704  104401  011712                  TYPE    ,65$            ;;TYPE ASCIZ STRING
2862  011710  000424                          BR      64$             ;;GET OVER THE ASCIZ
2863                                  ;;65$:  .ASCIZ  <200><7>"DWARF: ALL SWITCHES OFF,S2-1,S2-4 ON"
2864  011762                          64$:
2865  011762  104401  011770                  TYPE    ,67$            ;;TYPE ASCIZ STRING
2866  011766  000417                          BR      66$             ;;GET OVER THE ASCIZ
2867                                  ;;67$:  .ASCIZ  <200>"ST1 IN (TTL),SLOPE OUT (-)."
2868  012026                          66$:
2869  012026  104401  012034                  TYPE    ,69$            ;;TYPE ASCIZ STRING
2870  012032  000417                          BR      68$             ;;GET OVER THE ASCIZ
2871                                  ;;69$:  .ASCIZ  <200>"ST2 IN (TTL),SLOPE OUT (-)."<7>
2872  012072                          68$:
2873  012072  004737  014744                  JSR PC,ANY2
2874  012076  005077  167274          1$:     CLR     @ASR            ;CLEAR CSR
2875  012102  005737  001450                  TST     DWARF
2876  012106  001425                          BEQ     TST66           ;;
2877  012110  012777  177777  167262          MOV     #-1,@ABR        ;PRESET FOR OVERFLOW.
2878  012116  012777  000011  167252          MOV     #11,@ASR        ;SET 1 MHZ,GO.
2879  012124  000240                          NOP
2880  012126  000240                          NOP
2881  012130  000240                          NOP
2882  012132  000240                          NOP                     ;ALLOW TIME FOR OVERFLOW.
2883  012134  012737  102210  001124          MOV     #102210,$GDDAT  ;EXPECT ST1 AND ST2 FLAGS TO BE SET.
2884  012142  017737  167230  001126          MOV     @ASR,$BDDAT     ;READ CSR
```

```
012150  023737  001124  001126          CMP     $GDDAT,$BDDAT       ;OK?
012156  001401                          BEQ     TST66               ;;

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

012160  104001                          ERROR   1                   ;ST1 AND/OR ST2 FLAG FAILED TO SET

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

                                ;;*************************************************************
                                ;;*TEST 66      *DWArF TEST OF OVERFLOW OUT,ST1 IN AND OUT,AND ST2 IN.
                                ;;*
                                ;;*     IN THIS TEST,WE'LL TEST OVERFLOW OUT,ST1 IN,ST1 OUT, AND
                                ;;*     ST2 IN.
                                ;;*
                                ;;*************************************************************
012162  000004                  TST66:  SCOPE
012164  012737  000010  001160          MOV     #10,$TIMES          ;;DO 10 ITERATIONS
012172  012737  012344  0011C_          MOV     #1$,$LPADR
012200  012737  012344  001110          MOV     #1$,$IPERR
012206  005737  001450                  TST     DWARF               ;DWARF TESTS SELECTED?
012212  001505                          BEQ     TST67               ;;
012214  104401  012222                  TYPE    ,65$                ;;TYPE ASCIZ STRING
012220  000426                          BR      64$                 ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <200><7>"DWArF: ALL SWITCHES OFF,S2-2 AND S2-3 ON"
012276                          64$:
012276  104401  012304                  TYPE    ,67$                ;;TYPE ASCIZ STRING
012302  000416                          BR      66$                 ;;GET OVER THE ASCIZ
                                ;;67$:  .ASCIZ  <200>"PANEL: SAME AS LAST TEST"<7>
012340                          66$:
012340  004737  014744                  JSR PC,ANY2
012344  005077  167026          1$:     CLR     @ASR
012350  005737  001450                  TST     DWARF
012354  001424                          BEQ     TST67               ;;
012356  012777  177777  167014          MOV     #-1,@ABR                    ;PRESET FOR OVERFLOW.
012364  012777  000011  167004          MOV     #11,@ASR                    ;SET 1MHZ,GO.
012372  000240                          NOP
012374  000240                          NOP
012376  000240                          NOP
012400  012737  102210  001124          MOV     #102210,$GDDAT              ;EXPECT ST1,ST2 FLAGS SET.
012406  017737  166764  001126          MOV     @ASR,$BDDAT                 ;READ CSR.
012414  023737  001124  001126          CMP     $GDDAT,$BDDAT
012422  001401                          BEQ     TST67               ;;

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

012424  104001                          ERROR 1                             ;ST1 AND/OR ST2 FLAG(S) FAIL TO SET.

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

                                ;;*************************************************************
                                ;;*TEST 67      *IF ENABLED,CHECK THRESHOLD ST1 FROM TESTOR
```

```
                                ;;*****************************************************************
                        TST67:  SCOPE
                                
                                        MOV     #2,$TIMES               ;;DO 2 ITERATIONS
                                        TST     EXS                     ;OPERATING IN TESTOR MODE?
                                        BNE     4$                      ;YES DO THIS TEST.
                                        JMP     ENDP                    ;NO-END PASS
                                4$:
                                        TST     DWARF                   ;DWARF MODE??
                                        BEQ     40$
                                        TST     ASK                     ;QUESTION       ALLREADY BEEN ASKED?
                                        BNE     35$
                                        TYPE    ,65$                    ;;TYPE ASCIZ STRING
                                        BR      64$                     ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <200>#15 VOLT SUPPLY TO DWARF?(Y OR N)#
                                64$:
                                        RDCHR
                                        MOV     (SP)+,ASK
                                        BIC     #240,ASK                ;STRIP PARITY AND LOWER CASE.
                                35$:    CMPB    ASK,#'Y                 ;DID HE ANSWER YES?
                                        BEQ     40$
                                        CMPB    ASK,#'N
                                        BEQ     39$
                                        CLR     ASK
                                        BR      4$
                                
                                39$:    JMP     ENDP
                                40$:
                                        TYPE    ,67$                    ;;TYPE ASCIZ STRING
                                        BR      66$                     ;;GET OVER THE ASCIZ
                                ;;67$:  .ASCIZ  <200>#PANEL: ST1 +ST2 POTS OUT AND CCW#
                                66$:
                                        TYPE    ,69$                    ;;TYPE ASCIZ STRING
                                        BR      68$                     ;;GET OVER THE ASCIZ
                                ;;69$:  .ASCIZ  <200>#DWARF: S2-7 AND S2-8 ON,ALL OTHERS OFF#
                                68$:
                                        CLR     @ASR                    ;CLEAR CSR
                                        JSR PC,ANY2
                                        CLR     @ASR
                                        JSR PC,ANYKEY
                                        MOV     @ASR,$BDDAT             ;READ CSR
                                        MOV     #0,$GDDAT
                                        BIT     #BIT15!BIT10,$BDDAT     ;DID ANY FLAG SET?
                                        BEQ     TST70                   ;;
                                        ERROR 2                         ;ST1 OR ST2 THRESHOLD LEVEL ERROR
                                                                        ;FLAGS SHOULD NOT HAVE SET!
                                
                                ;;*****************************************************************
                                ;*TEST 70        *ST1,ST2 THRESHOLD TEST #2,POTS CW
                                ;;*****************************************************************
                        TST70:  SCOPE
                                        MOV     #1$,$LPERR
                                        MOV     #1$,$LPADR
```

```
                                        TYPE    ,65$            ;;TYPE ASCIZ STRING
                                        BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <200>#PANEL: TURN POTS CW#
                                64$:
                                1$:     CLR     @ASR
                                        JSR PC,ANY2
                                        CLR     @ASR
                                        JSR PC,ANYKEY
                                        MOV     @ASR,$BDDAT
                                        BIT     #BIT15!BIT10,$BDDAT     ;DID ANY FLAG SET?
                                        BEQ     TST71           ;;
                                        ERROR   2                       ;ST1 OR ST2 THRESHOLD ERROR.

                                ;;**************************************************************
                                ;*TEST 71        *ST1,ST2 THRESHOLD TEST #3 MID RANGE
                                ;;**************************************************************
                                TST71:  SCOPE
                                        MOV     #1$,$LPERR
                                        MOV     #1$,$LPADR
                                        TYPE    ,65$            ;;TYPE ASCIZ STRING
                                        BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <200>#PANEL: SET ST1,ST2 POTS MID-RANGE.#
                                64$:
                                1$:     CLR     @ASR
                                        JSR PC,ANY2
                                        CLR     @ASR
                                        JSR PC,ANYKEY
                                        MOV     @ASR,$BDDAT
                                        MOV     #BIT15!BIT10,$GDDAT
                                        BIC     #075777,$BDDAT
                                        CMP     $GDDAT,$BDDAT           ;AT MID RANGE THEY BOTH SHOLD SET.
                                        BEQ     TST72           ;;
                                        ERROR   2                       ;ST1 OR ST2 FAILED TO SET.

                                ;;**************************************************************
                                ;*TEST 72        *TEST CLOCK REPEATABILITY IF ON TESTOR
                                ;;**************************************************************
                                TST72:  SCOPE
                                        MOV     #10,$TIMES      ;;DO 10 ITERATIONS

                                        TST     EXS             ;TESTOR MODE EXABLED??
                                        BNE     10$
                                9$:     JMP     ENDP            ;NO REPORT END PASS.
                                10$:    TST     DWARF
                                        BNE     9$
                                        MOV     #1$,$LPERR
                                        MOV     #1$,$LPADR
                                        TYPE    ,65$            ;;TYPE ASCIZ STRING
                                        BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <200>#PANEL: ST1 POT OUT AND CW#
                                64$:
                                        TYPE    ,67$            ;;TYPE ASCIZ STRING
                                        BR      66$             ;;GET OVER THE ASCIZ
```

```
3047                                        ;;67$:  .ASCIZ  <200>#       ST2 POT  IN AND SLOPE OUT (-)#
3048  013436                                66$:
3049  013436  104401  013444                       TYPE    ,69$            ;;TYPE ASCIZ STRING
3050  013442  000422                             BR      68$             ;;GET OVER THE ASCIZ
3051                                        ;;69$:  .ASCIZ  <200>#DWARF: S2 ALL SWITCHES OFF,S2-6 ON#
3052  013510                                68$:
3053  013510  004737  014744                     JSR PC,ANY2
3054  013514  012777  020016  165654  1$:       MOV     #BIT13!16,@ASR  ;SET 1MHZ,MODE 3,ST2 GO ENABLE. TEST CLOCK
3055  013522  005077  165672                     CLR     @TSCLC          ;CLEAR STATUS REG.
3056  013526  012777  100000  165666             MOV     #100000,@TSCLD  ;PRESET COUNT REG.
3057  013534  012777  000013  165656             MOV     #13,@TSCLC      ;SET 1MHZ,MODE 1,GO
3058  013542  105777  165652          2$:        TSTB    @TSCLC          ;WAIT FOR CLOCK OVERFLOW.
3059  013546  100375                             BPL     2$
3060  013550  042777  100000  165620             BIC     #BIT15,@ASR
3061
3062  013556  042777  000200  165634             BIC     #200,@TSCLC     ;CLEAR OVERFLOW FLAG.
3063  013564  105777  165630          3$:        TSTB    @TSCLC          ;WAIT FOR NEXT OVERFLOW.
3064  013570  100375                             BPL     3$
3065
3066  013572  005077  165600                     CLR     @ASR            ;STOP CLOCK.
3067  013576  005077  165616                     CLR     @TSCLC
3068  013602  017737  165572  001126             MOV     @ABR,$BDDAT     ;READ RESULTS
3069  013610  012737  100000  001124             MOV     #100000,$GDDAT  ;S/B COUNT
3070  013616  013700  001124                     MOV     $GDDAT,R0
3071  013622  163700  001126                     SUB     $BDDAT,R0
3072  013626  100001                             BPL     4$              ;+DIF.
3073  013630  005100                             COM     R0              ;OTHERWIZE MAKE IT
3074  013632  020027  000007          4$:        CMP     R0,#7           ;SHOULD NOT VARY MORE THAN 7 COUNTS.
3075  013636  003401                             BLE     TST73           ;;
3076
3077  013640  104010                             ERROR   10              ;CLOCK REPEATABLITY ERROR.
3078
3079                                      ;;**************************************************************
3080                                      ;*TEST 73            END OF TESTS
3081                                      ;;**************************************************************
3082  013642  000004                      TST73:  SCOPE
3083                                              ;
3084                                              ;NOW WE'LL DETERMINE IF WE ARE ALLOWED TO AUTO-SIZE.
3085                                              ;IF SO, WE'LL FIND OUT IF THERE ARE OTHER CLOCKS OUT THERE
3086                                              ;TO TEST.
3087                                              ;
3088
3089  013644                              ENDP:
3090  013644  000005                             RESET
3091
3092
3093  013646  105737  001215                     TSTB    $ENVM           ;SEE IF APT WILL LET UP AUTO-SIZE.
3094  013652  100547                             BMI     4$              ;NO - EXIT.
3095
3096
3097  013654  023737  001440  001442             CMP     MDEVCT,TSTCNT   ;TESTED MAX. UNITS?
3098  013662  001543                             BEQ     4$              ;YES EXIT.
3099  013664  006337  001432                     ASL     ROTATE          ;POINT NEXT UNIT.
3100  013670  005237  001440                     INC     MDEVCT
```

```
3101
3102  013674  062737  000004  001376          ADD     #4,ASR          ;YES, ADD TO BASE ADDR.
3103  013702  013746  000004                  MOV     ERRVEC,-(6)     ;SAVE CONTENTS OF LOC 4.
3104  013706  012737  014154  000004          MOV     #1$,ERRVEC      ;SET UP IN CASE NO MORE CLOCKS.
3105
3106  013714  005777  165456                  TST     @ASR            ;TIME OUT HERE IF NO MORE CLOCKS.
3107
3108                                                                  ;IF HERE, ANOTHER CLOCK FOUND.
3109  013720  005737  001202                  TST     $PASS           ;IS THIS 1ST PASS?
3110  013724  001003                          BNE     3$              ;NO-GET OUT.
3111  013726  053737  001432  001434          BIS     ROTATE,UTEST    ;YES-RECORD THIS UNIT.
3112  013734                          3$:
3113  013734  104401  013742                  TYPE    ,65$            ;;TYPE ASCIZ STRING
3114  013740  000405                          BR      64$             ;;GET OVER THE ASCIZ
3115                                  ;;65$:  .ASCIZ  <15><12>"UNIT #"
3116  013754                          64$:
3117  013754  013746  001204                  MOV     $DEVCT,-(SP)    ;;SAVE $DEVCT FOR TYPEOUT
3118  013760  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
3119
3120  013762  104401  013770                  TYPE    ,67$            ;;TYPE ASCIZ STRING
3121  013766  000405                          BR      66$             ;;GET OVER THE ASCIZ
3122                                  ;;67$:  .ASCIZ  #  ADDR= #
3123  014002                          66$:
3124  014002  162737  000004  001376          SUB     #4,ASR
3125  014010  013746  001376                  MOV     ASR,-(SP)       ;;SAVE ASR FOR TYPEOUT
3126  014014  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
3127  014016  062737  000004  001376          ADD     #4,ASR
3128  014024  104401  014032                  TYPE    ,69$            ;;TYPE ASCIZ STRING
3129  014030  000406                          BR      68$             ;;GET OVER THE ASCIZ
3130                                  ;;69$:  .ASCIZ  #  VECTOR= #
3131  014046                          68$:
3132  014046  013746  001402                  MOV     VECT1,-(SP)     ;;SAVE VECT1 FOR TYPEOUT
3133  014052  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
3134  014054  104401  014062                  TYPE    ,71$            ;;TYPE ASCIZ STRING
3135  014060  000406                          BR      70$             ;;GET OVER THE ASCIZ
3136                                  ;;71$:  .ASCIZ  " COMPLETED "
3137  014076                          70$:
3138  014076  005237  001204                  INC     $DEVCT
3139  014102  104401  014110                  TYPE    ,73$            ;;TYPE ASCIZ STRING
3140  014106  000410                          BR      72$             ;;GET OVER THE ASCIZ
3141                                  ;;73$:  .ASCIZ  " TESTING UNIT #"
3142  014130                          72$:
3143  014130  013746  001204                  MOV     $DEVcT,-(SP)    ;;SAVE $DEVCT FOR TYPEOUT
3144  014134  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
3145  014136  012637  000004                  MOV     (6)+,ERRVEC     ;RESETORE LOC 4.
3146  014142  062737  000010  001402          ADD     #10,VECT1       ;UPDATE VECTOR ADDR.
3147  014150  000137  002312                  JMP     LOOP            ;TEST NEW UNIT.
3148
3149  014154                          1$:
3150  014154  062706  000004                  ADD     #4,SP                   ;/ADD #4 TO STACK POINTER.
3151  014160  012637  000004                  MOV     (6)+,ERRVEC     ;RESTORE LOC 4
3152  014164  162737  000004  001376          SUB     #4,ASR
3153
3154  014172                          4$:
```

```
3155  014172  104401  014200                    TYPE    ,75$            ;;TYPE ASCIZ STRING
3156  014176  000405                            BR      74$             ;;GET OVER THE ASCIZ
3157                                    ;;75$:  .ASCIZ  <15><12>"UNIT #"
3158  014212                            74$:
3159  014212  013746  001204                    MOV     $DEVCT,-(SP)    ;;SAVE $DEVCT FOR TYPEOUT
3160  014216  104402                            TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
3161  014220  104401  014226                    TYPE    ,77$            ;;TYPE ASCIZ STRING
3162  014224  000405                            BR      76$             ;;GET OVER THE ASCIZ
3163                                    ;;77$:  .ASCIZ  #  ADDR= #
3164  014240                            76$:
3165  014240  013746  001376                    MOV     ASR,-(SP)       ;;SAVE ASR FOR TYPEOUT
3166  014244  104402                            TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
3167  014246  104401  014254                    TYPE    ,79$            ;;TYPE ASCIZ STRING
3168  014252  000406                            BR      78$             ;;GET OVER THE ASCIZ
3169                                    ;;79$:  .ASCIZ  #; VECTOR= #
3170  014270                            78$:
3171  014270  013746  001402                    MOV     VECT1,-(SP)     ;;SAVE VECT1 FOR TYPEOUT
3172  014274  104402                            TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
3173  014276  104401  014304                    TYPE    ,81$            ;;TYPE ASCIZ STRING
3174  014302  000412                            BR      80$             ;;GET OVER THE ASCIZ
3175                                    ;;81$:  .ASCIZ  "; TEST  COMPLETED "
3176  014330                            80$:
3177
3178  014330  013737  001250  001376    2$:     MOV     $BASE,ASR
3179  014336  013737  001244  001402            MOV     $VECT1,VECT1
3180  014344  013737  001204  001446            MOV     $DEVCT,LCNT
3181  014352  005237  001446                    INC     LCNT
3182  014356  012737  000000  001204            MOV     #0,$DEVCT
3183
3184  014364  005037  001440                    CLR     MDEVCT          ;BEGIN TESTING 1ST UNIT.
3185  014370  012737  000001  001432            MOV     #1,ROTATE       ;POINT TO IT.
3186
3187
3188                                    .SBTTL  END OF PASS ROUTINE
3189
3190                                    ;;**************************************************************
3191                                    ;*INCREMENT THE PASS NUMBER ($PASS)
3192                                    ;*IF THERES A MONITOR GO TO IT
3193                                    ;*IF THERE ISN'T JUMP TO LOOP
3194
3195  014376                            $EOP:
3196  014376  000240                            NOP
3197  014400  005037  001102                    CLR     $TSTNM          ;;ZERO THE TEST NUMBER
3198  014404  005037  001160                    CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
3199  014410  005237  001202                    INC     $PASS           ;;INCREMENT THE PASS NUMBER
3200  014414  042737  100000  001202            BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
3201  014422  005327                            DEC     (PC)+           ;;LOOP?
3202  014424  000001                    $EOpCT: .WORD   1
3203  014426  003122                            BGT     $DOAGN          ;;YES
3204  014430  012737                            MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
3205  014432  000001                    $ENDCT: .WORD   1
3206  014434  014424                            $EOPCT
3207  014436  104401  014444                    TYPE    ,65$            ;;TYPE ASCIZ STRING
3208  014442  000406                            BR      64$             ;;GET OVER THE ASCIZ
```

```
3209
3210  014460
3211  014460  013746  001202                ;;65$:  .ASCIZ  <15><12>#ENDPASS  #
3212  014464  104402                        64$:
3213  014466  104401  014474                        MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
3214  014472  000411                                TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
3215                                                TYPE    ,67$            ;;TYPE ASCIZ STRING
3216  014516                                        BR      66$             ;;GET OVER THE ASCIZ
3217  014516  013746  001436                ;;67$:  .ASCIZ  #  TOTAL ERRORS  #
3218  014522  104402                        66$:
3219  014524  104401  014532                        MOV     ERCNT,-(SP)     ;;SAVE ERCNT FOR TYPEOUT
3220  014530  000407                                TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
3221                                                TYPE    ,69$            ;;TYPE ASCIZ STRING
3222  014550                                        BR      68$             ;;GET OVER THE ASCIZ
3223  014550  013746  001446                ;;69$:  .ASCIZ  #;  THERE ARE #
3224  014554  104402                        68$:
3225  014556  104401  014564                        MOV     LCNT,-(SP)      ;;SAVE LCNT FOR TYPEOUT
3226  014562  000411                                TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
3227                                                TYPE    ,71$            ;;TYPE ASCIZ STRING
3228  014606                                        BR      70$             ;;GET OVER THE ASCIZ
3229  014606  104401  014614                ;;71$:  .ASCIZ  #  (OCTAL) UNITS.#
3230  014612  000415                        70$:
3231                                                TYPE    ,73$            ;;TYPE ASCIZ STRING
3232  014646                                        BR      72$             ;;GET OVER THE ASCIZ
3233  014646  013746  001434                ;;73$:  .ASCIZ  <200>#THE GOOD UNITS (L TO R) #
3234  014652  104405                        72$:
3235  014654  013700  000042                        MOV     @#42,R0         ;;SAVE UTEST FOR TYPEOUT
3236  014660  001405                                TYPBN                   ;;GO TYPE--BINARY ASCII
3237  014662  000005                        $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
3238  014664  004710                                BEQ     $DOAgN          ;;BRANCH IF NO MONITOR
3239  014666  000240                                RESET                   ;;CLEAR THE WORLD
3240  014670  000240                        $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
3241  014672  000240                                NOP                     ;;SAVE ROOM
3242  014674                                        NOP                     ;;FOR
3243  014674  000137                                NOP                     ;;ACT11
3244  014676  002312                        $DOAgN:
3245  014700     377     377     000        $RTNAD: JMP     @(PC)+          ;;RETURN
3246             014704                     $ENULL: .WORD   LOOP
3247                                                .BYTE   -1,-1,0         ;;NULL CHARACTER STRING
3248                                                .EVEN
3249                                                ;
3250                                                ;THIS ROUTINE TYPES LAST MESSAGE AND WAITS FOR AN  OPERATOR
3251                                                ;RESPONCE.
3252                                                ;
3253  014704  105777  164236                ANYKEY: TSTB    @$TKB           ;CLEAR TTY READY FLAG.
3254  014710  104401  014716                        TYPE    ,65$            ;;TYPE ASCIZ STRING
3255  014714  000413                                BR      64$             ;;GET OVER THE ASCIZ
3256                                        ;;65$:  .ASCIZ  <200><7>#SWITCH ST1 3 TIMES#
3257  014744                                64$:
3258  014744                                ANY2:
3259  014744  104401  014752                        TYPE    ,65$            ;;TYPE ASCIZ STRING
3260  014750  000417                                BR      64$             ;;GET OVER THE ASCIZ
3261                                        ;;65$:  .ASCIZ  <200><7>#TYPE ANY KEY WHEN DONE...#<7>
3262  015010                                64$:
```

```
                                                                ;WAIT FOR OPERATOR.
015010  105777  164130          1$:     TSTB    @$TKS
015014  100375                          BPL     1$
                                                                ;CLEAR TTY READY FLAG.
015016  105777  164124                  TSTB    @$TKB
                                                                ;;TYPE ASCIZ STRING
015022  104401  015030                  TYPE    ,67$
                                                                ;;GET OVER THE ASCIZ
015026  000401                          BR      66$
                                ;;67$:  .ASCIZ  <200>**
015032                          66$:
015032  000207                          RTS PC

                                        ;
                                .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
                                ;               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

015034  104407          IOTST1: CKSWR                   ;CHECK THE SWR
015036  005077  164334    1$:   CLR     @ASR            ;CLEAR THE CSR
015042  005077  164332          CLR     @ABR            ;CLEAR THE BUFFER REG.
015046  012777  000061  164322  MOV     #61,@ASR        ;RATE ST1, MODE 0, GO.
015054  052777  001000  164314  BIS     #BIT9,@ASR      ;GENERATE A MAINTENANCE ST2.
015062  012777  000005  164306  MOV     #5,@ASR         ;NOW SET TO READ COUNT REG
015070  052777  001000  164300  BIS     #BIT9,@ASR      ;FORCE COUNT -> BUFFER REG.
015076  027727  164276  000001  CMP     @ABR,#1         ;DID COUNT REG ADVANCE ONCE?
015104  001753                  BEQ     IOTST1          ;YES - LOOP.
015106  104000                  ERROR                   ;ST2 OUT TO ST1 IN FAILED.
015110  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.
                                        ;ERRORS ARE REPORTED AS IN TH REGULAR LOGIC TEST AND
                                        ;THEIR PRINTOUT MAY BE INHIBITED.
                                        ;
015112  104407              IOTST2: CKSWR                   ;CHECK THE SWR.
015114  005077  164256              CLR     @ASR            ;CLEAR THE CSR.
015120  012777  177777  164252      MOV     #-1,@ABR        ;PRELOAD PRESET BUFFER.
015126  012777  000063  164242      MOV     #63,@ASR        ;RATE ST1, MODE 1, GO.
015134  052777  000400  164234      BIS     #BIT8,@ASR      ;GENERATE A MAIN. STI.
015142  000240                      NOP
015144  000240                      NOP
015146  005777  164224              TST     @ASR            ;DID OVERFLOW SET ST2 FLAG?
015152  100757                      BMI     IOTST2          ;YES - LOOP
015154  104000                      ERROR                   ;CLK OV OUT TO ST2 IN FAILED.
015156  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

015160  104407                  IOTST3: CKSWR                   ;CHECK THE SWR
015162  012777  000001  164206          MOV     #1,@ASR         ;SET GO BIT.
015170  052777  000400  164200          BIS     #BIT8,@ASR      ;GENERATE A MAIN, ST1.
015176  005777  164174                  TST     @ASR            ;DID ST2 FLAG SET?
015202  100401                          BMI     1$
015204  104000                          ERROR                   ;ST1 OUT TO ST2 IN FAILED

015206  032777  010000  164162  1$:     BIT     #BIT12,@ASR     ;DID "FOR" BIT SET?
015214  001761                          BEQ     IOTST3          ;NO - GOOD!
015216  104000                          ERROR                   ;"FOR" BIT SET ON ONLY 1 ST2.
015220  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

015222  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
015226  116637  000001  015445          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
```

```
3425  015234  112637  015447                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
3426  015240  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
3427  015244  000406                          BR      $TYPON
3428  015246  112737  000001  015445  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
3429  015254  112737  000006  015447          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
3430  015262  112737  000005  015444  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
3431  015270  010346                          MOV     R3,-(SP)        ;;SAVE R3
3432  015272  010446                          MOV     R4,-(SP)        ;;SAVE R4
3433  015274  010546                          MOV     R5,-(SP)        ;;SAVE R5
3434  015276  113704  015447                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
3435  015302  005404                          NEG     R4
3436  015304  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
3437  015310  110437  015446                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
3438  015314  113704  015445                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
3439  015320  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
3440  015324  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
3441  015326  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
3442  015330  000404                          BR      3$              ;;GO DO MSB
3443  015332  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
3444  015334  006105                          ROL     R5
3445  015336  006105                          ROL     R5
3446  015340  010503                          MOV     R5,R3
3447  015342  006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
3448  015344  105337  015446                  DECB    $OMODE          ;;TYPE THIS DIGIT?
3449  015350  100016                          BPL     7$              ;;BR IF NO
3450  015352  042703  177770                  BIC     #177770,R3      ;;GET RID OF JUNK
3451  015356  001002                          BNE     4$              ;;TEST FOR 0
3452  015360  005704                          TST     R4              ;;SUPPRESS THIS 0?
3453  015362  001403                          BEQ     5$              ;;BR IF YES
3454  015364  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
3455  015366  052703  000060                  BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
3456  015372  052703  000040          5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
3457  015376  110337  015442                  MOVB    R3,8$           ;;SAVE FOR TYPING
3458  015402  104401  015442                  TYPE    ,8$             ;;GO TYPE THIS DIGIT
3459  015406  105337  015444          7$:     DECB    $OCNT           ;;COUNT BY 1
3460  015412  003347                          BGT     2$              ;;BR IF MORE TO DO
3461  015414  002402                          BLT     6$              ;;BR IF DONE
3462  015416  005204                          INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
3463  015420  000744                          BR      2$              ;;GO DO THE LAST DIGIT
3464  015422  012605                  6$:     MOV     (SP)+,R5        ;;RESTORE R5
3465  015424  012604                          MOV     (SP)+,R4        ;;RESTORE R4
3466  015426  012603                          MOV     (SP)+,R3        ;;RESTORE R3
3467  015430  016666  000002  000004          MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
3468  015436  012616                          MOV     (SP)+,(SP)
3469  015440  000002                          RTI                     ;;RETURN
3470  015442     000                  8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
3471  015443     000                          .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
3472  015444     000          $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
3473  015445     000          $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
3474  015446  000000          $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
3475                          .SBTTL  BINARY TO ASCII AND TYPE ROUTINE
3476
3477                          ;;**************************************************************
3478                          ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 16-BIT
```

```
                                        ;*BINARY-ASCII NUMBER AND TYPE IT.
                                        ;*CALL:
                                        ;*      MOV     NUMBER,-(SP)    ;;NUMBER TO BE TYPED
                                        ;*      TYPBN                   ;;TYPE IT
                                        
015450  010146                  $TYPBN: MOV     R1,-(SP)        ;;SAVE R1 ON THE STACK
015452  016601  000006                  MOV     6(SP),R1        ;;GET THE INPUT NUMBER
015456  000261                          SEC                     ;;SET "C" SO CAN KEEP TRACK OF THE NUMBER OF BITS
015460  112737  000060  015522  1$:     MOVB    #'0,$BIN        ;;SET CHARACTER TO AN ASCII "0".
015466  006101                          ROL     R1              ;;GET THIS BIT
015470  001406                          BEQ     2$              ;;DONE?
015472  105537  015522                  ADCB    $BIN            ;;NO--SET THE CHARACTER EQUAL TO THIS BIT
015476  104401  015522                  TYPE    ,$BIN           ;;GO TYPE THIS BIT
015502  000241                          CLC                     ;;CLEAR "C" SO CAN KEEP TRACK OF BITS
015504  000765                          BR      1$              ;;GO DO THE NEXT BIT
015506  012601                  2$:     MOV     (SP)+,R1        ;;POP THE STACK INTO R1
015510  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
015516  012616                          MOV     (SP)+,(SP)
015520  000002                          RTI                     ;;RETURN TO USER
015522      000     000         $BIN:   .BYTE   0,0             ;;STORAGE FOR ASCII CHAR. AND TERMINATOR

                                        .SBTTL  ERROR HANDLER ROUTINE

                                        ;;**************************************************************
                                        ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
                                        ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
                                        ;*AND GO TO $ERRTYP ON ERROR
                                        ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
                                        ;*SW15=1         HALT ON ERROR
                                        ;*SW13=1         INHIBIT ERROR TYPEOUTS
                                        ;*SW10=1         BELL ON ERROR
                                        ;*SW09=1         LOOP ON ERROR
                                        ;*CALL
                                        ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER

015524                          $ERROR:
015524  104407                          CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
015526  105237  001103          7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
015532  001775                          BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
015534  013777  001102  163400          MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
015542  032777  002000  163370          BIT     #BIT10,@SWR     ;;BELL ON ERROR?
015550  001402                          BEQ     1$              ;;NO - SKIP
015552  104401  001164                  TYPE    ,$BELL          ;;RING BELL
015556  005237  001112          1$:     INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
015562  011637  001116                  MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
015566  162737  000002  001116          SUB     #2,$ERRPC
015574  117737  163316  001114          MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
015602  032777  020000  163330          BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
015610  001004                          BNE     20$             ;;SKIP TYPEOUTS
015612  004737  015732                  JSR     PC,$ERRTYP      ;;GO TO USER ERROR ROUTINE
015616  104401  001171                  TYPE    ,$CRLF
015622                          20$:
015622  122737  000001  001214          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
```

```
3533  015630  001007                        BNE     2$                      ;;NO, SKIP APT ERROR REPORT
3534  015632  113737  001114  015644        MOVB    $ITEMB,21$              ;;SET ITEM NUMBER AS ERROR NUMBER
3535  015640  004737  017424                JSR     PC,$ATYP                ;;REPORT FATAL ERROR TO APT
3536  015644     000                21$:    .BYTE   0
3537  015645     000                        .BYTE   0
3538  015646  000777                22$:    BR      22$                     ;;APT ERROR LOOP
3539  015650  005777  163264        2$:     TST     @SWR                    ;;HALT ON ERROR
3540  015654  100002                        BPL     3$                      ;;SKIP IF CONTINUE
3541  015656  000000                        HALT                            ;;HALT ON ERROR!
3542  015660  104407                        CKSWR                           ;;TEST FOR CHANGE IN SOFT-SWR
3543  015662  032777  001000  163250  3$:   BIT     #BIT09,@SWR             ;;LOOP ON ERROR SWITCH SET?
3544  015670  001402                        BEQ     4$                      ;;BR IF NO
3545  015672  013716  001110                MOV     $LPERR,(SP)             ;;FUDGE RETURN FOR LOOPING
3546  015676  005737  001162        4$:     TST     $ESCAPE                 ;;CHECK FOR AN ESCAPE ADDRESS
3547  015702  001402                        BEQ     5$                      ;;BR IF NONE
3548  015704  013716  001162                MOV     $ESCAPE,(SP)            ;;FUDGE RETURN ADDRESS FOR ESCAPE
3549  015710                         5$:
3550
3551  015710  005237  001436                INC     ERCNT                   ;/UPDATE ERROR COUNT.
3552  015714  001002                        BNE     10$                     ;/BUT DON'T LET IT OVERFLOW.
3553  015716  005337  001436                DEC     ERCNT                   ;/KEEP AT 177777 IF OVERFLOW.
3554  015722                         10$:
3555  015722  043737  001432  001434        BIC     ROTATE,UTEST            ;/REMOVE UNIT FROM LIST OF GOOD ONES.
3556  015730  000002                        RTI                             ;/EXIT.
3557
3558                                 .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE
3559
3560                                 ;;**********************************************************************
3561                                 ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
3562                                 ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
3563                                 ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
3564
3565  015732                         $ERrTYP:
3566  015732  104401  001171                TYPE    ,$CRLF                  ;;"CARRIAGE RETURN" & "LINE FEED"
3567  015736  010046                        MOV     R0,-(SP)                ;;SAVE R0
3568  015740  005000                        CLR     R0                      ;;PICKUP THE ITEM INDEX
3569  015742  153700  001114                BISB    @#$ITEMB,R0
3570  015746  001004                        BNE     1$                      ;;IF ITEM NUMBER IS ZERO, JUST
3571                                                                         ;;TYPE THE PC OF THE ERROR
3572  015750  013746  001116                MOV     $ERRPC,-(SP)            ;;SAVE $ERRPC FOR TYPEOUT
3573                                                                         ;;ERROR ADDRESS
3574  015754  104402                        TYPOC                           ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
3575  015756  000426                        BR      6$                      ;;GET OUT
3576  015760  005300                1$:     DEC     R0                      ;;ADJUST THE INDEX SO THAT IT WILL
3577  015762  006300                        ASL     R0                      ;;       WORK FOR THE ERROR TABLE
3578  015764  006300                        ASL     R0
3579  015766  006300                        ASL     R0
3580  015770  062700  001256                ADD     #$ERRTB,R0              ;;FORM TABLE POINTER
3581  015774  012037  016004                MOV     (R0)+,2$                ;;PICKUP "ERROR MESSAGE" POINTER
3582  016000  001404                        BEQ     3$                      ;;SKIP TYPEOUT IF NO POINTER
3583  016002  104401                        TYPE                            ;;TYPE THE "ERROR MESSAGE"
3584  016004  000000                2$:     .WORD   0                       ;;"ERROR MESSAGE" POINTER GOES HERE
3585  016006  104401  001171                TYPE    ,$CRLF                  ;;"CARRIAGE RETURN" & "LINE FEED"
3586  016012  012037  016022        3$:     MOV     (R0)+,4$                ;;PICKUP "DATA HEADER" POINTER
```

```
3587  016016  001404                          BEQ     5$              ;;SKIP TYPEOUT IF 0
3588  016020  104401                          TYPE                    ;;TYPE THE "DATA HEADER"
3589  016022  000000                  4$:     .WORD   0               ;;"DATA HEADER" POINTER GOES HERE
3590  016024  104401  001171                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
3591  016030  011000                  5$:     MOV     (R0),R0         ;;PICKUP "DATA TABLE" POINTER
3592  016032  001004                          BNE     7$              ;;GO TYPE THE DATA
3593  016034  012600                  6$:     MOV     (SP)+,R0        ;;RESTORE R0
3594  016036  104401  001171                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
3595  016042  000207                          RTS     PC              ;;RETURN
3596  016044                          7$:
3597  016044  013046                          MOV     @(R0)+,-(SP)    ;;SAVE @(R0)+ FOR TYPEOUT
3598  016046  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
3599  016050  005710                          TST     (R0)            ;;IS THERE ANOTHER NUMBER?
3600  016052  001770                          BEQ     6$              ;;BR IF NO
3601  016054  104401  016062                  TYPE    ,8$             ;;TYPE TWO(2) SPACES
3602  016060  000771                          BR      7$              ;;LOOP
3603  016062  020040      000         8$:     .ASCIZ  /  /            ;;TWO(2) SPACES
3604          016066                          .EVEN
3605                                  .SBTTL  SCOPE HANDLER ROUTINE
3606
3607                                  ;;**********************************************************************
3608                                  ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
3609                                  ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
3610                                  ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
3611                                  ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
3612                                  ;*SW14=1         LOOP ON TEST
3613                                  ;*SW11=1         INHIBIT ITERATIONS
3614                                  ;*SW09=1         LOOP ON ERROR
3615                                  ;*SW08=1         LOOP ON TEST IN SWR<7:0>
3616                                  ;*CALL
3617                                  ;*      SCOPE           ;;SCOPE=IOT
3618
3619  016066                          $SCOPE:
3620  016066  104407                          CKSWR                   ;;TEST  FOR CHANGE IN SOFT-SWR
3621  016070  104407                          CKSWR
3622  016072  032777  040000  163040  1$:     BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
3623  016100  001114                          BNE     $OVER           ;;YES IF SW14=1
3624                                  ;*****START OF CODE FOR THE XOR TESTER*****
3625  016102  000416                  $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
3626                                                                  ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
3627  016104  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
3628  016110  012737  016130  000004          MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
3629  016116  005737  177060                  TST     @#177060        ;;TIME OUT ON XOR?
3630  016122  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
3631  016126  000463                          BR      $SVLAD          ;;GO TO THE NEXT TEST
3632  016130  022626                  5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
3633  016132  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
3634  016136  000423                          BR      7$              ;;LOOP ON THE PRESENT TEST
3635  016140                          6$:;*****END OF CODE FOR THE XOR TESTER*****
3636  016140  032777  000400  162772          BIT     #BIT08,@SWR     ;;LOOP ON SPEC. TEST?
3637  016146  001404                          BEQ     2$              ;;BR IF NO
3638  016150  127737  162764  001102          CMPB    @SWR,$TSTNM     ;;ON THE RIGHT TEST?    SWR<7:0>
3639  016156  001465                          BEQ     $OVER           ;;BR IF YES
3640  016160  105737  001103          2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
```

```
3641  016164  001421                        BEQ     3$                ;;BR IF NO
3642  016166  123737  001115  001103        CMPB    $ERMAX,$ERFLG     ;;MAX. ERRORS FOR THIS TEST OCCURRED?
3643  016174  101015                        BHI     3$                ;;BR IF NO
3644  016176  032777  001000  162734        BIT     #BIT09,@SWR       ;;LOOP ON ERROR?
3645  016204  001404                        BEQ     4$                ;;BR IF NO
3646  016206  013737  001110  001106  7$:   MOV     $LPERR,$LPADR     ;;SET LOOP ADDRESS TO LAST SCOPE
3647  016214  000446                        BR      $OVER
3648  016216  105037  001103          4$:   CLRB    $ERFLG            ;;ZERO THE ERROR FLAG
3649  016222  005037  001160                CLR     $TIMES            ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
3650  016226  000415                        BR      1$                ;;ESCAPE TO THE NEXT TEST
3651  016230  032777  004000  162702  3$:   BIT     #BIT11,@SWR       ;;INHIBIT ITERATIONS?
3652  016236  001011                        BNE     1$                ;;BR IF YES
3653  016240  005737  001202                TST     $PASS             ;;IF FIRST PASS OF PROGRAM
3654  016244  001406                        BEQ     1$                ;;      INHIBIT ITERATIONS
3655  016246  005237  001104                INC     $ICNT             ;;INCREMENT ITERATION COUNT
3656  016252  023737  001160  001104        CMP     $TIMES,$ICNT      ;;CHECK THE NUMBER OF ITERATIONS MADE
3657  016260  002024                        BGE     $OVER             ;;BR IF MORE ITERATION REQUIRED
3658  016262  012737  000001  001104  1$:   MOV     #1,$ICNT          ;;REINITIALIZE THE ITERATION COUNTER
3659  016270  013737  016346  001160        MOV     $MXCNT,$TIMES     ;;SET NUMBER OF ITERATIONS TO DO
3660  016276  105237  001102  $SVLAD: INCB  $TSTNM                    ;;COUNT TEST NUMBERS
3661  016302  113737  001102  001200        MOVB    $TSTNM,$TESTN     ;;SET TEST NUMBER IN APT MAILBOX
3662  016310  011637  001106                MOV     (SP),$LPADR       ;;SAVE SCOPE LOOP ADDRESS
3663  016314  011637  001110                MOV     (SP),$LPERR       ;;SAVE ERROR LOOP ADDRESS
3664  016320  005037  001162                CLR     $ESCAPE           ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
3665  016324  112737  000001  001115        MOVB    #1,$ERMAX         ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
3666  016332  013777  001102  162602  $OVER:  MOV     $TSTNM,@DISPLAY   ;;DISPLAY TEST NUMBER
3667  016340  013716  001106                MOV     $LPADR,(SP)       ;;FUDGE RETURN ADDRESS
3668  016344  000002                        RTI                       ;;FIXES PS
3669  016346  003720                $MXCNT: 2000.                     ;;MAX. NUMBER OF ITERATIONS
3670                                .SBTTL  TTY INPUT ROUTINE
3671
3672                                ;;**********************************************************************
3673                                .ENABL  LSB
3674
3675                                ;;**********************************************************************
3676                                ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
3677                                ;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
3678                                ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
3679                                ;*WHEN OPERATING IN TTY FLAG MODE.
3680  016350  022737  000176  001140  $CKSWR: CMP     #SWREG,SWR        ;;IS THE SOFT-SWR SELECTED?
3681  016356  001074                        BNE     15$               ;;BRANCH IF NO
3682  016360  105777  162560                TSTB    @$TKS             ;;CHAR THERE?
3683  016364  100071                        BPL     15$               ;;IF NO, DON'T WAIT AROUND
3684  016366  117746  162554                MOVB    @$TKB,-(SP)       ;;SAVE THE CHAR
3685  016372  042716  177600                BIC     #^C177,(SP)       ;;STRIP-OFF THE ASCII
3686  016376  022726  000007                CMP     #7,(SP)+          ;;IS IT A CONTROL G?
3687  016402  001062                        BNE     15$               ;;NO, RETURN TO USER
3688  016404  123727  001134  000001        CMPB    $AUTOB,#1         ;;ARE WE RUNNING IN AUTO-MODE?
3689  016412  001456                        BEQ     15$               ;;BRANCH IF YES
3690
3691  016414  104401  017075                TYPE    ,$CNTLG           ;;ECHO THE CONTROL-G (^G)
3692  016420  104401  017102  $GTSWR: TYPE  ,$MSWR                    ;;TYPE CURRENT CONTENTS
3693  016424  013746  000176                MOV     $SWREG,-(SP)      ;;SAVE SWREG FOR TYPEOUT
3694  016430  104402                        TYPOC                     ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
```

```
3695  016432  104401  017113          TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
3696  016436  005046                  CLR     -(SP)           ;;CLEAR COUNTER
3697  016440  005046                  CLR     -(SP)           ;;THE NEW SWR
3698  016442  105777  162476  7$:     TSTB    @$TKS           ;;CHAR THERE?
3699  016446  100375                  BPL     7$              ;;IF NOT TRY AGAIN
3700
3701  016450  117746  162472          MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
3702  016454  042716  177600          BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
3703
3704
3705
3706  016460  021627  000025  9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
3707  016464  001005                  BNE     10$             ;;BRANCH IF NOT
3708  016466  104401  017070          TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (^U)
3709  016472  062706  000006  20$:     ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
3710  016476  000757                  BR      19$             ;;LET'S TRY IT AGAIN
3711
3712
3713  016500  021627  000015  10$:     CMP     (SP),#15        ;;IS IT A <CR>?
3714  016504  001022                  BNE     16$             ;;BRANCH IF NO
3715  016506  005766  000004          TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
3716  016512  001403                  BEQ     11$             ;;BRANCH IF YES
3717  016514  016677  000002  162416  MOV     2(SP),@SWR      ;;SAVE NEW SWR
3718  016522  062706  000006  11$:     ADD     #6,SP           ;;CLEAR UP STACK
3719  016526  104401  001171  14$:     TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
3720  016532  123727  001135  000001  CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
3721  016540  001003                  BNE     15$             ;;BRANCH IF NOT
3722  016542  012777  000100  162374  MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
3723  016550  000002          15$:     RTI                     ;;RETURN
3724  016552  004737  017336  16$:     JSR     PC,$TYPEC       ;;ECHO CHAR
3725  016556  021627  000060          CMP     (SP),#60        ;;CHAR < 0?
3726  016562  002420                  BLT     18$             ;;BRANCH IF YES
3727  016564  021627  000067          CMP     (SP),#67        ;;CHAR > 7?
3728  016570  003015                  BGT     18$             ;;BRANCH IF YES
3729  016572  042726  000060          BIC     #60,(SP)+       ;;STRIP-OFF ASCII
3730  016576  005766  000002          TST     2(SP)           ;;IS THIS THE FIRST CHAR
3731  016602  001403                  BEQ     17$             ;;BRANCH IF YES
3732  016604  006316                  ASL     (SP)            ;;NO, SHIFT PRESENT
3733  016606  006316                  ASL     (SP)            ;;    CHAR OVER TO MAKE
3734  016610  006316                  ASL     (SP)            ;;    ROOM FOR NEW ONE.
3735  016612  005266  000002  17$:     INC     2(SP)           ;;KEEP COUNT OF CHAR
3736  016616  056616  177776          BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
3737  016622  000707                  BR      7$              ;;GET THE NEXT ONE
3738  016624  104401  001170  18$:     TYPE    ,$QUES          ;;TYPE ?<CR><LF>
3739  016630  000720                  BR      20$             ;;SIMULATE CONTROL-U
3740                        .DSABL  LSB
3741
3742
3743                        ;;*****************************************************************
3744                        ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
3745                        ;*CALL:
3746                        ;*      RDCHR                   ;;INPUT A SINGLE CHARACTER FROM THE TTY
3747                        ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
3748                        ;*                              ;;WITH PARITY BIT STRIPPED OFF
```

```
                                        ;
                                $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
                                        MOV     4(SP),2(SP)     ;;SAVE THE PS
                                1$:     TSTB    @$TKS           ;;WAIT FOR
                                        BPL     1$              ;;A CHARACTER
                                        MOVB    @$TKB,4(SP)     ;;READ THE TTY
                                        BIC     #^C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
                                        CMP     4(SP),#23       ;;IS IT A CONTROL-S?
                                        BNE     3$              ;;BRANCH IF NO
                                2$:     TSTB    @$TKS           ;;WAIT FOR A CHARACTER
                                        BPL     2$              ;;LOOP UNTIL ITS THERE
                                        MOVB    @$TKB,-(SP)     ;;GET CHARACTER
                                        BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
                                        CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
                                        BNE     2$              ;;IF NOT DISCARD IT
                                        BR      1$              ;;YES, RESUME
                                3$:     CMP     4(SP),#140      ;;IS IT UPPER CASE?
                                        BLT     4$              ;;BRANCH IF YES
                                        CMP     4(SP),#175      ;;IS IT A SPECIAL CHAR?
                                        BGT     4$              ;;BRANCH IF YES
                                        BIC     #40,4(SP)       ;;MAKE IT UPPER CASE
                                4$:     RTI                     ;;GO BACK TO USER
                                ;;**************************************************************
                                ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
                                ;*CALL:
                                ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
                                ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
                                ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
                                $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
                                1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
                                2$:     CMP     #$TTYIN+8.,R3   ;;BUFFER FULL?
                                        BLOS    4$              ;;BR IF YES
                                        RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
                                        MOVB    (SP)+,(R3)      ;;GET CHARACTER
                                10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
                                        BNE     3$              ;;SKIP IF NOT
                                4$:     TYPE    ,$QUES          ;;TYPE A '?'
                                        BR      1$              ;;CLEAR THE BUFFER AND LOOP
                                3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
                                        TYPE    ,9$
                                        CMPB    #15,(R3)+       ;;CHECK FOR RETURN
                                        BNE     2$              ;;LOOP IF NOT RETURN
                                        CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
                                        TYPE    ,$LF            ;;TYPE A LINE FEED
                                        MOV     (SP)+,R3        ;;RESTORE R3
                                        MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
                                        MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
                                        MOV     #$TTYIN,4(SP)
                                        RTI                     ;;RETURN
                                9$:     .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
                                        .BYTE   0               ;;TERMINATOR
                                $TTYIN: .BLKB   8.              ;;RESERVE 8 BYTES FOR TTY INPUT
```

```
 3803  017070  052536  005015      000  $CNTLU: .ASCIZ  /↑U/<15><12>     ;;CONTROL "U"
 3804  017075     136  006507   000012  $CNTLG: .ASCIZ  /↑G/<15><12>     ;;CONTROL "G"
 3805  017102  005015  053523   020122  $MSWR:  .ASCIZ  <15><12>/SWR = /
 3806  017110  020075     000
 3807  017113     040  047040   053505  $MNEW:  .ASCIZ  /  NEW = /
 3808  017120  036440  000040
 3809                                   .SBTTL  TYPE ROUTINE
 3810
 3811                                   ;;*****************************************************************
 3812                                   ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERminATE WITH A 0 BYTE.
 3813                                   ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
 3814                                   ;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
 3815                                   ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
 3816                                   ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
 3817                                   ;*
 3818                                   ;*CALL:
 3819                                   ;*1) USING A TRAP INSTRUCTION
 3820                                   ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
 3821                                   ;*OR
 3822                                   ;*      TYPE
 3823                                   ;*      MESADR
 3824                                   ;*
 3825
 3826  017124  105737  001157  $TYPE:   TSTB    $TPFLG          ;;IS THERE A TERMINAL?
 3827  017130  100002                   BPL     1$              ;;BR IF YES
 3828  017132  000000                   HALT                    ;;HALT HERE IF NO TERMINAL
 3829  017134  000430                   BR      3$              ;;LEAVE
 3830  017136  010046          1$:      MOV     R0,-(SP)        ;;SAVE R0
 3831  017140  017600  000002          MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
 3832  017144  122737  000001  001214  CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
 3833  017152  001011                   BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
 3834  017154  132737  000100  001215  BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
 3835  017162  001405                   BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
 3836  017164  010037  017174          MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
 3837  017170  004737  017414          JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
 3838  017174  000000          61$:     .WORD   0               ;;MESSAGE ADDRESS
 3839  017176  132737  000040  001215  62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
 3840  017204  001003                   BNE     60$             ;;YES,SKIP TYPE OUT
 3841  017206  112046          2$:      MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
 3842  017210  001005                   BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
 3843  017212  005726                   TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
 3844  017214  012600          60$:     MOV     (SP)+,R0        ;;RESTORE R0
 3845  017216  062716  000002  3$:      ADD     #2,(SP)         ;;ADJUST RETURN PC
 3846  017222  000002                   RTI                     ;;RETURN
 3847  017224  122716  000011  4$:      CMPB    #HT,(SP)        ;;BRANCH IF <HT>
 3848  017230  001430                   BEQ     8$
 3849  017232  122716  000200          CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
 3850  017236  001006                   BNE     5$
 3851  017240  005726                   TST     (SP)+           ;;POP  <CR><LF> EQUIV
 3852  017242  104401                   TYPE                    ;;TYPE A CR AND LF
 3853  017244  001171                   $CRLF
 3854  017246  105037  017402          CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
 3855  017252  000755                   BR      2$              ;;GET NEXT CHARACTER
 3856  017254  004737  017336  5$:      JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
```

```
3857  017260  123726  001156          6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
3858  017264  001350                          BNE     2$              ;;IF NO GO GET NEXT CHAR.
3859  017266  013746  001154                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
3860                                                                  ;;AND THE NULL CHAR.
3861  017272  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
3862  017276  002770                          BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
3863  017300  004737  017336                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
3864  017304  105337  017402                  DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
3865  017310  000770                          BR      7$              ;;LOOP
3866
3867                                          ;HORIZONTAL TAB PROCESSOR
3868
3869  017312  112716  000040          8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
3870  017316  004737  017336          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
3871  017322  132737  000007  017402          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
3872  017330  001372                          BNE     9$              ;;TAB STOP
3873  017332  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
3874  017334  000724                          BR      2$              ;;GET NEXT CHARACTER
3875  017336  105777  161606          $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
3876  017342  100375                          BPL     $TYPEC
3877  017344  116677  000002  161600          MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
3878  017352  122766  000015  000002          CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
3879  017360  001003                          BNE     1$              ;;BRANCH IF NO
3880  017362  105037  017402                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
3881  017366  000406                          BR      $TYPEX          ;;EXIT
3882  017370  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
3883  017376  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
3884  017400  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
3885  017402  000000          $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
3886  017404  000207          $TYPEX: RTS     PC
3887
3888                                          .SBTTL  APT COMMUNICATIONS ROUTINE
3889
3890                                          ;;**************************************************************
3891  017406  112737  000001  017652  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
3892  017414  112737  000001  017650  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
3893  017422  000403                          BR      $ATYC
3894  017424  112737  000001  017652  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
3895  017432                          $ATYC:
3896  017432  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
3897  017434  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
3898  017436  105737  017650                  TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
3899  017442  001450                          BEQ     5$              ;;IF NOT:  BR
3900  017444  122737  000001  001214          CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
3901  017452  001031                          BNE     3$              ;;IF NOT:  BR
3902  017454  132737  000100  001215          BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
3903  017462  001425                          BEQ     3$              ;;IF NOT:  BR
3904  017464  017600  000004                  MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
3905  017470  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
3906  017476  005737  001174          1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
3907  017502  001375                          BNE     1$              ;;IF NOT:  WAIT
3908  017504  010037  001210                  MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
3909  017510  105720                  2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
3910  017512  001376                          BNE     2$
```

```
3911   017514  163700  001210                  SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
3912   017520  006200                          ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
3913   017522  010037  001212                  MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
3914   017526  012737  000004  001174          MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
3915   017534  000413                          BR      5$
3916   017536  017637  000004  017562  3$:     MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
3917   017544  062766  000002  000004          ADD     #2,4(SP)        ;;BUMP RETURN ADDRESS
3918   017552  013746  177776                  MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
3919   017556  004737  017124                  JSR     PC,$TYPE        ;;CALL TYPE MACRO
3920   017562  000000                  4$:     .WORD   0
3921   017564                          5$:
3922   017564  105737  017652          10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
3923   017570  001416                          BEQ     12$             ;;IF NOT:  BR
3924   017572  005737  001214                  TST     $ENV            ;;RUNNING UNDER APT?
3925   017576  001413                          BEQ     12$             ;;IF NOT:  BR
3926   017600  005737  001174          11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
3927   017604  001375                          BNE     11$             ;;IF NOT:  WAIT
3928   017606  017637  000004  001176          MOV     @4(SP),$FATAL   ;;GET ERROR #
3929   017614  062766  000002  000004          ADD     #2,4(SP)        ;;BUMP RETURN ADDR.
3930   017622  005237  001174                  INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
3931   017626  105037  017652          12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
3932   017632  105037  017651                  CLRB    $LFLG           ;;CLEAR LOG FLAG
3933   017636  105037  017650                  CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
3934   017642  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
3935   017644  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
3936   017646  000207                          RTS     PC              ;;RETURN
3937   017650     000          $MFLG:  .BYTE   0               ;;MESSG. FLAG
3938   017651     000          $LFLG:  .BYTE   0               ;;LOG FLAG
3939   017652     000          $FFLG:  .BYTE   0               ;;FATAL FLAG
3940           017654                  .EVEN
3941           000200          APTSIZE=200
3942           000001          APTENV=001
3943           000100          APTSPOOL=100
3944           000040          APTCSUP=040
3945                           .SBTTL  POWER DOWN AND UP ROUTINES
3946
3947                           ;;*****************************************************************
3948                           ;POWER DOWN ROUTINE
3949   017654  012737  020014  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
3950   017662  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
3951   017670  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
3952   017672  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
3953   017674  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
3954   017676  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
3955   017700  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
3956   017702  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
3957   017704  017746  161230                  MOV     @$WR,-(SP)      ;;PUSH @$WR ON STACK
3958   017710  010637  020020                  MOV     SP,$SAVR6       ;;SAVE SP
3959   017714  012737  017726  000024          MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
3960   017722  000000                          HALT
3961   017724  000776                          BR      .-2             ;;HANG UP
3962
3963                           ;;*****************************************************************
3964                           ;POWER UP ROUTINE
```

```
017726  012737  020014  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC        ;;SET FOR FAST DOWN
017734  013706  020020                  MOV     $SAVR6,SP               ;;GET SP
017740  005037  020020                  CLR     $SAVR6                  ;;WAIT LOOP FOR THE TTY
017744  005237  020020          1$:     INC     $SAVR6                  ;;WAIT FOR THE INC
017750  001375                          BNE     1$                      ;;OF WORD
017752  012677  161162                  MOV     (SP)+,@SWR              ;;POP STACK INTO @SWR
017756  012605                          MOV     (SP)+,R5                ;;POP STACK INTO R5
017760  012604                          MOV     (SP)+,R4                ;;POP STACK INTO R4
017762  012603                          MOV     (SP)+,R3                ;;POP STACK INTO R3
017764  012602                          MOV     (SP)+,R2                ;;POP STACK INTO R2
017766  012601                          MOV     (SP)+,R1                ;;POP STACK INTO R1
017770  012600                          MOV     (SP)+,R0                ;;POP STACK INTO R0
017772  012737  017654  000024          MOV     #$PWRDN,@#PWRVEC        ;;SET UP THE POWER DOWN VECTOR
020000  012737  000340  000026          MOV     #340,@#PWRVEC+2         ;;PRIO:7
020006  104401                          TYPE                            ;REPORT THE POWER FAILURE
020010  020022                  $PWRMG: .WORD   $POWER                  ;;POWER FAIL MESSAGE POIN.ER
020012  000002                          RTI
020014  000000                  $ILLUP: HALT                            ;;THE POWER UP SEQUENCE WAS STARTED
020016  000776                          BR      .-2                     ;; BEFORE THE POWER DOWN WAS COMPLETE
020020  000000                  $SAVR6: 0                               ;;PUT THE SP HERE
020022  005015  047520  042527  $POWER: .ASCIZ  <15><12>"POWER"
020030  000122
                                        .EVEN
                                        ;*
                                        ;*THIS ROUTINE WILL PROTECT THE PROGRAM
                                        ;*FROM INTERUPTS (BAD ONES).
                                        ;*
                                        ;*THE TRAP CATCHER IS SET UP FOR
                                        ;*      .WORD   .+2
                                        ;*      JSR PC,R0
                                        ;*
                                        ;*ILLEGAL INTERRUPTS OR INTERRUPTS TO THE WRONG VECTOR
                                        ;*GOTO THE VECTOR AND PCITK UP THE ".+2" AS AN ADDRESS

                                        ;*AND "4700" AS NEW STATUS.
                                        ;*THE .+2 AS A PC WILL CAUSE EXECUTION OF THE "JSR PC,R0" (AN ILLEGAL INSTR.).
                                        ;*AND TRAP TO LOCATION "4". IN LOCATION 4 WE HAVE A
                                        ;*POINTER HERE. IF THIS CONDITION CAUSES A TRAP TO LOC. 4.
                                        ;*WE WILL REPORT IT IN THE SAME MANNER THAT WER WOULD
                                        ;*REPORT ANY OTHER ERROR.
                                        ;*IF A BUSS ERROR TRAP DID OCCUT AND CAUSE A TRAP TO 4.
                                        ;*WE WILL HALT.

020032  011637  020206          IOTRD:  MOV     (6),TRTO                ;GET WHERE WE CAME TO.
020036  162737  000004  020206          SUB     #4,TRTO                 ;FORM READ ADDR.

020044  023727  020206  001000          CMP     TRTO,#1000              ;DID TRAP FROM LESS THAN ADDR. 1000?
020052  003402                          BLE     2$                      ;NO-CONTINUE.

020054  000000                  1$:     HALT                            ;A BUSS ERROR TIME OUT TRAP BROUGHT US HERE.
                                                                        ;ADDRESS CONTAINED IN TRTO.

020056  000776                          BR      1$                      ;DON'T ALLOW CONTINUE.
```

```
020060  016637  000004  020210  2$:     MOV     4(6),TRFRO              ;GET TRAPPED FROM ADDR.
020066  062706  000004                  ADD     #4,SP                           ;/ADD #4 TO STACK POINTER.

020072  122737  000044  001102          CMPB    #44,$TSTNM              ;LESS THAN INTERRUPT TESTS?
020100  003402                          BLE     3$                      ;NO MUST BE WRONG VECTOR.

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

020102  104004                          ERROR   4                       ;ERROR! ILLEGAL INTERRUPT OR
                                                                        ;INTERRUPT TO WRONG VECTOR.
                                                                        ;IF TEST NO. IS LESS THAN 10,ITS
                                                                        ;LIKELY(BUT NO EXCLUSIVELY)TO BE A
                                                                        ;DEVICE OTHER THAN THE DEVICE UNDER TEST.
                                                                        ;IF THE INTERRUPT OCCURED
                                                                        ;DURING AN INTERRUPT TEST. I'D
                                                                        ;SUSPECT A PROBLEM WITH THE DEVICE UNDER TEST.
                                                                        ;IF THE ADDRESS THE INTERRUPT
                                                                        ;VECTORED TO IS WITHIN THE RANGE OF
                                                                        ;VECTORS ASSIGNED TO THE DEVICE,
                                                                        ;THEN I'D SUSPECT THE DEVICE
                                                                        ;INTERRUPTD ILLEGALLY.
                                                                        ;IF THE ADDRESS THE INTERRUPT
                                                                        ;VECTORED TO IS OUTSIDE OF THE
                                                                        ;RANGE ASSIGNED TO THE DEVICE
                                                                        ;I'D SUSPECT THAT THE
                                                                        ;DEVICE PUT THE WRONG INTERRUPT
                                                                        ;VICTOR ON THE BUS DURING THE INTERRUPT
                                                                        ;PROCESS.
                                                                        ;       NOTE:
                                                                        ;FOR THIS ERROR - DON'T USE
                                                                        ; "LOOP ON ERROR" OPTION.
                                                                        ;ALSO EXPECT THAT THE INTERRUPT TEST TO
                                                                        ;WILL REPOT THAT THE DEVICE DIDN'T
                                                                        ;INTERRUPT.
                                                                        ;FOLLOW THE RECOMMENDED PROCEEDURE
                                                                        ; IN THE DOCUMENT (ON THIS DIAGNOSTIC)
                                                                        ;FOR LOOPING ON TEST.

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

020104  000002                          RTI
020106  013777  001404  161266  3$:     MOV     VECTP,@VECT1
020114  013777  001410  161264          MOV     VECT2P,@VECT2
020122  013737  020206  001402          MOV     TRTO,VECT1
020130  042737  000007  001402          BIC     #7,VECT1
020136  013737  001402  001404          MOV     VECT1,VECTP
020144  062737  000002  001404          ADD     #2,VECTP
020152  013737  001402  001406          MOV     VECT1,VECT2
020160  062737  000004  001406          ADD     #4,VECT2
020166  013737  001406  001410          MOV     VECT2,VECT2P
020174  062737  000002  001410          ADD     #2,VECT2P
020202  000177  160700                  JMP     @$LPADR                 ;START TEST OVER AGAIN.
020206  000000                  TRTO:   .WORD   0                       ;CONTAINS ADDR. WE TRAPPED OR INTERRUPTED TO.
```

```
                                TRFRQ:  .WORD   0               ;CONTAINS ADDR. WE TRAPPED OR INTR. FROM.
                                .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.

020212  010046                  $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
020214  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
020220  005740                          TST     -(R0)           ;;BACKUP BY 2
020222  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
020224  006300                          ASL     R0              ;;POSITION FOR INDEXING
020226  016000  020246                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
020232  000200                          RTS     R0              ;;GO TO ROUTINE

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

020234  011646                  $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
020236  016666  000004  000002          MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
020244  000002                          RTI                     ;;RESTORE THE PSW

                                .SBTTL  TRAP TABLE

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

                                ;       ROUTINE
                                ;       -------
020246  020234                  $TRPAD: .WORD   $TRAP2
020250  017124                          $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
020252  015246                          $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
020254  015222                          $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
020256  015262                          $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
020260  015450                          $TYPBN  ;;CALL=TYPBN    TRAP+5(104405)  TYPE BINARY (ASCII) NUMBER

020262  016420                          $GTSWR  ;;CALL=GTSWR    TRAP+6(104406)  GET SOFT-SWR SETTING

020264  016350                          $CKSWR  ;;CALL=CKSWR    TRAP+7(104407)  TEST FOR CHANGE IN SOFT-SWR
020266  016632                          $RDCHR  ;;CALL=RDCHR    TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
020270  016752                          $RDLIN  ;;CALL=RDLIN    TRAP+11(104411) TTY TYPEIN STRING ROUTINE
020272  005015  046103  041517  EM1:    .ASCIZ  <15><12>/CLOCK SR FUNCTION ERROR/
020300  020113  051123  043040
020306  047125  052103  047511
020314  020116  051105  047522
020322  000122
020324  005015  046103  041517  EM2:    .ASCIZ  <15><12>/CLOCK SR DATA ERROR/
020332  020113  051123  042040
020340  052101  020101  051105
020346  047522  000122
020352  005015  046103  041517  EM3:    .ASCIZ  <15><12>/CLOCK BR DATA ERROR/
020360  020113  051102  042040
```

```
4127  020366  052101  020101  052105
4128  020374  047522  000122
4129  020400  044600  052116  051105  EM4:    .ASCIZ  <200>/INTERRUPT ERROR/
4130  020406  052522  052120  042440
4131  020414  051122  051117     000
4132  020421     015  041412  052517  EM5:    .ASCIZ  <15><12>/COUNT REG. ERROR/
4133  020426  052116  051040  043505
4134  020434  020056  051105  047522
4135  020442  000122
4136  020444  005015  047503  047125  EM11:   .ASCIZ  <15><12>*COUNT ERROR *
4137  020452  020124  051105  047522
4138  020460  020122     000
4139  020463     015  041412  052517  EM12:   .ASCIZ  <15><12>*COUNT FUNCTION ERROR*
4140  020470  052116  043040  047125
4141  020476  052103  047511  020116
4142  020504  051105  047522  000122
4143  020512  005015  046103  041517  EM16:   .ASCIZ  <15><12>*CLOCK INTERRUPT ERROR *
4144  020520  020113  047111  042524
4145  020526  051122  050125  020124
4146  020534  051105  047522  020122
4147  020542     000
4148  020543     015  051012  050105  EM20:   .ASCIZ  <15><12>*REPEATABILITY ERROR *
4149  020550  040505  040524  044502
4150  020556  044514  054524  042440
4151  020564  051122  051117  000040
4152  020572  005015  042101  051104  EM26:   .ASCIZ  <15><12>*ADDRESSING ERROR*
4153  020600  051505  044523  043516
4154  020606  042440  051122  051117
4155  020614     000
4156
4157  020615     015  042412  051122  DH1:    .ASCIZ  <15><12>*ERRPC  ASR     WAS     S/B*
4158  020622  041520  040411  051123
4159  020630  053411  051501  051411
4160  020636  041057     000
4161  020641     015  042412  051122  DH3:    .ASCIZ  <15><12>*ERRPC  ABR     WAS     S/B*
4162  020646  041520  040411  051102
4163  020654  053411  051501  051411
4164  020662  041057     000
4165  020665     200  051105  050122  DH4A:   .ASCIZ  <200>*ERRPC   TO      FROM ADDR.*
4166  020672  020103  020040  047524
4167  020700  020040  020040  020040
4168  020706  051106  046517  040440
4169  020714  042104  027122     000
4170  020721     015  042412  051122  DH12:   .ASCIZ  <15><12>*ERRPC  ASR     *
4171  020726  041520  040411  051123
4172  020734  000011
4173  020736  005015  051105  050122  DH20:   .ASCIZ  <15><12>*ERRPC  ASR     2NDCNT  1STNCT  *
4174  020744  004503  051501  004522
4175  020752  047062  041504  052116
4176  020760  030411  052123  041516
4177  020766  004524     000
4178  020771     015  042412  051122  DH26:   .ASCIZ  <15><12>*ERRPC  CLOCK ADDR.*
4179  020776  041520  041411  047514
4180  021004  045503  040440  042104
```

```
4181   021012  027122  000
4182   021015  200     042115  030461  WRNM1:  .ASCII  <200>"MD11-DVMNC-A"
4183   021022  042055  046526  041516
4184   021030  040455
4185   021032  050200  042514  051501          .ASCII  <200>"PLEASE PULL OUT ST SWITCHES AND THEN TURn"
4186   021040  020105  052520  046114
4187   021046  047440  052125  051440
4188   021054  020124  053523  052111
4189   021062  044103  051505  040440
4190   021070  042116  052040  042510
4191   021076  020116  052524  047122
4192   021104  052200  042510  020115          .ASCIZ  <200>"THEM COMPLETELY CW OR CCW"<200><7>
4193   021112  047503  050115  042514
4194   021120  042524  054514  041440
4195   021126  020127  051117  041440
4196   021134  053503  003600  000
4197
4198           021142                          .EVEN
4199
4200   021142  001116  001376  001126  DT1:    .WORD   $ERRPC,ASR,$BDDAT,$GDDAT,0
4201   021150  001124  000000
4202   021154  001116  001400  001126  DT3:    .WORD   $ERRPC,ABR,$BDDAT,$GDDAT,0
4203   021162  001124  000000
4204   021166  001116  020206  020210  DT4:    .WORD   $ERrPC,TRT0,TRFR0,0
4205   021174  000000
4206   021176  001116  001376  000000  DT12:   .WORD   $ERrPC,ASR,0
4207   021204  001116  001376  001126  DT20:   .WORD   $ERRPC,ASR,$BDDAT,$GDDAT,0
4208   021212  001124  000000
4209   021216  001116  001376  001126  DT22:   .WORD   $ERrPC,ASR,$BDDAT,$TMP0,0
4210   021224  001424  000000
4211   021230  001116  001424  000000  DT26:   .WORD   $ERRPC,$TMp0,0
4212
4213   021236  000000  000000          DF0:    .WORD   0,0
4214
4215           000001                          .END
```

```
ABASE = 170420          ABR     001400          ACDW1 = 000000          ACDW2 = 000000
ACPUOP= 000000          ADDW0 = 000000          ADDW1 = 000000          ADDW10= 000000
ADDW11= 000000          ADDW12= 000000          ADDW13= 000000          ADDW14= 000000
ADDW15= 000000          ADDW2 = 000000          ADDW3 = 000000          ADDW4 = 000000
ADDW5 = 000000          ADDW6 = 000000          ADDW7 = 000000          ADDW8 = 000000
ADDW9 = 000000          ADEVCT= 000000          ADEVM = 000000          AENV  = 000000
AENVM = 000000          AFATAL= 000000          AMADR1= 000000          AMADR2= 000000
AMADR3= 000000          AMADR4= 000000          AMAMS1= 000000          AMAMS2= 000000
AMAMS3= 000000          AMAMS4= 000000          AMSGAD= 000000          AMSGLG= 000000
AMSGTY= 000000          AMTYP1= 000000          AMTYP2= 000000          AMTYP3= 000000
AMTYP4= 000000          ANYKEY  014704          ANY2    014744          APASS = 000000
APRIOR= 000200          APTCSU= 000040          APTENV= 000001          APTSIZ= 000200
APTSPO= 000100          ASK     001452          ASR     001376          ASWREG= 000000
ATESTN= 000000          AUNIT = 000000          AUSWR = 000000          AVECT1= 000440
AVECT2= 000000          BIT0  = 000001          BIT00 = 000001          BIT01 = 000002
BIT02 = 000004          BIT03 = 000010          BIT04 = 000020          BIT05 = 000040
BIT06 = 000100          BIT07 = 000200          BIT08 = 000400          BIT09 = 001000
BIT1  = 000002          BIT10 = 002000          BIT11 = 004000          BIT12 = 010000
BIT13 = 020000          BIT14 = 040000          BIT15 = 100000          BIT2  = 000004
BIT3  = 000010          BIT4  = 000020          BIT5  = 000040          BIT6  = 000100
BIT7  = 000200          BIT8  = 000400          BIT9  = 001000          BPTVEC= 000014
CKSWR = 104407          CR    = 000015          CRLF  = 000200          DDISP = 177570
DFO     021236          DH1     020615          DH12    020721          DH20    020736
DH26    020771          DH3     020641          DH4A    020665          DISPLA  001142
DISPRE  000174          DR      001414          DR2     001416          DSWR  = 177570
DT1     021142          DT12    021176          DT20    021204          DT22    021216
DT26    021230          DT3     021154          DT4     021166          DWARF   001450
DWARFT= 001454          EMTVEC= 000030          EM1     020272          EM11    020444
EM12    020463          EM16    020512          EM2     020324          EM20    020543
EM26    020572          EM3     020352          EM4     020400          EM5     020421
ENDP    013644          ERCNT   001436          ERRVEC= 000004          EXS     001444
GTSWR = 104406          HT    = 000011          IOTRD   020032          IOTST1  015034
IOTST2  015112          IOTST3  015160          IOTVEC= 000020          LCNT    001446
LF    = 000012          LOOP    002312          MDEVCT  001440          PC    =%000007
PIRQ  = 177772          PIRQVE= 000240          PRIOR   001412          PR0   = 000000
PR1   = 000040          PR2   = 000100          PR3   = 000140          PR4   = 000200
PR5   = 000240          PR6   = 000300          PR7   = 000340          PS    = 177776
PSW   = 177776          PWRVEC= 000024          RDCHR = 104410          RDLIN = 104411
RESVEC= 000010          ROTATE  001432          RSTART  002162          R0    =%000000
R1    =%000001          R2    =%000002          R3    =%000003          R4    =%000004
R5    =%000005          R6    =%000006          R7    =%000007          SP    =%000006
STACK = 001100          START = 001534          STKLMT= 177774          SWR     001140
SWREG   000176          SW0   = 000001          SW00  = 000001          SW01  = 000002
SW02  = 000004          SW03  = 000010          SW04  = 000020          SW05  = 000040
SW06  = 000100          SW07  = 000200          SW08  = 000400          SW09  = 001000
SW1   = 000002          SW10  = 002000          SW11  = 004000          SW12  = 010000
SW13  = 020000          SW14  = 040000          SW15  = 100000          SW2   = 000004
SW3   = 000010          SW4   = 000020          SW5   = 000040          SW6   = 000100
SW7   = 000200          SW8   = 000400          SW9   = 001000          TBITVE= 000014
TKVEC = 000060          TPVEC = 000064          TRAPVE= 000034          TRFR0   020210
TRT0    020206          TRTVEC= 000014          TSCLC   001420          TSCLD   001422
TSTCNT  001442          TSTSTR= 001474          TST1    002400          TST10   003230
TST11   003326          TST12   003424          TST13   003522          TST14   003620
TST15   003716          TST16   004006          TST17   004076          TST2    002500
```

```
TST20   004150    TST21   004232    TST22   004360    TST23   004506
TST24   004602    TST25   004662    TST26   004766    TST27   005034
TST3    002542    TST30   005102    TST31   005212    TST32   005424
TST33   005546    TST34   005610    TST35   005656    TST36   006006
TST37   006136    TST4    002640    TST40   006266    TST41   006416
TST42   006546    TST43   006706    TST44   007050    TST45   007110
TST46   007236    TST47   007350    TST5    002736    TST50   007454
TST51   007530    TST52   007604    TST53   007670    TST54   007762
TST55   010116    TST56   010250    TST57   010404    TST6    003034
TST60   010540    TST61   010674    TST62   011030    TST63   011164
TST64   011442    TST65   011652    TST66   012162    TST67   012426
TST7    003132    TST70   012776    TST71   013110    TST72   013254
TST73   013642    TYPBN =  104405   TYPE  =  104401   TYPOC =  104402
TYPON =  104404   TYPOS =  104403   UTEST   001434    VECTP   001404
VECT1   001402    VECT2   001406    VECT2P  001410    WRNM1   021015
WSTART=  001514   $APTHD  001000    $ATYC   017432    $ATY1   017406
$ATY3   017414    $ATY4   017424    $AUTOB  001134    $BASE   001250
$BDADR  001122    $BDDAT  001126    $BELL   001164    $BIN    015522
$CDW1   001254    $CHARC  017402    $CKSWR  016350    $CMTAG  001100
$CM3  =  000000   $CNTLG  017075    $CNTLU  017070    $CPUOP  001222
$CRLF   001171    $DEVCT  001204    $DEVM   001252    $DOAGN  014674
$ENDAD  014664    $ENDCT  014432    $ENULL  014700    $ENV    001214
$ENVM   001215    $EOP    014376    $EOpCT  014424    $ERFLG  001103
$ERMAX  001115    $ERROR  015524    $ERRPC  001116    $ERRTB  001256
$ERRTY  015732    $ERTTL  001112    $ESCAP  001162    $ETABL  001214
$ETEND  001256    $FATAL  001176    $FFLG   017652    $FILLC  001156
$FIlLS  001155    $GDADR  001120    $GDDAT  001124    $GET42  014654
$GTSWR  016420    $HD   =  000001   $HIBTS  001000    $ICNT   001104
$ILLUP  020014    $INTAG  001135    $ITEMB  001114    $LF     001172
$LFLG   017651    $LPADR  001106    $LPERR  001110    $MADR1  001226
$MADR2  001232    $MADR3  001236    $MADR4  001242    $MAIL   001174
$MAMS1  001224    $MAMS2  001230    $MAMS3  001234    $MAMS4  001240
$MBADR  001002    $MFLG   017650    $MNEW   017113    $MSGAD  001210
$MSGLG  001212    $MSGTY  001174    $MSWR   017102    $MTYP1  001225
$MTYP2  001231    $MTYP3  001235    $MTYP4  001241    $MXCNT  016346
$NULL   001154    $NWTST=  000001   $OCNT   015444    $OMODE  015446
$OVER   016332    $PASS   001202    $PASTM  001006    $POWER  020022
$PWRDN  017654    $PWRMG  020010    $PWRUP  017726    $QUES   001170
$RDCHR  016632    $RDLIN  016752    $RDSZ =  000010   $RTNAD  014676
$SAVR6  020020    $SCOPE  016066    $SETUP=  000117   $STUP =  177777
$SVLAD  016276    $SVPC =  000254   $SWR  =  167400   $SWREG  001216
$SWRMK=  000000   $TESiN  001200    $TIMES  001160    $TKB    001146
$TKS    001144    $TMP0   001424    $TMP1   001426    $TMP3   001430
$TN   =  000074   $TPB    001152    $TPFLG  001157    $TPS    001150
$TRAP   020212    $TRAP2  020234    $TRP  =  000012   $TRPAD  020246
$TSTM   001004    $TSTNM  001102    $TTYIN  017060    $TYPBN  015450
$TYpE   017124    $TYPEC  017336    $TYPEX  017404    $TYPOC  0i5246
$TYPON  015262    $TYPOS  015222    $UNIT   001206    $UNIiM  001010
$USWR   001220    $VECT1  001244    $VECT2  001246    $XTSTR  016102
%%GET4=  000000   $OFILL  015445    .$X   =  001000    .       =  021242

ERRORS DETECTED:  0
```