KWV11

KWV11A DIAGNOSTIC
CVKWAC0

AH-8224C-MC

FICHE 1 OF 1

JAN 1980
digital
MADE IN USA

IDENTIFICATION

Product Code: AC-8222C-MC

Product Name: CVKWAC0 KWV11A Diagnostic

Date Created: October 1976

Date Revised: July 1979

Maintainer: Diagnostic Engineering

Table of Contents

1.0 ABSTRACT

This program allows the user check-out or debug the KWV11A, Programmable Real-Time Clock. The logic test is self contained and needs no external maintenance hardware or operator intervention with only one exception: If the customer hardware connected to the KWV11 could inject signals on ST2, ST1, or SLAVE IN inputs, it must be disconnected.

Even though the KWV11 is a Q BUS option, this program was designed to run on any PDP-11 Family computer. If the user is unfamilar with an LSI-11 he should review sections 8.4 and 8.5. A software switch register is included with this program.

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 (i.e., TTY).
2. KWV11 under test.

2.2 Storage

This program occupies and uses 8K of memory.

3.0 LOADING PROCEDURE

3.1 Method

Standards procedure for normal binary tapes should be followed.

1. Absolute loader must be in memory.
2. Place binary tape in reader.
3. 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 KWV11 with a standard address and vector. If any of these are different on the KW11K you are testing, change the corresponding location in memory before starting this test.

```
LOCATION    TAG       CURRENT CONTENTS    COMMENTS
--------    ---       ----------------    --------

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

4.0 STARTING PROCEDURE

4.1 Control Switch Setting

Before starting the diagnostic, set all switch register bits as desired, see section 5.1.

4.2 Starting Addresses

200 Start of Logic Tests
204 Restart Address for Logic Test
210 I/O Signal Test #1
214 I/O Signal Test #2
220 I/O Signal Test #3
230 Production Starting Address
240 Testor Starting Address

4.3 Program and/or Operator Action

All switches in switch pack 2 should be in the "OFF" position except when instructed.

1. Load program into memory.
2. Enter keyboard "ODT".
3. Alter location "$SWREG" (Address 176) 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

All switches in switch pack 2 should be in the "OFF" position except when instructed.

5.3.1 Logic Test

The first pass through the program will be made with iterations inhibited. Successive passes will enable iterations if SWR11=0.

If not inhibited by APT, the program will look for more KWV11's to exercise, one pass will exercise all KWV11's.

If four units are detected, the following will be typed:

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

At End of Pass when all units have been tested, the following typeout will occur:

```
'ENDPASS 12  -  TOTAL ERRORS 4 THERE ARE 4  (OCTAL)  UNITS -
 GOOD UNITS (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 KWV11 it detects on the system. To inhibit this feature, set bit 15 of location "$ENVM". Also, to test an individual KWV11 in a group, set this bit and refer to section 3.2 for changing the base address of the KWV11 under test.

6.0 ERRORS

6.1 Error Printout

Printout varies with the error detected. The error PC typed out is the actual location of the error call.

A HALT at location '$TYPE'+10 when running with no terminal indicates an error has occurred. To find out the number of the error, examine location '$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 '$TSTNM' and found a 5(101) we go to location '$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 S/B
        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

Bus errors will cause a Halt to the routine "IOTRD". The address that caused this trap will be in address "TRTC".

7.0 RESTRICTIONS

All switches in switch pack 2 should be in the "OFF" position except when instructed.

7.1 External Inputs

External inputs such as "SLAVE IN", "ST1" and "ST2" must not be connected to any customer hardware that might generate these signal while the diagnostic is running.

7.2 Starting Restriction

If a free-running clock, such as 60Hz from the power supply, is attached to the "BEVNT" bus line on both Rev level C/D and E systems, an interrupt to location 100 will occur when using the "G" and "L" commands prior to executing the first instruction. Therefore this program can not disable the BEVNT bus line by inhibiting interrupts.

User systems requiring a free-running clock attached to the BEVNT bus line can temporarily avoid this situation by setting the PSW(RS) to 200, loading the PC with the starting address instead of using the "G" command, and then using the "P" command. Before using the "L" command, the PSW(RS) can be set to 200, thereby inhibiting interrupts, to avoid receiving the event interrupt after loading the ABS loader.

7.3 Possible Program ''BOMBS''

The first two tests of this program check to see if the KWV11 responds to the address the program thinks its at. If the KWV11 does not respond, a bus error occurs. Also bus errors can occur during the time the program sizes to see how many KWV11 are on you system.

For more information on the next subject, see JAN. 1976 LSI-11 ENGINEERING BULLETIN issued by The Digital Components Group.

Bus errors may alter the preset contents of location 4 before the trap is executed, thereby transferring program control to area in the program that was not set up to handle the trap. If this happens, the program will ''BOMB'' and possibly rewite parts of itself.

8.0 MISCELLANEOUS

8.1 Power Fail

After a power failure occurs, the program execution will continue at the point where the power occurred. The program will type ''POWER''.

8.2 XXDP, ACT, APT

The program is chainable under XXDP, ACT, or APT. Although ''APT Hooks'' have been installed, they have not been tested.

8.3 Execution Time

0.5 minutes (30 sec) iteration inhibited - no errors
2.5 minutes (150 sec) with iterations - no errors

8.4 LSI-11 "ODT" Commands

FORMAT	DESCRIPTION
<CR> return	Close opened location and accept next command.
<LF> line feed	Close current location; open next sequential location.
^(uparrow)	Open previous location.
_ (left arrow)	Take contents of opened location, indexed by contents of PC, and open that location.
@	Take contents of opened location as absolute address and open that location.
R/	Open the word at location R.
/	Reopen the last location.
$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 command listed in section 8.4.

8.6 Use of Program Software SWR

The program software switch register is enabled if

1. No hardware SWR exists;

2. If you start with all ones (SWR=177777) in the switch register.

The software switch register may be changed by typing ^G (Control and letter G keys typed simultaneously). When ^G is typed, the program responds by typing "SWR=XXXXXX" where XXXXXX equals the former contents of the switch register.

If you wish to keep the current value, type <CR>. If you wish to change the value, type the new value followed by a <CR>.

It is important to note that the diagnostic is not running after the ^G until a <CR> is typed.

8.7 Special I/O Signal Tests

Three tests were included to enable checkout of I/O signals: ST1, ST2, and Clock Overflow. These tests have a special starting address. Since end-passes are immediate, no "END of Pass" message is reported. Errors are reported by typing out the PC where the error was detected. When started, the program remains in a loop generating and detecting the specified signals. HALT ON ERROR and INHIBIT ERROR typeout options may be used.

Logic test must have already been run on the KWV11.

8.7.1 I/O Signal Test #1 ST1 IN, ST2 OUT

Switch pack S2 must be set up as follows:

SWITCH	STATE
1	OFF
2	ON
3	OFF
4	OFF
5	ON
6	ON
7	not used

The following jumper must be installed.

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	OFF
6	ON
7	not used

The following jumper must be installed.

J1-RR (clock overflow) to J1-TT (ST2 in)

Load and start at location 214.

8.7.3 I/O Signal Test #3 ST1 out and ST2 in

Switch Pack S2 must be set up as follows:

SWITCH	STATE
1	OFF
2	OFF
3	OFF
4	ON
5	ON
6	ON
7	not used

The following jumper must be installed:

J1-UU (ST1 out) to J1-TT (ST2 in)

Load and start at location 220.

8.8 Production Starting Address

A special starting address has been provided for In-house production to use to start the logic diagnostic and inform the test that production is using it.

In the field only enough addresses were alloted for 4 sequential KWV11s. When the logic tests are started at location 200, we only auto-size up to 4 KWV11s.

In house testing may wish to exercise up to 16 KWV11s at one time. The logic tests may be started at location 230 and the program will auto size up to 16 KWV11s.

8.9 Testor Starting Address

A special starting address has been provided for manufacturing to use to start the logic diagnostic and inform the program that the clock module is cabled to an in-house testor.

Manual intervention is needed in this sequence of testing. The program will type out all instructions. A cable should connect J1 on the clock module to J10 on the testor. Switches 1 and 3 of S2 (on the clock module) should be on, all other switches on S2 should be off.

8.10 Trap Catcher

The Trap Catcher in this diagnostic employs a new concept. This concept will enable the user of this diagnostic to gain more knowledge of the events that lead the program to this area.

The Trap Catch consists of PC+2 and JSR PC,R0. (i.e., Location 300 would contain 302 and location 302 would contain 4700.)

When a device interrupts unexpectedly to the Trap Catcher, it would pick up the PC+2 of the trap as an address of the interrupt service routine.

The program would then pick up "4700" as the new PSW. Bit 7 of the new PSW having been set, would cause further interrupts from happening. When the CPU attempts to execute "4700" (JSR PC,R0), a Buss-time-out trap will occur to location 4. Location 4 contains a pointer to "IOTRD", a routine that will report the trap as an error.

To guard against "Real" Bus errors routing us through location 4 to "IOTRD", we check to see if the trap that brought us to location 4 really came from the Trap Catcher area. If not we'll halt and leave the Trap Address in "TRTO".

More about the interrupt error can be found in the description of the error in the program listing in the routine "IOTRD".

```
                                .NLIST  MC,MD,CND
                                .LIST   ME
                                .ENABL  ABS
                                .ENABL  AMA
          167400                $SWR=   167400


                        .TITLE  KWV11A DISAGNOSTIC  MAINDEC-11-CVKWA-C
                        ;*COPYRIGHT (C) 1979
                        ;*DIGITAL EQUIPMENT CORP.
                        ;*MAYNARD, MASS. 01754
                        ;*
                        ;*
                        ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                        ;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
                        ;*
          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,R0'' SEQUENCE TO CATCH ILLEGAL INTERRUPTS.
                        ;*AND INTERRUPTS TO THE WRONG VECTOR.
                        ;*LOCATION 0 CONTAINS A 0 TO CATCH IMPROPERLY LOADED
                        ;*VECTORS.
          000004        .=4
000004    017102  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
000100    000104  000200  000002        .WORD   104,200,2       ;IF 'B EVENT'ON Q-BUS IS
                                                        ;CONNECTED,WE NEED A WAY OF
                                                        ;IGNORING ITS INTERRUPTS.

          000200        .=200
000200    000137  001506        JMP     @#START
000204    000137  001444        JMP     @#QSTART
```

```
                                                                    G 2
    88  000210  000137  014046                JMP     @#OITST1
    89  000214  000137  014136                JMP     @#OITST2
    90  000220  000137  014216                JMP     @#OITST3
    91
    92          000230                        .=230
    93  000230  000137  001472                JMP     @#WSTART            ;WESTFIELD STARTING ADDRESS
    94          000240                        .=240
    95  000240  000137  001456                JMP     @#TSTSTR            ;ALL TESTER TESTS
    96                                                                     ;IF STARTED HERE.
    97                                                                     ;ALLOWS PRODUCTION TO EXERCISE
    98                                                                     ;UP TO 16 CLOCKS.NORMAL=4.
    99
   100                              .SBTTL  BASIC DEFINITIONS
   (1)
   (1)                              ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
   (1)          001100              STACK=  1100
   (1)                              .EQUIV  EMT,ERROR       ;;BASIC DEFINITION OF ERROR CALL
   (1)                              .EQUIV  IOT,SCOPE       ;;BASIC DEFINITION OF SCOPE CALL
   (1)
   (1)                              ;*MISCELLANEOUS DEFINITIONS
   (1)          000011              HT=     11              ;;CODE FOR HORIZONTAL TAB
   (1)          000012              LF=     12              ;;CODE FOR LINE FEED
   (1)          000015              CR=     15              ;;CODE FOR CARRIAGE RETURN
   (1)          000200              CRLF=   200             ;;CODE FOR CARRIAGE RETURN-LINE FEED
   (1)          177776              PS=     177776          ;;PROCESSOR STATUS WORD
   (1)                              .EQUIV  PS,PSW
   (1)          177774              STKLMT= 177774          ;;STACK LIMIT REGISTER
   (1)          177772              PIRQ=   177772          ;;PROGRAM INTERRUPT REQUEST REGISTER
   (1)          177570              DSWR=   177570          ;;HARDWARE SWITCH REGISTER
   (1)          177570              DDISP=  177570          ;;HARDWARE DISPLAY REGISTER
   (1)
   (1)                              ;*GENERAL PURPOSE REGISTER DEFINITIONS
   (1)          000000              R0=     %0              ;;GENERAL REGISTER
   (1)          000001              R1=     %1              ;;GENERAL REGISTER
   (1)          000002              R2=     %2              ;;GENERAL REGISTER
   (1)          000003              R3=     %3              ;;GENERAL REGISTER
   (1)          000004              R4=     %4              ;;GENERAL REGISTER
   (1)          000005              R5=     %5              ;;GENERAL REGISTER
   (1)          000006              R6=     %6              ;;GENERAL REGISTER
   (1)          000007              R7=     %7              ;;GENERAL REGISTER
   (1)          000006              SP=     %6              ;;STACK POINTER
   (1)          000007              PC=     %7              ;;PROGRAM COUNTER
   (1)
   (1)                              ;*PRIORITY LEVEL DEFINITIONS
   (1)          000000              PR0=    0               ;;PRIORITY LEVEL 0
   (1)          000040              PR1=    40              ;;PRIORITY LEVEL 1
   (1)          000100              PR2=    100             ;;PRIORITY LEVEL 2
   (1)          000140              PR3=    140             ;;PRIORITY LEVEL 3
   (1)          000200              PR4=    200             ;;PRIORITY LEVEL 4
   (1)          000240              PR5=    240             ;;PRIORITY LEVEL 5
   (1)          000300              PR6=    300             ;;PRIORITY LEVEL 6
   (1)          000340              PR7=    340             ;;PRIORITY LEVEL 7
   (1)
   (1)                              ;*''SWITCH REGISTER'' SWITCH DEFINITIONS
   (1)          100000              SW15=   100000
   (1)          040000              SW14=   40000
```

```
(1)          020000                  SW13=   20000
(1)          010000                  SW12=   10000
(1)          004000                  SW11=   4000
(1)          002000                  SW10=   2000
(1)          001000                  SW09=   1000
(1)          000400                  SW08=   400
(1)          000200                  SW07=   200
(1)          000100                  SW06=   100
(1)          000040                  SW05=   40
(1)          000020                  SW04=   20
(1)          000010                  SW03=   10
(1)          000004                  SW02=   4
(1)          000002                  SW01=   2
(1)          000001                  SW00=   1
(1)                                  .EQUIV  SW09,SW9
(1)                                  .EQUIV  SW08,SW8
(1)                                  .EQUIV  SW07,SW7
(1)                                  .EQUIV  SW06,SW6
(1)                                  .EQUIV  SW05,SW5
(1)                                  .EQUIV  SW04,SW4
(1)                                  .EQUIV  SW03,SW3
(1)                                  .EQUIV  SW02,SW2
(1)                                  .EQUIV  SW01,SW1
(1)                                  .EQUIV  SW00,SW0
(1)
(1)                                  ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
(1)          100000                  BIT15=  100000
(1)          040000                  BIT14=  40000
(1)          020000                  BIT13=  20000
(1)          010000                  BIT12=  10000
(1)          004000                  BIT11=  4000
(1)          002000                  BIT10=  2000
(1)          001000                  BIT09=  1000
(1)          000400                  BIT08=  400
(1)          000200                  BIT07=  200
(1)          000100                  BIT06=  100
(1)          000040                  BIT05=  40
(1)          000020                  BIT04=  20
(1)          000010                  BIT03=  10
(1)          000004                  BIT02=  4
(1)          000002                  BIT01=  2
(1)          000001                  BIT00=  1
(1)                                  .EQUIV  BIT09,BIT9
(1)                                  .EQUIV  BIT08,BIT8
(1)                                  .EQUIV  BIT07,BIT7
(1)                                  .EQUIV  BIT06,BIT6
(1)                                  .EQUIV  BIT05,BIT5
(1)                                  .EQUIV  BIT04,BIT4
(1)                                  .EQUIV  BIT03,BIT3
(1)                                  .EQUIV  BIT02,BIT2
(1)                                  .EQUIV  BIT01,BIT1
(1)                                  .EQUIV  BIT00,BIT0
(1)
(1)                                  ;*BASIC ''CPU'' TRAP VECTOR ADDRESSES
(1)          000004                  ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
(1)          000010                  RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
```

```
(1)          000014                  TBITVEC=14              ;;"T" BIT
(1)          000014                  TRTVEC= 14              ;;TRACE TRAP
(1)          000014                  BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
(1)          000020                  IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
(1)          000024                  PWRVEC= 24              ;;POWER FAIL
(1)          000030                  EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
(1)          000034                  TRAPVEC=34              ;;"TRAP" TRAP
(1)          000060                  TKVEC=  60              ;;TTY KEYBOARD VECTOR
(1)          000064                  TPVEC=  64              ;;TTY PRINTER VECTOR
(1)          000240                  PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR
101
102          170420                          ABASE=  170420
103          000440                          AVECT1= 440
104          000200                          APRIOR= 200
105
106          167400                          $SWR=   167400
107          000001                          $TN=    1
108
109                                  .SBTTL  ACT11 HOOKS
(1)
(2)                                  ;;******************************************************************
(1)                                  ;HOOKS REQUIRED BY ACT11
(1)          000244                          $SVPC=.                 ;SAVE PC
(1)          000046                          .=46
(1)  000046  013720                          $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
(1)          000052                          .=52
(1)  000052  000000                          .WORD   0               ;;2)SET LOC.52 TO ZERO
(1)          000244                          .=$SVPC                 ;; RESTORE PC
110          001000                          .=1000
111                                  .SBTTL  APT PARAMETER BLOCK
(1)
(2)                                  ;;******************************************************************
(1)                                  ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
(2)                                  ;;******************************************************************
(1)          001000                          .$X=.   ;;SAVE CURRENT LOCATION
(1)          000024                          .=24    ;;SET POWER FAIL TO POINT TO START OF PROGRAM
(1)  000024  000200                          200     ;;FOR APT START UP
(1)          000044                          .=44    ;;POINT TO APT INDIRECT ADDRESS PNTR.
(1)  000044  001000                          $APTHDR ;;POINT TO APT HEADER BLOCK
(1)          001000                          .=.$X   ;;RESET LOCATION COUNTER
(2)                                  ;;******************************************************************
(1)                                  ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
(1)                                  ;INTERFACE SPEC.
(1)
(1)  001000                          $APTHD:
(1)  001000  000000                  $HIBTS: .WORD   0       ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
(1)  001002  001174                  $MBADR: .WORD   $MAIL   ;;ADDRESS OF APT MAILBOX (BITS 0-15)
(1)  001004  000002                  $TSTM:  .WORD   2       ;;RUN TIM OF LONGEST TEST
(1)  001006  000170                  $PASTM: .WORD   120.    ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
(1)  001010  000170                  $UNITM: .WORD   120.    ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
(1)  001012  000031                          .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
```

```
                                .SBTTL  COMMON TAGS

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

                        001100          .=1100
001100                  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                  001174  $MAIL:                          ;;APT MAILBOX
001174  000000                 $MSGTY:  .WORD   AMSGTY          ;;MESSAGE TYPE CODE
001176  000000                 $FATAL:  .WORD   AFATAL          ;;FATAL ERROR NUMBER
001200  000000                 $TESTN:  .WORD   ATESTN          ;;TEST NUMBER
001202  000000                 $PASS:   .WORD   APASS           ;;PASS COUNT
001204  000000                 $DEVCT:  .WORD   ADEVCT          ;;DEVICE COUNT
001206  000000                 $UNIT:   .WORD   AUNIT           ;;I/O UNIT NUMBER
001210  000000                 $MSGAD:  .WORD   AMSGAD          ;;MESSAGE ADDRESS
```

```
(2)  001212  000000                $MSGLG: .WORD   AMSGLG  ;;MESSAGE LENGTH
(2)  001214                         $ETABLE:                ;;APT ENVIRONMENT TABLE
(2)  001214      000                $ENV:   .BYTE   AENV    ;;ENVIRONMENT BYTE
(2)  001215      000                $ENVM:  .BYTE   AENVM   ;;ENVIRONMENT MODE BITS
(2)  001216  000000                $SWREG: .WORD   ASWREG  ;;APT SWITCH REGISTER
(2)  001220  000000                $USWR:  .WORD   AUSWR   ;;USER SWITCHES
(2)  001222  000000                $CPUOP: .WORD   ACPUOP  ;;CPU TYPE,OPTIONS
(2)                                 ;*                      BITS 15-11=CPU TYPE
(2)                                 ;*                              11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
(2)                                 ;*                              11/70=06,PDQ=07,Q=10
(2)                                 ;*                      BIT 10=REAL TIME CLOCK
(2)                                 ;*                      BIT  9=FLOATING POINT PROCESSOR
(2)                                 ;*                      BIT  8=MEMORY MANAGEMENT
(2)  001224      000                $MAMS1: .BYTE   AMAMS1  ;;HIGH ADDRESS,M.S. BYTE
(2)  001225      000                $MTYP1: .BYTE   AMTYP1  ;;MEM. TYPE,BLK#1
(2)                                 ;*                      MEM.TYPE BYTE   --  (HIGH BYTE)
(2)                                 ;*                              900 NSEC CORE=001
(2)                                 ;*                              300 NSEC BIPOLAR=002
(2)                                 ;*                              500 NSEC MOS=003
(2)  001226  000000                $MADR1: .WORD   AMADR1  ;;HIGH ADDRESS,BLK#1
(2)                                 ;*                        MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF ''TYPE'' ABOVE
(2)  001230      000                $MAMS2: .BYTE   AMAMS2  ;;HIGH ADDRESS,M.S. BYTE
(2)  001231      000                $MTYP2: .BYTE   AMTYP2  ;;MEM.TYPE,BLK#2
(2)  001232  000000                $MADR2: .WORD   AMADR2  ;;MEM.LAST ADDRESS,BLK#2
(2)  001234      000                $MAMS3: .BYTE   AMAMS3  ;;HIGH ADDRESS,M.S.BYTE
(2)  001235      000                $MTYP3: .BYTE   AMTYP3  ;;MEM.TYPE,BLK#3
(2)  001236  000000                $MADR3: .WORD   AMADR3  ;;MEM.LAST ADDRESS,BLK#3
(2)  001240      000                $MAMS4: .BYTE   AMAMS4  ;;HIGH ADDRESS,M.S.BYTE
(2)  001241      000                $MTYP4: .BYTE   AMTYP4  ;;MEM.TYPE,BLK#4
(2)  001242  000000                $MADR4: .WORD   AMADR4  ;;MEM.LAST ADDRESS,BLK#4
(2)  001244  000440                $VECT1: .WORD   AVECT1  ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
(2)  001246  000000                $VECT2: .WORD   AVECT2  ;;INTERRUPT VECTOR#2BUS PRIORITY#2
(2)  001250  170420                $BASE:  .WORD   ABASE   ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
(2)  001252  000000                $DEVM:  .WORD   ADEVM   ;;DEVICE MAP
(2)  001254  000000                $CDW1:  .WORD   ACDW1   ;;CONTROLLER DESCRIPTION WORD#1
(2)  001256                         $ETEND:
(2)                                 .MEXIT
```

```
(1)                                     .SBTTL  ERROR POINTER TABLE
(1)
(1)                                     ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
(1)                                     ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
(1)                                     ;*LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
(1)                                     ;*NOTE1:        IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
(1)                                     ;*NOTE2:        EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:
(1)
(1)                                     ;*      EM              ;;POINTS TO THE ERROR MESSAGE
(1)                                     ;*      DH              ;;POINTS TO THE DATA HEADER
(1)                                     ;*      DT              ;;POINTS TO THE DATA
(1)                                     ;*      DF              ;;POINTS TO THE DATA FORMAT
(1)
(1)
(1)  001256                             $ERRTB:
113
118
119                                     ;ITEM   1
120
121  001256  017232                             EM1                     ;CLOCK   SR FUNCTION ERROR
122  001260  017555                             DH1                     ;ERRPC   ASR     WAS     S/B
123  001262  017764                             DT1                     ;$ERRPC,ASR,$BDDAT,$GDDAT
124  001264  020062                             DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
(1)
125
126                                     ;ITEM   2
127
128  001266  017264                             EM2                     ;CLOCK   SR DATA ERROR
129  001270  017555                             DH1                     ;ERRPC   ASR     WAS     S/B
130  001272  017764                             DT1                     ;$ERRPC,ASR,$BDDAT,$GDDAT
131  001274  020062                             DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
(1)
132
133                                     ;ITEM   3
134
135  001276  017312                             EM3                     ;CLOCK  BR DATA ERROR
136  001300  017601                             DH3                     ;ERRPC   ABR     WAS
137  001302  017776                             DT3                     ;$ERRPC,ABR,$BDDAT,$GDDAT
138  001304  020062                             DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
(1)
139
140                                     ;ITEM   4
141
142  001306  017340                             EM4             ;INTERRUPT ERROR.
143  001310  017625                             DH4A            ;ERRPC   TO      ROM ADDR.
144  001312  020010                             DT4             ;$ERRPC,        TRTO,TRFRO
145  001314  020062                             DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
(1)
146
147                                     ;ITEM   5
148
149  001316  017361                             EM5                     ;CLOCK   COUNT REG ERROR
150  001320  017555                             DH1                     ;ERRPC   ASR   WAS   S/B
151  001322  017764                             DT1                     ;$ERRPC, ACR, $BDDAT, $GDDAT
152  001324  020062                             DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
(1)
```

```
                                        ;ITEM   6
                                
001326  017423                                  EM12                    ;CLOCK   COUNT FUNCTION ERROR
001330  017661                                  DH12                    ;ERRPC   ASR
001332  020020                                  DT12                    ;ERRPC, ASR
001334  020062                                  DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
(1)
                                        ;ITEM   7
                                
001336  017452                                  EM16                    ;CLOCK   INTERRUPT ERROR
001340  017661                                  DH12                    ;ERRPC   ASR
001342  020020                                  DT12                    ;$ERRPC, ASR
001344  020062                                  DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
(1)
                                        ;ITEM   10
                                
001346  017503                                  EM20                    ;CLOCK   REPEATABILITY ERROR
001350  017676                                  DH20                    ;ERROR   ASR   2ND CNT 1ST CNT  3RD CNT
001352  020026                                  DT20                    ;$ERRPC, ASR, $BDDAT, $GDDAT, $TMP0
001354  020062                                  DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
(1)
                                        ;ITEM   11
                                
001356  017404                                  EM11                    ;CLOCK   COUNT ERROR
001360  017555                                  DH1                     ;ERRPC   ASR   WAS   S/B
001362  020042                                  DT22                    ;$ERRPC, ASR, $BDDAT, $TMP0
001364  020062                                  DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
(1)
                                        ;ITEM   12
                                
001366  017532                                  EM26                    ;CLOCK ADDRESSING ERROR
001370  017737                                  DH26                    ;ERRPC   CLOCK ADDR.
001372  020054                                  DT26                    ;$ERRPC,$TMP0
001374  020062                                  DF0                     ;ALL NUMBERS ARE IN OCTAL FORM
(1)
                                
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  000000          $TMP0:  .WORD   0               ;TEMP STORAGE.
001422  000000          $TMP1:  .WORD   0               ;TMP STORAGE.
001424  000000          $TMP3:  .WORD   0
001426  000000          ROTATE: .WORD   0                       ;POINT TO DEVICE UNDER TEST.
001430  000000          UTEST:  .WORD   0                       ;KEEPS TRACK OF GOOD UNITS.
001432  000000          ERCNT:  .WORD   0                       ;COUNTS ERRORS.
```

```
  204  001434  000000                MDEVCT: .WORD   0                       ;COUNTS DEVICES TESTED.
  205  001436  000000                TSTCNT: .WORD   0                       ;MAX DEVICES TO BE TESTED.
  206  001440  000000                EXS:    .WORD   0                       ;=0, NORMAL: =1 SPECIAL TESTOR START, BY L+S @ 2
  207  001442  000000                LCNT:   .WORD   0                       ;TOTAL UNITS TESTED.
  208
  209
  211  001444  012737  002144  001420 QSTART: MOV     #RSTART,$TMP0           ;LOAD SETUP RETURN ADDRESS
  212  001452  000137  001526                JMP     INIT                    ;INIT THE PROGRAM VECTOR SPACE
  213
  214  001456  005237  001440        TSTSTR: INC     EXS                     ;SET FOR TESTOR.
  215  001462  012737  000020  001436        MOV     #16.,TSTCNT             ;ALLOW 16 UNITS
  216  001470  000413                       BR      1$
  217          001472                WSTART=.
  218  001472  012737  000020  001436        MOV     #16.,TSTCNT     ;TEST UP TO 16 UNITS.
  219  001500  005037  001440                CLR     EXS
  220  001504  000405                       BR      1$
  221          001506                START=.
  222  001506  012737  000004  001436        MOV     #4,TSTCNT       ;TEST UP TO FOUR UNITS.
  223  001514  005037  001440                CLR     EXS
  224  001520  012737  001774  001420 1$:     MOV     #ZSTART,$TMP0           ;LOAD SETUP RETURN
  225  001526                        INIT:
  (1)                             .SBTTL  INITIALIZE THE COMMON TAGS
  (1)                             ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
  (1)  001526  012706  001100            MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
  (1)  001532  005026                    CLR     (R6)+           ;;CLEAR MEMORY LOCATION
  (1)  001534  022706  001140            CMP     #SWR,R6 ;;DONE?
  (1)  001540  001374                    BNE     .-6             ;;LOOP BACK IF NO
  (1)  001542  012706  001100            MOV     #STACK,SP       ;;SETUP THE STACK POINTER
  (1)                             ;;INITIALIZE A FEW VECTORS
  (1)  001546  012737  015136  000020    MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
  (1)  001554  012737  000340  000022    MOV     #340,@#IOTVEC+2 ;;LEVEL 7
  (1)  001562  012737  014574  000030    MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
  (1)  001570  012737  000340  000032    MOV     #340,@#EMTVEC+2 ;;LEVEL 7
  (1)  001576  012737  017152  000034    MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
  (1)  001604  012737  000340  000036    MOV     #340,@#TRAPVEC+2;LEVEL 7
  (1)  001612  012737  016724  000024    MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
  (1)  001620  012737  000340  000026    MOV     #340,@#PWRVEC+2 ;;LEVEL 7
  (1)  001626  005037  C01160            CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
  (1)  001632  005037  001162            CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
  (1)  001636  112737  000001  001115    MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
  (1)  001644  012737  001644  001106    MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
  (1)  001652  012737  001652  001110    MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
  (2)                             ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
  (2)                             ;;EQUAL TO A ''-1'', SETUP FOR A SOFTWARE SWITCH REGISTER.
  (2)  001660  013746  000004            MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
  (2)  001664  012737  001720  000004    MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
  (2)  001672  012737  177570  001140    MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
  (2)  001700  012737  177570  001142    MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
  (2)  001706  022777  177777  177224    CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
  (2)  001714  001012                    BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
  (2)                                                            ;;AND  THE HARDWARE SWR IS NOT = -1
  (2)  001716  000403                    BR      65$             ;;BRANCH IF NO TIMEOUT
  (2)  001720  012716  001726    64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
  (2)  001724  000002                    RTI
  (2)  001726  012737  000176  001140    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
  (2)  001734  012737  000174  001142    MOV     #DISPREG,DISPLAY
```

```
(2)  001742  012637  000004          66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR
(1)
(2)  001746  005037  001202                  CLR     $PASS           ;;CLEAR PASS COUNT
(2)  001752  132737  000200  001215          BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
(2)  001760  001403                          BEQ     67$             ;;YES,USE NON-APT SWITCH
(2)  001762  012737  001216  001140          MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
(2)  001770                          67$:
226  001770  000177  177424                  JMP     @$TMP0                  ;EXIT PROGRAM VECTOR SETUP SPACE
227
228  001774                          ZSTART:
(1)
(1)  001774  012746  000340                  MOV     #340,-(SP)              ;SET CPU PRIORITY ON RETERN.
(1)  002000  012746  002006                  MOV     #64$,-(SP)              ;SHOW RETURN ADDRESS.
(1)  002004  000002                          RTI                             ;CAUSE A RETURN(PUTS STATUS IN STATUS REG.).
(1)  002006                          64$:
(1)
229  002006  005037  001204                  CLR     $DEVCT          ;ZERO DEVICE COUNT.
230  002012  012737  017102  000004          MOV     #IOTRD,@#ERRVEC ;FIX TRAP CATCHER.
231  002020  013737  001244  001402          MOV     $VECT1,VECT1    ;NOW FIX VECTOR ADDR.
232  002026  013737  001250  001376          MOV     $BASE,ASR       ;FIX ADDRESS OF CSR.
233
234                                  .SBTTL  TYPE PROGRAM NAME
(1)                                  ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
(1)  002034  005227  177777                  INC     #-1             ;;FIRST TIME?
(1)  002040  001041                          BNE     65$             ;;BRANCH IF NO
(1)  002042  104401  002110                  TYPE    ,66$            ;;TYPE ASCIZ STRING
(2)                                  .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
(2)  002046  005737  000042                  TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
(2)  002052  001012                          BNE     67$             ;;BRANCH IF YES
(2)  002054  123727  001214  000001          CMPB    $ENV,#1         ;;ARE WE RUNNING UNDER APT?
(2)  002062  001406                          BEQ     67$             ;;BRANCH IF YES
(2)  002064  023727  001140  000176          CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
(2)  002072  001005                          BNE     68$             ;;BRANCH IF NO
(2)  002074  104406                          GTSWR                   ;;GET SOFT-SWR SETTINGS
(2)  002076  000403                          BR      68$
(2)  002100  112737  000001  001134  67$:    MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
(2)  002106                          68$:
(1)  002106  000416                          BR      65$             ;;GET OVER THE ASCIZ
(1)                                  ;;66$:  .ASCIZ  <CRLF>#CVKWAC   KWV11 DIAGNOSTIC#<CRLF>
(1)  002144                          65$:
235  002144                          RSTART:
236  002144  005737  001440                  TST     EXS             ;TESTOR MODE ENABLED??
237  002150  001441                          BEQ     1$              ;NO DON'T TYPE NEXT MESSAGE.
238  002152  104401  002160                  TYPE    ,65$            ;;TYPE ASCIZ STRING
(1)  002156  000436                          BR      64$             ;;GET OVER THE ASCIZ
(1)                                  ;;65$:  .ASCIZ  <15><12>#TESTOR MODE ENABLED--SEE DOCUMENTATION FOR INSTRUCTIONS.#
(1)  002254                          64$:
239  002254                          1$:
(1)  002254  104401  002262                  TYPE    ,67$            ;;TYPE ASCIZ STRING
(1)  002260  000411                          BR      66$             ;;GET OVER THE ASCIZ
(1)                                  ;;67$:  .ASCIZ  <15><12>#TEST RUNNING...#
(1)  002304                          66$:
240  002304  005037  001434                  CLR     MDEVCT          ;TESTING FIRST UNIT.
241  002310  005037  001432                  CLR     ERCNT           ;NO ERRORS.
242  002314  005037  001202                  CLR     $PASS           ;NO PASSES.
243  002320  012737  000001  001426          MOV     #1,ROTATE       ;POINT TO FIRST UNIT.
```

```
                                        MOV     ROTATE,UTEST
                                LOOP:                                   ;COME HERE FOR NEXT UNIT,OR END PASS.

                                        BIC     #170000,VECT1           ;CLEAR OUT PRIORITY BITS.
                                        MOV     VECT1,VECTP             ;NOW FIX VECTOR +2 ADDR.
                                        ADD     #2,VECTP
                                        MOV     VECT1,VECT2             ;LETS FIX ST2 VECTOR ADDR.
                                        ADD     #4,VECT2                ;ITS 4 GREATER THEN THE 1ST.
                                        MOV     VECT2,VECT2P            ;VECTOR +2 ADDR.
                                        ADD     #2,VECT2P


                                        MOV     ASR,ABR                 ;FIX ADDR OF PRESET REG=
                                        ADD     #2,ABR                  ;CSR + 2

                                ;;**********************************************************************
                                ;*TEST 1         *TEST THE ADDRESSABILITY OF CLOCK CSR
                                ;;**********************************************************************
                                TST1:   NOP
                                        MOV     #50,$TIMES              ;;DO 50 ITERATIONS
                                        MOV     #1$,$LPADR              ;;SET SCOPE LOOP ADDRESS
                                        MOVB    #1,$TSTNM               ;SET TEST #1.
                                        MOV     #1$,$LPERR


                                1$:     MOV     @#ERRVEC,-(SP)          ;SAVE CONTENTS OF ADDRS 6.
                                        MOV     #2$,@#ERRVEC            ;SET TIME-OUT TRAP VECTOR TO HANDLER IN CASE.
                                                                        ;WE TIME-OUT WHEN ADDRESSING THE KW11.
                                        TST     @ASR                    ;ADDRESS THE CLOCK!
                                                                        ;IF CLOCK DOES NOT RETURN
                                                                        ;'BUS SSYN' THEN WE'LL TIME-OUT

                                        BR      3$                      ;THE CLOCK WAS THERE! EXIT SUB-TEST.
                                2$:
                                        ADD     #4,SP                           ;/ADD #4 TO STACK POINTER.
                                        MOV     ASR,$TMP0               ;FOR ERROR TYPEOUT.


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

                                        ERROR   12                      ;REPORT ERROR=CLOCK CSR 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.


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

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


                                ;;**********************************************************************
```

```
(4)                                     ;*TEST 2        *TEST THE ADDRESSABILITY OF CLOCK BUFFER REG.
(4)                                     ;;*****************************************************************
(3)  002514  000004                     TST2:   SCOPE
(1)
(1)
(1)  002516  013746  000004             1$:     MOV     @#ERRVEC,-(SP)  ;SAVE CONTENTS OF ADDRS 6.
(1)  002522  012737  002536  000004             MOV     #2$,@#ERRVEC    ;SET TIME-OUT TRAP VECTOR TO HANDLER IN CASE.
(1)                                                                     ;WE TIME-OUT WHEN ADDRESSING THE KW11.
(1)  002530  005777  176644                     TST     @ABR            ;ADDRESS THE CLOCK!
(1)                                                                     ;IF CLOCK DOES NOT RETURN
(1)                                                                     ;'BUS SSYN'' THEN WE'LL TIME-OUT
(1)
(1)  002534  000406                             BR      3$              ;THE CLOCK WAS THERE! EXIT SUB-TEST.
(1)  002536                             2$:
(2)  002536  062706  000004                     ADD     #4,SP                   ;/ADD #4 TO STACK POINTER.
(1)  002542  013737  001400  001420             MOV     ABR,$TMP0       ;FOR ERROR TYPEOUT.
(1)
(2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  002550  104012                             ERROR   12              ;REPORT ERROR=CLOCK BUFFER REG. FAILED TO RETURN
(1)                                                                     ;'BUS SSYN'' WHEN ADDRESSED.
(1)                                                                     ;NOTE: IF PROGRAM HAS INCORRECT
(1)                                                                     ;ADDRESS THEN WE MIG NOT BE
(1)                                                                     ;TALKING TO THE CLOCK. MAKE SURE
(1)                                                                     ;OF CLOCK ADDRESS.
(1)
(2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  002552  012637  000004             3$:     MOV     (SP)+,@#ERRVEC
(1)
303
341
342
```

```
(1)                                             ;/#
(5)                                             ;;*****************************************************************
(4)                                             ;*TEST 3        *TEST THAT CLOCK A STATUS REGISTER BIT 14 CAN BE SET AND CLEARED
(5)                                             ;*
(5)                                             ;*CLOCK STATUS REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
(5)                                             ;*F/FS OR GATES
(5)                                             ;*
(5)
(4)                                             ;;*****************************************************************
(3)  002556  000004                             TST3:   SCOPE
(2)  002560  012737  000100  001160                    MOV     #100,$TIMES     ;;DO 100 ITERATIONS
(1)
(1)  002566  005077  176604                             CLR     @ASR            ;/CLEAR THE STATUS REGISTER.
(1)  002572  052777  040000  176576                     BIS     #BIT14,@ASR     ;/SET BIT 14.
(1)  002600  012737  040000  001124                     MOV     #BIT14,$GDDAT   ;/SET FOR ERROR TYPEOUT S/B.
(1)  002606  017737  176564  001126                     MOV     @ASR,$BDDAT     ;/READ THE STATUS REGISTER.
(1)  002614  023737  001124  001126                     CMP     $GDDAT,$BDDAT   ;/DID BIT 14 AND ONLY BIT 14 SET?
(1)  002622  001402                                     BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(2)
                                                ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  002624  104002                                     ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER
(1)                                                                             ;/BIT 14 FAILED TO BIT SET.
(2)
                                                ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  002626  000412                                     BR      2$              ;/BR TO END SUBTEST.
(1)
(1)  002630  042777  040000  176540  1$:       BIC     #BIT14,@ASR     ;/TRY CLEARING BIT 14.
(1)  002636  005037  001124                     CLR     $GDDAT          ;/CLEAR S/B FOR TYPEOUT IF ANY.
(1)  002642  017737  176530  001126             MOV     @ASR,$BDDAT     ;/NOW READ IT BACK.
(1)  002650  001401                             BEQ     2$              ;/IF ZERO - NO ERROR!
(1)
(2)
                                                ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  002652  104002                                     ERROR   2               ;/ERROR - CLOCK A STATUS REGISTER.
(1)                                                                             ;/BIT 14 FAILED TO CLEAR.
(1)
(2)
                                                ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  002654                                     2$:
(1)
343
```

```
                                        ;/#
                                ;;**********************************************************************
                                ;*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
                                ;*
                                ;;**********************************************************************
(3)  002654  000004             TST4:   SCOPE
(2)  002656  012737  000100  001160        MOV     #100,$TIMES     ;;DO 100 ITERATIONS
(1)
(1)  002664  005077  176506                 CLR     @ASR            ;/CLEAR THE STATUS REGISTER.
(1)  002670  052777  020000  176500         BIS     #BIT13,@ASR     ;/SET BIT 13.
(1)  002676  012737  020000  001124         MOV     #BIT13,$GDDAT   ;/SET FOR ERROR TYPEOUT S/B.
(1)  002704  017737  176466  001126         MOV     @ASR,$BDDAT     ;/READ THE STATUS REGISTER.
(1)  002712  023737  001124  001126         CMP     $GDDAT,$BDDAT   ;/DID BIT 13 AND ONLY BIT 13 SET?
(1)  002720  001402                         BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(2)
                                ;;$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  002722  104002                        ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER
(1)                                                             ;/BIT 13 FAILED TO BIT SET.
(2)
                                ;;$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  002724  000412                         BR      2$              ;/BR TO END SUBTEST.
(1)
(1)  002726  042777  020000  176442  1$:    BIC     #BIT13,@ASR     ;/TRY CLEARING BIT 13.
(1)  002734  005037  001124                 CLR     $GDDAT          ;/CLEAR S/B FOR TYPEOUT IF ANY.
(1)  002740  017737  176432  001126         MOV     @ASR,$BDDAT     ;/NOW READ IT BACK.
(1)  002746  001401                         BEQ     2$              ;/IF ZERO - NO ERROR!
(1)
(2)
                                ;;$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  002750  104002                        ERROR   2               ;/ERROR - CLOCK A STATUS REGISTER.
(1)                                                             ;/BIT 13 FAILED TO CLEAR.
(1)
(2)
                                ;;$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  002752                     2$:
(1)
344
```

```
(1)                                             ;/#
(5)                                     ;;**********************************************************************
(4)                                     ;*TEST 5         *TEST THAT CLOCK A STATUS REGISTER BIT 11 CAN BE SET AND CLEARED
(5)                                     ;*
(5)                                     ;*CLOCK STATUS REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
(5)                                     ;*F/FS OR GATES
(5)                                     ;*
(5)
(4)                                     ;;**********************************************************************
(3)  002752  000004                     TST5:   SCOPE
(2)  002754  012737  000100  001160             MOV     #100,$TIMES     ;;DO 100 ITERATIONS
(1)
(1)  002762  005077  176410                      CLR     @ASR            ;/CLEAR THE STATUS REGISTER.
(1)  002766  052777  004000  176402              BIS     #BIT11,@ASR     ;/SET BIT 11.
(1)  002774  012737  004000  001124              MOV     #BIT11,$GDDAT   ;/SET FOR ERROR TYPEOUT S/B.
(1)  003002  017737  176370  001126              MOV     @ASR,$BDDAT     ;/READ THE STATUS REGISTER.
(1)  003010  023737  001124  001126              CMP     $GDDAT,$BDDAT   ;/DID BIT 11 AND ONLY BIT 11 SET?
(1)  003016  001402                              BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  003020  104002                              ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER
(1)                                                                     ;/BIT 11 FAILED TO BIT SET.
(2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  003022  000412                              BR      2$              ;/BR TO END SUBTEST.
(1)
(1)  003024  0427.7  004000  176344  1$:       BIC     #BIT11,@ASR     ;/TRY CLEARING BIT 11.
(1)  003032  005037  001124                      CLR     $GDDAT          ;/CLEAR S/B FOR TYPEOUT IF ANY.
(1)  003036  017737  176334  001126              MOV     @ASR,$BDDAT     ;/NOW READ IT BACK.
(1)  003044  001401                              BEQ     2$              ;/IF ZERO - NO ERROR!
(1)
(2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  003046  104002                              ERROR   2               ;/ERROR - CLOCK A STATUS REGISTER.
(1)                                                                     ;/BIT 11 FAILED TO CLEAR.
(1)
(2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  003050                             2$:
(1)
345
```

```
                                                        ;/#
                                                ;;**********************************************************************
                                                ;*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
                                                ;*
                                                ;;**********************************************************************
003050  000004                                  TST6:   SCOPE
003052  012737  000100  001160                          MOV     #100,$TIMES     ;;DO 100 ITERATIONS

003060  005077  176312                          CLR     @ASR            ;/CLEAR THE STATUS REGISTER.
003064  052777  000100  176304                  BIS     #BIT6,@ASR      ;/SET BIT 6.
003072  012737  000100  001124                  MOV     #BIT6,$GDDAT    ;/SET FOR ERROR TYPEOUT S/B.
003100  017737  176272  001126                  MOV     @ASR,$BDDAT     ;/READ THE STATUS REGISTER.
003106  023737  001124  001126                  CMP     $GDDAT,$BDDAT   ;/DID BIT 6 AND ONLY BIT 6 SET?
003114  001402                                  BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.

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

003116  104002                                  ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER
                                                                        ;/BIT 6 FAILED TO BIT SET.

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

003120  000412                                  BR      2$              ;/BR TO END SUBTEST.

003122  042777  000100  176246  1$:     BIC     #BIT6,@ASR      ;/TRY CLEARING BIT 6.
003130  005037  001124                  CLR     $GDDAT          ;/CLEAR S/B FOR TYPEOUT IF ANY.
003134  017737  176236  001126          MOV     @ASR,$BDDAT     ;/NOW READ IT BACK.
003142  001401                          BEQ     2$              ;/IF ZERO - NO ERROR!

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

003144  104002                                  ERROR   2               ;/ERROR - CLOCK A STATUS REGISTER.
                                                                        ;/BIT 6 FAILED TO CLEAR.

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

003146                                  2$:
346
```

```
(1)                                             ;/#
(5)                                     ;;**********************************************************************
(4)                                     ;*TEST 7         *TEST THAT CLOCK A STATUS REGISTER BIT 5 CAN BE SET AND CLEARED
(5)                                     ;*
(5)                                     ;*CLOCK STATUS REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
(5)                                     ;*F/FS OR GATES
(5)                                     ;*
(5)
(4)                                     ;;**********************************************************************
(3)  003146  000004                     TST7:   SCOPE
(2)  003150  012737  000100  001160             MOV     #100,$TIMES     ;;DO 100 ITERATIONS
(1)
(1)  003156  005077  176214             CLR     @ASR            ;/CLEAR THE STATUS REGISTER.
(1)  003162  052777  000040  176206     BIS     #BIT5,@ASR      ;/SET BIT 5.
(1)  003170  012737  000040  001124     MOV     #BIT5,$GDDAT    ;/SET FOR ERROR TYPEOUT S/B.
(1)  003176  017737  176174  001126     MOV     @ASR,$BDDAT     ;/READ THE STATUS REGISTER.
(1)  003204  023737  001124  001126     CMP     $GDDAT,$BDDAT   ;/DID BIT 5 AND ONLY BIT 5 SET?
(1)  003212  001402                     BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  003214  104002                             ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER
(1)                                                                     ;/BIT 5 FAILED TO BIT SET.
(2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  003216  000412                             BR      2$              ;/BR TO END SUBTEST.
(1)
(1)  003220  042777  000040  176150  1$:      BIC     #BIT5,@ASR      ;/TRY CLEARING BIT 5.
(1)  003226  005037  001124             CLR     $GDDAT          ;/CLEAR S/B FOR TYPEOUT IF ANY.
(1)  003232  017737  176140  001126     MOV     @ASR,$BDDAT     ;/NOW READ IT BACK.
(1)  003240  001401                     BEQ     2$              ;/IF ZERO - NO ERROR!
(1)
(2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  003242  104002                             ERROR   2               ;/ERROR - CLOCK A STATUS REGISTER.
(1)                                                                     ;/BIT 5 FAILED TO CLEAR.
(1)
(2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  003244                             2$:
(1)
347
```

```
(1)                                         ;/#
(5)                                 ;;**********************************************************************
(4)                                 ;*TEST 10        *TEST THAT CLOCK A STATUS REGISTER BIT 4 CAN BE SET AND CLEARED
(5)                                 ;*
(5)                                 ;*CLOCK STATUS REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
(5)                                 ;*F/FS OR GATES
(5)                                 ;*
(5)
(4)                                 ;;**********************************************************************
(3) 003244  000004                  TST10:  SCOPE
(2) 003246  012737  000100  001160          MOV     #100,$TIMES     ;;DO 100 ITERATIONS
(1)
(1) 003254  005077  176116                  CLR     @ASR            ;/CLEAR THE STATUS REGISTER.
(1) 003260  052777  000020  176110          BIS     #BIT4,@ASR      ;/SET BIT 4.
(1) 003266  012737  000020  001124          MOV     #BIT4,$GDDAT    ;/SET FOR ERROR TYPEOUT S/B.
(1) 003274  017737  176076  001126          MOV     @ASR,$BDDAT     ;/READ THE STATUS REGISTER.
(1) 003302  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;/DID BIT 4 AND ONLY BIT 4 SET?
(1) 003310  001402                          BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(2)
                                    ;;$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1) 003312  104002                          ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER
(1)                                                                 ;/BIT 4 FAILED TO BIT SET.
(2)
                                    ;;$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1) 003314  000412                          BR      2$              ;/BR TO END SUBTEST.
(1)
(1) 003316  042777  000020  176052  1$:     BIC     #BIT4,@ASR      ;/TRY CLEARING BIT 4.
(1) 003324  005037  001124                  CLR     $GDDAT          ;/CLEAR S/B FOR TYPEOUT IF ANY.
(1) 003330  017737  176042  001126          MOV     @ASR,$BDDAT     ;/NOW READ IT BACK.
(1) 003336  001401                          BEQ     2$              ;/IF ZERO - NO ERROR!
(1)
(2)
                                    ;;$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1) 003340  104002                          ERROR   2               ;/ERROR - CLOCK A STATUS REGISTER.
(1)                                                                 ;/BIT 4 FAILED TO CLEAR.
(1)
(2)
                                    ;;$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1) 003342                          2$:
(1)
348
```

```
(1)                                             ;/#
(5)                                     ;;**********************************************************************
(4)                                     ;*TEST 11        *TEST THAT CLOCK A STATUS REGISTER BIT 3 CAN BE SET AND CLEARED
(5)                                     ;*
(5)                                     ;*CLOCK STATUS REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
(5)                                     ;*F/FS OR GATES
(5)                                     ;*
(5)
(4)                                     ;;**********************************************************************
(3)  003342  000004                     TST11:  SCOPE
(2)  003344  012737  000100  001160             MOV     #100,$TIMES     ;;DO 100 ITERATIONS
(1)
(1)  003352  005077  176020             CLR     @ASR            ;/CLEAR THE STATUS REGISTER.
(1)  003356  052777  000010  176012     BIS     #BIT3,@ASR      ;/SET BIT 3.
(1)  003364  012737  000010  001124     MOV     #BIT3,$GDDAT    ;/SET FOR ERROR TYPEOUT S/B.
(1)  003372  017737  176000  001126     MOV     @ASR,$BDDAT     ;/READ THE STATUS REGISTER.
(1)  003400  023737  001124  001126     CMP     $GDDAT,$BDDAT   ;/DID BIT 3 AND ONLY BIT 3 SET?
(1)  003406  001402                     BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  003410  104002                             ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER
(1)                                                                     ;/BIT 3 FAILED TO BIT SET.
(2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  003412  000412                             BR      2$              ;/BR TO END SUBTEST.
(1)
(1)  003414  042777  000010  175754  1$:     BIC     #BIT3,@ASR      ;/TRY CLEARING BIT 3.
(1)  003422  005037  001124             CLR     $GDDAT          ;/CLEAR S/B FOR TYPEOUT IF ANY.
(1)  003426  017737  175744  001126     MOV     @ASR,$BDDAT     ;/NOW READ IT BACK.
(1)  003434  001401                     BEQ     2$              ;/IF ZERO - NO ERROR!
(1)
(2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  003436  104002                             ERROR   2               ;/ERROR - CLOCK A STATUS REGISTER.
(1)                                                                     ;/BIT 3 FAILED TO CLEAR.
(1)
(2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  003440                             2$:
(1)
349
```

```
(1)                                             ;/#
(5)                                     ;;**********************************************************************
(4)                                     ;*TEST 12        *TEST THAT CLOCK A STATUS REGISTER BIT 2 CAN BE SET AND CLEARED
(5)                                     ;*
(5)                                     ;*CLOCK STATUS REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
(5)                                     ;*F/FS OR GATES
(5)                                     ;*
(5)
(4)                                     ;;**********************************************************************
(3)  003440  000004                     TST12:  SCOPE
(2)  003442  012737  000100  001160             MOV     #100,$TIMES     ;;DO 100 ITERATIONS
(1)
(1)  003450  005077  175722                     CLR     @ASR            ;/CLEAR THE STATUS REGISTER.
(1)  003454  052777  000004  175714             BIS     #BIT2,@ASR      ;/SET BIT 2.
(1)  003462  012737  000004  001124             MOV     #BIT2,$GDDAT    ;/SET FOR ERROR TYPEOUT S/B.
(1)  003470  017737  175702  001126             MOV     @ASR,$BDDAT     ;/READ THE STATUS REGISTER.
(1)  003476  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;/DID BIT 2 AND ONLY BIT 2 SET?
(1)  003504  001402                             BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  003506  104002                             ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER
(1)                                                                     ;/BIT 2 FAILED TO BIT SET.
(2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  003510  000412                             BR      2$              ;/BR TO END SUBTEST.
(1)
(1)  003512  042777  000004  175656  1$:      BIC     #BIT2,@ASR      ;/TRY CLEARING BIT 2.
(1)  003520  005037  001124                     CLR     $GDDAT          ;/CLEAR S/B FOR TYPEOUT IF ANY.
(1)  003524  017737  175646  001126             MOV     @ASR,$BDDAT     ;/NOW READ IT BACK.
(1)  003532  001401                             BEQ     2$              ;/IF ZERO - NO ERROR!
(1)
(2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  003534  104002                             ERROR   2               ;/ERROR - CLOCK A STATUS REGISTER.
(1)                                                                     ;/BIT 2 FAILED TO CLEAR.
(1)
(2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  003536                             2$:
(1)
350
```

```
                                        T12     *TEST THAT CLOCK A STATUS REGISTER BIT 2 CAN BE SET AND CLEARED          SEQ 0038

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

                                        ;;**********************************************************************
003536  000004                          TST13:  SCOPE
003540  012737  000100  001160                  MOV     #100,$TIMES     ;;DO 100 ITERATIONS

003546  005077  175624                          CLR     @ASR            ;/CLEAR THE STATUS REGISTER.
003552  052777  000002  175616                  BIS     #BIT1,@ASR      ;/SET BIT 1.
003560  012737  000002  001124                  MOV     #BIT1,$GDDAT    ;/SET FOR ERROR TYPEOUT S/B.
003566  017737  175604  001126                  MOV     @ASR,$BDDAT     ;/READ THE STATUS REGISTER.
003574  023737  001124  001126                  CMP     $GDDAT,$BDDAT   ;/DID BIT 1 AND ONLY BIT 1 SET?
003602  001402                                  BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.

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

003604  104002                                  ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER
                                                                        ;/BIT 1 FAILED TO BIT SET.

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

003606  000412                                  BR      2$              ;/BR TO END SUBTEST.

003610  042777  000002  175560          1$:     BIC     #BIT1,@ASR      ;/TRY CLEARING BIT 1.
003616  005037  001124                          CLR     $GDDAT          ;/CLEAR S/B FOR TYPEOUT IF ANY.
003622  017737  175550  001126                  MOV     @ASR,$BDDAT     ;/NOW READ IT BACK.
003630  001401                                  BEQ     2$              ;/IF ZERO - NO ERROR!

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

003632  104002                                  ERROR   2               ;/ERROR - CLOCK A STATUS REGISTER.
                                                                        ;/BIT 1 FAILED TO CLEAR.

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

003634                                  2$:
351
```

```
(1)                                             ;/#
(5)                                             ;;**********************************************************************
(4)                                             ;*TEST 1%        *TEST THAT CLOCK A STATUS REGISTER BIT 0 CAN BE SET AND CLEARED
(5)                                             ;*
(5)                                             ;*CLOCK STATUS REGISTER BIT EXERCISE. ON FAILURE-SUSPECT INDIVIDUAL
(5)                                             ;*F/FS OR GATES
(5)                                             ;*
(5)
(4)                                             ;;**********************************************************************
(3)  003634  000004                             TST14:  SCOPE
(2)  003636  012737  000100  001160                    MOV     #100,$TIMES     ;;DO 100 ITERATIONS
(1)
(1)  003644  005077  175526                             CLR     @ASR            ;/CLEAR THE STATUS REGISTER.
(1)  003650  052777  000001  175520                     BIS     #BIT0,@ASR      ;/SET BIT 0.
(1)  003656  012737  000001  001124                     MOV     #BIT0,$GDDAT    ;/SET FOR ERROR TYPEOUT S/B.
(1)  003664  017737  175506  001126                     MOV     @ASR,$BDDAT     ;/READ THE STATUS REGISTER.
(1)  003672  023737  001124  001126                     CMP     $GDDAT,$BDDAT   ;/DID BIT 0 AND ONLY BIT 0 SET?
(1)  003700  001402                                     BEQ     1$              ;/IF SO-LETS TRY CLEARING IT.
(2)
                                                ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  003702  104002                                     ERROR   2               ;/ERROR CLOCK AS STATUS REGISTER
(1)                                                                             ;/BIT 0 FAILED TO BIT SET.
(2)
                                                ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  003704  000412                                     BR      2$              ;/BR TO END SUBTEST.
(1)
(1)  003706  042777  000001  175462  1$:       BIC     #BIT0,@ASR      ;/TRY CLEARING BIT 0.
(1)  003714  005037  001124                     CLR     $GDDAT          ;/CLEAR S/B FOR TYPEOUT IF ANY.
(1)  003720  017737  175452  001126             MOV     @ASR,$BDDAT     ;/NOW READ IT BACK.
(1)  003726  001401                             BEQ     2$              ;/IF ZERO - NO ERROR!
(1)
(2)
                                                ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  003730  104002                                     ERROR   2               ;/ERROR - CLOCK A STATUS REGISTER.
(1)                                                                             ;/BIT 0 FAILED TO CLEAR.
(1)
(2)
                                                ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  003732                                     2$:
(1)
352             000010                                  .RADIX  8
383
```

```
  385
  (5)                                   ;;*********************************************************************
  (4)                                   ;*TEST 15        *TEST THAT PATERN 125252 WILL SET AND CLEAR IN BUFFER REG.
  (4)                                   ;;*********************************************************************
  (3)  003732  000004                   TST15:  SCOPE
  (1)
  (1)  003734  005077  175440                   CLR     @ABR            ;/CLEAR THE BUFFER REG.
  (1)  003740  012737  125252  001124           MOV     #125252,$GDDAT  ;/RECORD PATTERN: 125252 .
  (1)  003746  013777  001124  175424           MOV     $GDDAT,@ABR     ;/SET PATTERN IN BUFFER REG.
  (1)  003754  017737  175420  001126           MOV     @ABR,$BDDAT     ;/READ THE BUFFER REG.
  (1)
  (1)  003762  023737  001124  001126           CMP     $GDDAT,$BDDAT   ;/DID THE PATTERN SET OK?
  (1)  003770  001402                           BEQ     1$              ;/YES-TRY CLEARING IT.
  (1)
  (2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

  (1)  003772  104003                           ERROR   3               ;/ERROR PATTERN 125252 FAILED TO
  (1)                                                                   ;/SET PROPERLY IN BUFFER REG.
  (2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

  (1)  003774  000412                           BR      2$              ;/GOTO SCOPE LOOP.
  (1)
  (1)  003776  042777  125252  175374  1$:     BIC     #125252,@ABR    ;/TRY CLEARING PATTERN.
  (1)  004004  005037  001124                   CLR     $GDDAT          ;/EXPECT ZERO BACK.
  (1)  004010  017737  175364  001126           MOV     @ABR,$BDDAT     ;/READ BUFFER REG.,WAS IT ZERO?
  (1)  004016  001401                           BEQ     2$              ;/YES-NEXT TEST.
  (1)
  (2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

  (1)  004020  104003                           ERROR   3               ;/BUFFER REG. COULD NOT BE LOADED
  (1)                                                                   ;/TO A ZERO.
  (2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

  (1)  004022                           2$:
  (1)
```

```
387
(5)                                     ;;*********************************************************************
(4)                                     ;*TEST 16        *TEST THAT PATERN 052525 WILL SET AND CLEAR IN BUFFER REG.
(4)                                     ;;*********************************************************************
(3)  004022  000004                     TST16:  SCOPE
(1)
(1)  004024  005077  175350                     CLR     @ABR            ;/CLEAR THE BUFFER REG.
(1)  004030  012737  052525  001124             MOV     #052525,$GDDAT  ;/RECORD PATTERN: 052525 .
(1)  004036  013777  001124  175334             MOV     $GDDAT,@ABR     ;/SET PATTERN IN BUFFER REG.
(1)  004044  017737  175330  001126             MOV     @ABR,$BDDAT     ;/READ THE BUFFER REG.
(1)
(1)  004052  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;/DID THE PATTERN SET OK?
(1)  004060  001402                             BEQ     1$              ;/YES-TRY CLEARING IT.
(1)
(2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  004062  104003                             ERROR   3               ;/ERROR PATTERN 052525 FAILED TO
(1)                                             ;/SET PROPERLY IN BUFFER REG.
(2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  004064  000412                             BR      2$              ;/GOTO SCOPE LOOP.
(1)
(1)  004066  042777  052525  175304  1$:       BIC     #052525,@ABR    ;/TRY CLEARING PATTERN.
(1)  004074  005037  001124                     CLR     $GDDAT          ;/EXPECT ZERO BACK.
(1)  004100  017737  175274  001126             MOV     @ABR,$BDDAT     ;/READ BUFFER REG.,WAS IT ZERO?
(1)  004106  001401                             BEQ     2$              ;/YES-NEXT TEST.
(1)
(2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  004110  104003                             ERROR   3               ;/BUFFER REG. COULD NOT BE LOADED
(1)                                             ;/TO A ZERO.
(2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  004112                             2$:
(1)
388
389
390                                             .SBTTL  *
391                                             .SBTTL  * PHASE 2 ADVANCED BASIC LOGIC TESTS
392                                             .SBTTL  *
393
402
```

```
                                ;;*****************************************************************
                                ;*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.
                                ;*
                                ;;*****************************************************************
004112  000004                TST17:  SCOPE
004114  012737  000050  001160        MOV     #50,$TIMES      ;;DO 50 ITERATIONS

004122  005077  175250              CLR     @ASR            ;MAKE SURE THE STATUS REGISTER IS CLEAR.
004126  112777  127677  175242      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.

004134  017777  175236  174764      MOV     @ASR,@$BDDAT    ;NOW EXAMINE THE
                                                            ;STATUS REGISTER.
004142  013737  001126  001124      MOV     $BDDAT,$GDDAT   ;FIX $GDDAT FOR ERROR TYPEOUT IF
004150  105037  001125              CLRB    $GDDAT+1        ;ANY RROR HAS OCCURRED, UPPER BYTE CLEARED.

004154  105737  001127              TSTB    $BDDAT+1        ;ARE ANY BITS IN THE UPPER BYTE
                                                            ;OF THE STATUS REGISTER SET?
004160  001401                      BEQ     1$              ;BRANCH NEXT TEST IF UPPER BYTE=0.


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

004162  104001                      ERROR   1               ;ERROR - WROTE INTO UPPER BYTE OF
                                                            ;CLOCK'S STATUS WHEN
                                                            ;DOING A DATOB TO THE LOW BYTE.


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

004164                          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.
                                ;*
                                ;;*****************************************************************
004164  000004                  TST20:  SCOPE
004166  012737  000050  001160          MOV     #50,$TIMES      ;;DO 50 ITERATIONS

004174  005077  175176          CLR     @ASR            ;CLEAR THE STATUS REGISTER.

004200  005237  001376          INC     ASR             ;ADD #1 TO THE STATUS REGISTER'S ADDRESS
                                                        ;SO THAT WE WILL BE WRITING INTO
                                                        ;THE HIGH BYTE.

004204  112777  177313  175164  MOVB    #177313,@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.

004212  005337  001376          DEC     ASR             ;FIX ADDRESS OF THE STATUS REGISTER ADDR.
                                                        ;SO WE CAN LOOK AT THE WHOLE WORD.
004216  017737  175154  001126  MOV     @ASR,$BDDAT     ;READ BACK WHAT THE STATUS REG. CONTAINS
004224  013737  001126  001124  MOV     $BDDAT,$GDDAT   ;FIX $GDDAT FOR ERROR TYPEOUT IF AN ERROR
004232  105037  001124          CLRB    $GDDAT          ;OCCURRED, LOWER BYTE CLEARED.
004236  105737  001126          TSTB    $BDDAT          ;IS LOWER BYTE CLEAR?
004242  001401                  BEQ     1$              ;BR IF YES TO NEXT SUBTEST.


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

004244  104001                          ERROR   1               ;ERROR - WROTE INTO LOWER BYTE
                                                                ;OF CLOCKS STATUS REGISTER WHEN
                                                                ;DOING A DATOB TO THE HIGH BYTE.

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

004246                          1$:
```

```
                        ;;*********************************************************************
                        ;*TEST 21       *TEST CLOCK'S COUNT REGISTER WITH 125252 PATTERN
                        ;;*********************************************************************
004246  000004          TST21:  SCOPE
004250  012737  000100  001160          MOV     #100,$TIMES     ;;DO 100 ITERATIONS

004256  005077  175114          CLR     @ASR            ;SELECT MODE 0.
004262  012777  125252  175110          MOV     #125252,@ABR    ;LOAD THE BUFFER REGISTER WITH
                                                        ;PATTERN 125252. IT WILL BE
                                                        ;TRANSFERRED TO THE COUNT REGISTER
                                                        ;SINCE THIS IS MODE 0.
004270  052777  000001  175100          BIS     #BIT0,@ASR      ;SET GO BIT(ALLOWS BUFFER-COUNT REG XFER).

004276  012737  125252  001124          MOV     #125252,$GDDAT  ;SET EXPECTED TO PATTERN IN CASE OF
                                                        ;NEED OF ERROR TYPEOUT.
004304  017746  175066          MOV     @ASR,-(6)       ;/SAVE CSR
004310  011637  001424          MOV     (6),$TMP3       ;/GET CSR.
004314  042737  177707  001424          BIC     #177707,$TMP3   ;/SAVE RATE BITS.
004322  052737  004005  001424          BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
004330  013777  001424  175040          MOV     $TMP3,@ASR              ;/LOAD CSR.
                                                        ;/THIS MUST BE DONE IN
                                                        ;/ORDER TO XFERR COUNTER
                                                        ;/TO BUFFER ON ST2.
004336  052777  001000  175032          BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
004344  017737  175030  001126          MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
                                                        ;/PREVIOUS COUNTER
004352  012677  175020          MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
004356  005737  001126          TST     $BDDAT          ;/RESTORE CSR

004362  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;DID ALL THE BITS AND NO OTHER BITS
                                                        ;COME THROUGH?
004370  001401          BEQ     1$              ;BR IF YES TO NEXT TEST.


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

004372  104005          ERROR   5               ;DATA ERROR CLOCK - PATTERN "125252"
                                                ;FAILED TO TRANSFER PROPERLY BETWEEN
                                                ;BUFFER AND COUNT REGISTERS.


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

004374          1$:
```

```
                                ;;*****************************************************************
                                ;*TEST 22        *TEST CLOCKS COUNTER REGISTER WITH 052525 PATTERN
                                ;;*****************************************************************
004374  000004                  TST22:  SCOPE
004376  012737  000050  001160          MOV     #50,$TIMES      ;;DO 50 ITERATIONS

004404  005077  174766          CLR     @ASR            ;SELECT MODE 0.
004410  012777  052525  174762  MOV     #052525,@ABR    ;LOAD THE BUFFER REGISTER WITH
                                                        ;PATTERN 052525. IT WILL BE
                                                        ;TRANSFERRED TO THE COUNT REGISTER
                                                        ;SINCE THIS IS MODE 0.
004416  052777  000001  174752  BIS     #BIT0,@ASR      ;SET BO BIT(ALLOWS BUFFER-COUNT REG XFER).

004424  012737  052525  001124  MOV     #052525,$GDDAT  ;SET EXPECTED TO PATTERN IN CASE OF
                                                        ;NEED OF ERROR TYPEOUT.
004432  017746  174740          MOV     @ASR,-(6)       ;/SAVE CSR
004436  011637  001424          MOV     (6),$TMP3       ;/GET CSR.
004442  042737  177707  001424  BIC     #177707,$TMP3   ;/SAVE RATE BITS.
004450  052737  004005  001424  BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
004456  013777  001424  174712  MOV     $TMP3,@ASR              ;/LOAD CSR.
                                                        ;/THIS MUST BE DONE IN
                                                        ;/ORDER TO XFERR COUNTER
                                                        ;/TO BUFFER ON ST2.
004464  052777  001000  174704  BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
004472  017737  174702  001126  MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
                                                        ;/PREVIOUS COUNTER
004500  012677  174672          MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
004504  005737  001126          TST     $BDDAT          ;/RESTORE CSR

004510  023737  001124  001126  CMP     $GDDAT,$BDDAT   ;DID ALL THE BITS AND NO OTHER BITS
                                                        ;COME THROUGH?
004516  001401                  BEQ     1$              ;BR IF YES TO NEXT TEST.


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

004520  104005                          ERROR   5               ;DATA ERROR CLOCK - PATTERN ''052525''
                                                                ;FAILED TO TRANSFER PROPERLY BETWEEN
                                                                ;BUFFER AND COUNT REGISTERS.


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

004522                          1$:
```

```
519                                     ;;*********************************************************************
(3)                                     ;*TEST 23       *TEST THAT INIT CLEARS STATUS REGISTER
(4)                                     ;*
(4)                                     ;*TESTING OF THE INIT LOGIC AS RECEIVED FROM THE QBUS AND BUFFERED
(4)                                     ;*TO STATUS REGISTER F/FS.
(4)                                     ;*
(3)                                     ;;*********************************************************************
(2) 004522  000004                      TST23:  SCOPE
(1) 004524  012737  000005  001160              MOV     #5,$TIMES       ;;DO 5 ITERATIONS
520
521 004532  005037  001124              CLR     $GDDAT          ;EXPECTED DATA IS ZERO.
522 004536  012777  177777  174632      MOV     #177777,@ASR    ;SET ALL BITS IN THE STATUS REG.
523
524 004544  000005                      RESET                   ;SYSTEM INITIALIZE.
525
526 004546  017737  174624  001126      MOV     @ASR,$BDDAT     ;READ THE STATUS REG., ALL BITS SHOULD
527                                                             ;HAVE BEEN CLEARED BY INIT.
528 004554  001402                      BEQ     1$              ;BR IF YES TO NEXT TEST.
529
530
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

531
532 004556  104002                      ERROR   2               ;ERROR - SYSTEM INIT FAILED TO CLEAR
533                                                             ;STATUS REGISTER CLOCK A.
534
535
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

536 004560  000413                      BR      TST24           ;;
537 004562                          1$:
538 004562  005737  001440              TST     EXS             ;TEST EXTERNAL SIGS?
539 004566  001410                      BEQ     TST24           ;;
540 004570  052777  016000  174620      BIS     #BIT11!BIT12!BIT10,@DR2 ;ENABLE ST1,ST2 TO LATCH.
541 004576  032777  000006  174610      BIT     #6,@DR          ;ST1,ST2, OVERFLOW SET?
542 004604  001401                      BEQ     TST24           ;;
543
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

544 004606  104006                      ERROR   6               ;INIT FAILED TO CLEAR
545                                                             ;ST1,ST2, AND/OR OVERFLOW
546
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

554
```

```
                                ;;*****************************************************************
                                ;*TEST 24        *TEST THAT INIT CLEARS BUFFER REGISTER
                                ;*
                                ;*THIS TEST IS DESIGNED TO SEE IF ''INIT H''
                                ;*CLEARS THE BUFFER REGISTER. WE ALREADY
                                ;*KNOW IT CLEARS THE STATUS REG.
                                ;*
                                ;;*****************************************************************
004610  000004                  TST24:  SCOPE
004612  012737  000005  001160          MOV     #5,$TIMES       ;;DO 5 ITERATIONS

004620  005037  001124          CLR     $GDDAT          ;CLEAR EXPECTED DATA.
004624  012777  177777  174546  MOV     #177777,@ABR    ;SET ALL BITS IN THE BUFFER REGISTER.

004632  000005                  RESET                   ;ISSUE SYSTEM INITIALIZE.

004634  017737  174540  001126  MOV     @ABR,$BDDAT     ;READ THE BUFFER REGISTER, ALL BITS
                                                        ;SHOULD HAVE BEEN CLEARED BY INIT.
004642  001401                  BEQ     1$              ;BR IF YES TO NEXT SUBTEST.


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

004644  104003                  ERROR   3               ;ERROR - SYSTEM INIT FAILED
                                                        ;TO CLEAR BUFFER REGISTER A.


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

004646                          1$:
004646  005737  001440                  TST     EXS             ;TEST EXTERNAL SIGS?
004652  001403                          BEQ     TST25           ;;
004654  052777  016000  174534          BIS     #BIT11!BIT12!BIT10,@DR2 ;ENABLE THEM
```

```
                                ;;*****************************************************************
                                ;*TEST 25       *TEST THE SETTING OF MAINTENANCE ST2 IN CLOCK BIT 15 TO SET
                                ;;*****************************************************************
(2) 004662  000004              TST25:  SCOPE

004664  012777  000001  174504          MOV     #1,@ASR         ;SET THE GO BIT(ENABLES ST2 TO SET FLG).
004672  052777  001000  174476          BIS     #BIT9,@ASR      ;SET MAINTENANCE ST2.

004700  005777  174472          TST     @ASR            ;DID BIT15 (ST2 FLAG) SET?
004704  100402                  BMI     1$              ;BR IF YES - NEXT TEST


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

004706  104001                  ERROR   1               ;ERROR - MAINTENANCE ST2 (BIT9)
                                                        ;DID NOT SET BIT15 (ST2 FLAG).


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

004710  000410                  BR      TST26           ;;
004712  005737  001440  1$:     TST     EXS             ;TEST EXTERNAL SIGNALS?
004716  001405                  BEQ     TST26           ;;
004720  032777  000004  174466  BIT     #BIT2,@DR       ;DID EXTERNAL ST2 GET SET?
004726  001001                  BNE     TST26           ;;


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

004730  104006                  ERROR   6               ;ST2 OUT NOT DETECTED
                                                        ;BY TESTOR


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

```
                                ;;*****************************************************************
                                ;*TEST 26        *TEST THAT BIT00 IN CLOCK STATUS REG. WILL SET WHEN BIT13 AND MAIN. ST2
                                ;;*****************************************************************
004732  000004                  TST26:  SCOPE

004734  012777  020000  174434          MOV     #BIT13,@ASR     ;SET ''ST2 ENB COUNTER'' IN CLK  STATUS REG.
004742  052777  001000  174426          BIS     #BIT9,@ASR      ;GENERATE A MAINTENANCE ST2.
004750  032777  000001  174420          BIT     #BIT00,@ASR     ;DID BIT00 (GO) SET?
004756  001001                          BNE     1$              ;BR IF YES - NEXT TEST.


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

004760  104001                          ERROR   1               ;ERROR - BIT00 OF CLOCK'S STATUS REGISTER
                                                                ;FAILED TO SET WHEN BIT13 WAS SET
                                                                ;AND A MAINTENANCE ST2 GENERATED.


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

004762  005077  174410          1$:     CLR     @ASR            ;LEAVE SUBTEST WITH CLOCK CLEAR.

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

```
623                                     ;;*****************************************************************
(3)                                     ;*TEST 27       *TEST TO SEE IF THE COUNTER WILL INCREMENT
(3)                                     ;;*****************************************************************
(2)  004766  000004                     TST27:  SCOPE
624
625  004770  005077  174402                    CLR     @ASR            ;CLEAR THE CSR.
626  004774  005077  174400                    CLR     @ABR            ;CLEAR THE BUFFER
627  005000  052777  000061  174370            BIS     #BIT5!BIT4!BIT0,@ASR ;SET RATE:ST1, GO.
628  005006  052777  000400  174362            BIS     #BIT8,@ASR      ;GENERATE A MAINTENANCE ST1.
629                                                                     ;DID THE CLOCK COUNT?
630  005014  017746  174356                    MOV     @ASR,-(6)       ;/SAVE CSR
(1)  005020  011637  001424                    MOV     (6),$TMP3       ;/GET CSR.
(1)  005024  042737  177707  001424            BIC     #177707,$TMP3   ;/SAVE RATE BITS.
(1)  005032  052737  004005  001424            BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
(1)  005040  013777  001424  174330            MOV     $TMP3,@ASR              ;/LOAD CSR.
(1)                                                                     ;/THIS MUST BE DONE IN
(1)                                                                     ;/ORDER TO XFERR COUNTER
(1)                                                                     ;/TO BUFFER ON ST2.
(1)  005046  052777  001000  174322            BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
(1)  005054  017737  174320  001126            MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
(1)                                                                     ;/PREVIOUS COUNTER
(1)  005062  012677  174310                    MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
(1)  005066  005737  001126                    TST     $BDDAT          ;/RESTORE CSR
631  005072  001001                           BNE     1$              ;YES, NEXT TEST.
632
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

633  005074  104006                           ERROR   6               ;CLOCK FAILED TO INCREMENT.
634
635
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

636  005076                             1$:
637
```

```
                                ;;*********************************************************************
                                ;*TEST 30        *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.
                                ;*
                                ;;*********************************************************************
(2) 005076  000004              TST30:  SCOPE
(1) 005100  012737  000010  001160      MOV     #10,$TIMES      ;;DO 10 ITERATIONS

005106  005077  174264          CLR     @ASR            ;CLEAR THE CSR.
005112  005077  174262          CLR     @ABR            ;CLEAR THE BUFFER REG
005116  012737  000000  001124  MOV     #0,$GDDAT       ;CLEAR EXPECTED.
005124  012737  005266  001110  MOV     #2$,$LPERR

005132  052777  000061  174236  1$:     BIS     #BIT5!BIT4!BIT0,@ASR ;START CLOCK, RATE:ST1.

005140  052777  000400  174230  BIS     #BIT8,@ASR      ;GENERATE A MAINTENANCE ST1
                                                        ;CLOCK SHOULD COUNT ONCE.
005146  005737  001440          TST     EXS             ;TEST EXTERNAL SIGNALS?
005152  001406                  BEQ     10$             ;NO
005154  032777  000002  174232  BIT     #BIT1,@DR       ;YES - DID ST1 GET SET?
005162  001002                  BNE     10$             ;YES

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

005164  104006                  ERROR   6               ;ST1 OUT NOT DETECTED
                                                        ;BY TESTOR

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

005166  000447                  BR      TST31           ;;
005170                  10$:
005170  017746  174202          MOV     @ASR,-(6)       ;/SAVE CSR
(1) 005174  011637  001424      MOV     (6),$TMP3       ;/GET CSR.
(1) 005200  042737  177707  001424  BIC     #177707,$TMP3   ;/SAVE RATE BITS.
(1) 005206  052737  004005  001424  BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
(1) 005214  013777  001424  174154  MOV     $TMP3,@ASR              ;/LOAD CSR.
(1)                                                     ;/THIS MUST BE DONE IN
(1)                                                     ;/ORDER TO XFERR COUNTER
(1)                                                     ;/TO BUFFER ON ST2.
(1) 005222  052777  001000  174146  BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
(1) 005230  017737  174144  001126  MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
(1)                                                     ;/PREVIOUS COUNTER
(1) 005236  012677  174134          MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
(1) 005242  005737  001126          TST     $BDDAT          ;/RESTORE CSR
005246  005237  001124          INC     $GDDAT          ;COUNT=OLD COUNT+1
005252  023737  001124  001126  CMP     $GDDAT,$BDDAT   ;COUNT READ=COUNT EXP'ED?
005260  001402                  BEQ     2$              ;YES - SEE IF WE'RE THROUGH.

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

005262  104011                  ERROR   11              ;CLOCK FAILED TO COUNT UP PROPERLY.
```

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

675  005264  000410                            BR      3$              ;GOTO SCOPE LOOP.
676
677  005266  005077  174104            2$:     CLR     @ASR
678  005272  013777  001124  174100            MOV     $GDDAT,@ABR
679  005300  005737  001124                    TST     $GDDAT          ;ALL DONE?
680  005304  001312                            BNE     1$              ;NO DO NEXT INCREMENT.
681  005306                            3$:
682
683                                     ;;**********************************************************************
(3)                                     ;*TEST 31        *TEST THAT OVERFLOW (CSR BIT07) WILL SET ON OVERFLOW
(3)                                     ;;**********************************************************************
(2)  005306  000004                     TST31:  SCOPE
684
685  005310  005737  001440                    TST     EXS             ;TESTOR MODE ENABLED??
686  005314  001411                            BEQ     2$              ;NO-THEN SKIP NEXT SECTION OF CODE.
687
688  005316  005037  170500                    CLR     @#170500        ;CLEAR TESTOR A/D
689  005322  005737  170502                    TST     @#170502        ;DUMB READ OF A/D BUFFER.
690  005326  052737  000040  170500            BIS     #BIT5,@#170500  ;ENABLE EXTRENAL START OF A/D.
691  005334  012700  000010                    MOV     #8.,R0          ;SET TIME OUT NUMBER.
692  005340                            2$:
693  005340  005077  174032                    CLR     @ASR            ;CLEAR THE CSR
694  005344  012777  177777  174026            MOV     #-1,@ABR        ;SET PRESET BUFFER TO ALL ONES.
695
696  005352  052777  000061  174016            BIS     #BIT5!BIT4!BIT0,@ASR ;START CLOCK, RATE ST1.
697
698  005360  052777  000400  174010            BIS     #BIT8,@ASR      ;COUNT CLOCK ONCE, OVERFLOW
699                                                                     ;SHOULD OCCUR.
700  005366  105777  174004                    TSTB    @ASR            ;DID OVERFLOW SET?
701  005372  100402                            BMI     1$              ;YES - THEN NEXT TEST
702
703
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

704  005374  104006                            ERROR   6               ;ERROR - OVERFLOW, CSR BIT0?
705                                                                     ;FAILED TO SET ON OVERFLOW
706  005376  000411                            BR      TST32           ;;
707  005400  005737  001440            1$:     TST     EXS             ;TEST EXTERNAL SIGNALS?
708  005404  001406                            BEQ     TST32           ;;
709  005406  105737  170500                    TSTB    @#170500        ;IF  OVERFLOW GOT OUT,IT GAVE A/D START,
710                                                                     ;WE'RE LOOKING FOR A/D DONE-DID IT GET SET?
711  005412  100403                            BMI     TST32           ;;
712  005414  005300                            DEC     R0              ;DID WE ALLOW ENOUGH TIME??
713  005416  001370                            BNE     1$              ;NO-THEN WAIT.
714
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

715  005420  104006                            ERROR   6               ;OVERFLOW OUT NOT DETECTED
716                                                                     ;BY TESTOR
717
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$


718
719
```

```
                                ;;*****************************************************************
                                ;*TEST 32       *TEST THAT OVERFLOW WILL CLEAR THE GO BIT
                                ;;*****************************************************************
005422  000004                  TST32:  SCOPE

005424  005077  173746                  CLR     @ASR            ;CLEAR THE CSR.

005430  012777  177777  173742          MOV     #-1,@ABR        ;PRESET CLOCK TO -1.

005436  052777  000061  173732          BIS     #BIT5!BIT4!BIT0,@ASR ;START CLOCK, RATE:ST1

005444  052777  000400  173724          BIS     #BIT8,@ASR      ;COUNT ONCE, OVERFLOW
                                                                ;SHOULD OCCUR CLEARING
                                                                ;ENABLE (CSR BIT00)

005452  032777  000001  173716          BIT     #BIT0,@ASR      ;DID THE ENABLE CLEAR?
005460  001401                          BEQ     1$              ;YES - NEXT TEST.


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

005462  104006                          ERROR   6               ;ERROR - OVERFLOW FAILED
                                                                ;TO CLEAR ENABLE (CSR BIT00)

005464                          1$:

                                ;;*****************************************************************
                                ;*TEST 33       *TEST THAT GO BIT DOES NOT CLEAR ON OVERFLOW, IF MODE 1
                                ;;*****************************************************************
005464  000004                  TST33:  SCOPE

005466  005077  173704                  CLR     @ASR            ;CLEAR THE CSR.
005472  012777  177777  173700          MOV     #-1,@ABR        ;PRESET BUFFER=ONE COUNT FROM OVERFLOW.
005500  052777  000063  173670          BIS     #63,@ASR        ;MODE 1, RATE:ST1, GO.

005506  052777  000400  173662          BIS     #BIT8,@ASR      ;GENERATE MAINTENANCE ST1.

005514  032777  000001  173654          BIT     #BIT0,@ASR      ;DID ENABLE (GO BIT) CLEAR?
005522  001001                          BNE     1$              ;NO (GOOD) NEXT TEST.


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

005524  104006                          ERROR   6               ;GO BIT CLEARED ON OVERFLOW
                                                                ;WHEN MODE 1 WAS SELECTED


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


005526  005077  173644          1$:     CLR     @ASR            ;CLEAR THE CLOCK.
```

```
806
(5)                                     ;;*********************************************************************
(4)                                     ;*TEST 34        *TEST THE ABILITY OF CLOCK TO COUNT AT 1MHZ RATE
(5)                                     ;*
(5)                                     ;*THIS TEST IS DESIGNED TO TEST THE CLOCK'S ABILITY
(5)                                     ;*TO COUNT AT 1MHZ RATE.
(5)                                     ;*
(4)                                     ;;*********************************************************************
(3)  005532  000004                     TST34:  SCOPE
(2)  005534  012737  000005  001160             MOV     #5,$TIMES       ;;DO 5 ITERATIONS
(1)
(1)
(1)  005542  005077  173630             CLR     @ASR                    ;/CLEAR CLOCK
(1)  005546  005077  173626             CLR     @ABR                    ;/CLEAR PRESET BUFFER
(1)  005552  012777  000011  173616     MOV     #BIT0!10,@ASR           ;/START CLOCK, MODE0, RATE:1MHZ
(1)  005560  005000                     CLR     R0                      ;/NOW WE'LL DO A LITTLE DELAY. THIS DELAY
(1)
(1)  005562  005200                     1$:     INC     R0              ;/WILL AMOUNT TO APPROXIMATELY
(1)  005564  001376                     BNE     1$                      ;/369 MS.
(1)
(2)  005566  017746  173604             MOV     @ASR,-(6)               ;/SAVE CSR
(2)  005572  011637  001424             MOV     (6),$TMP3               ;/GET CSR.
(2)  005576  042737  177707  001424     BIC     #177707,$TMP3           ;/SAVE RATE BITS.
(2)  005604  052737  004005  001424     BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
(2)  005612  013777  001424  173556     MOV     $TMP3,@ASR                      ;/LOAD CSR.
(2)                                                                     ;/THIS MUST BE DONE IN
(2)                                                                     ;/ORDER TO XFERR COUNTER
(2)                                                                     ;/TO BUFFER ON ST2.
(2)  005620  052777  001000  173550     BIS     #BIT9,@ASR              ;/GENERATE ON ST2 PULSE
(2)  005626  017737  173546  001126     MOV     @ABR,$BDDAT             ;/READ THE PRESET BUFFER,
(2)                                                                     ;/PREVIOUS COUNTER
(2)  005634  012677  173536             MOV     (6)+,@ASR               ;/CONTENTS ARE IN $BDDAT.
(2)  005640  005737  001126             TST     $BDDAT                  ;/RESTORE CSR
(1)  005644  001004                     BNE     2$                      ;/YES - NEXT TEST.
(1)  005646  105766  177776             TSTB    -2(6)                   ;/AT HIGH RATE MAY HAVE HAD OVERFLOW
(1)                                                                     ;/NOTE: CSR HAD BEEN PUT ON STACK.
(1)  005652  100401                     BMI     2$                      ;/NEXT TEST IF OVERFLOW.
(1)
(2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  005654  104006                             ERROR   6               ;/CLOCK FAILED TO COUNT AT
(1)                                                                     ;/RATE:1MHZ
(2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)
(1)  005656  005077  173514             2$:     CLR     @ASR            ;/CLEAR THE CLOCK.
(1)
(1)
```

```
808
(5)                                     ;;*********************************************************************
(4)                                     ;*TEST 35        *TEST THE ABILITY OF CLOCK TO COUNT AT 100KHZ RATE
(5)
(5)                                     ;*
(5)                                     ;*THIS TEST IS DESIGNED TO TEST THE CLOCK'S ABILITY
(5)                                     ;*TO COUNT AT 100KHZ RATE.
(5)                                     ;*
(4)                                     ;*********************************************************************
(3)  005662  000004                     TST35:  SCOPE
(2)  005664  012737  000005  001160             MOV     #5,$TIMES       ;;DO 5 ITERATIONS
(1)
(1)
(1)  005672  005077  173500              CLR     @ASR                    ;/CLEAR CLOCK
(1)  005676  005077  173476              CLR     @ABR                    ;/CLEAR PRESET BUFFER
(1)  005702  012777  000021  173466      MOV     #BIT0!20,@ASR           ;/START CLOCK, MODE0, RATE:100KHZ
(1)  005710  005000                      CLR     R0                      ;/NOW WE'LL DO A LITTLE DELAY. THIS DELAY
(1)
(1)  005712  005200                     1$:     INC     R0              ;/WILL AMOUNT TO APPROXIMATELY
(1)  005714  001376                      BNE     1$                      ;/369 MS.
(1)
(2)  005716  017746  173454              MOV     @ASR,-(6)               ;/SAVE CSR
(2)  005722  011637  001424              MOV     (6),$TMP3               ;/GET CSR.
(2)  005726  042737  177707  001424      BIC     #177707,$TMP3           ;/SAVE RATE BITS.
(2)  005734  052737  004005  001424      BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
(2)  005742  013777  001424  173426      MOV     $TMP3,@ASR                      ;/LOAD CSR.
(2)                                                                     ;/THIS MUST BE DONE IN
(2)                                                                     ;/ORDER TO XFERR COUNTER
(2)                                                                     ;/TO BUFFER ON ST2.
(2)  005750  052777  001000  173420              BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
(2)  005756  017737  173416  001126              MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
(2)                                                                     ;/PREVIOUS COUNTER
(2)  005764  012677  173406                      MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
(2)  005770  005737  001126                      TST     $BDDAT          ;/RESTORE CSR
(1)  005774  001004                              BNE     2$              ;/YES - NEXT TEST.
(1)  005776  105766  177776                      TSTB    -2(6)           ;/AT HIGH RATE MAY HAVE HAD OVERFLOW
(1)                                                                     ;/NOTE: CSR HAD BEEN PUT ON STACK.
(1)  006002  100401                              BMI     2$              ;/NEXT TEST IF OVERFLOW.
(1)
(2)

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

(1)  006004  104006                              ERROR   6               ;/CLOCK FAILED TO COUNT AT
(1)                                                                     ;/RATE:100KHZ
(2)

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

(1)
(1)  006006  005077  173364             2$:     CLR     @ASR            ;/CLEAR THE CLOCK.
(1)
(1)
```

```
                                ;;*****************************************************************
                                ;*TEST 36        *TEST THE ABILITY OF CLOCK TO COUNT AT 10KHZ RATE
                                ;*
                                ;*THIS TEST IS DESIGNED TO TEST THE CLOCK'S ABILITY
                                ;*TO COUNT AT 10KHZ RATE.
                                ;*
                                ;*****************************************************************
006012  000004                  TST36:  SCOPE
006014  012737  000005  001160          MOV     #5,$TIMES       ;;DO 5 ITERATIONS

006022  005077  173350          CLR     @ASR            ;/CLEAR CLOCK
006026  005077  173346          CLR     @ABR            ;/CLEAR PRESET BUFFER
006032  012777  000031  173336  MOV     #BIT0!30,@ASR   ;/START CLOCK, MODE0, RATE:10KHZ
006040  005000                  CLR     R0              ;/NOW WE'LL DO A LITTLE DELAY. THIS DELAY

006042  005200                  1$:     INC     R0              ;/WILL AMOUNT TO APPROXIMATELY
006044  001376                  BNE     1$              ;/369 MS.

006046  017746  173324          MOV     @ASR,-(6)       ;/SAVE CSR
006052  011637  001424          MOV     (6),$TMP3       ;/GET CSR.
006056  042737  177707  001424  BIC     #177707,$TMP3   ;/SAVE RATE BITS.
006064  052737  004005  001424  BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
006072  013777  001424  173276  MOV     $TMP3,@ASR              ;/LOAD CSR.
                                                        ;/THIS MUST BE DONE IN
                                                        ;/ORDER TO XFERR COUNTER
                                                        ;/TO BUFFER ON ST2.
006100  052777  001000  173270  BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
006106  017737  173266  001126  MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
                                                        ;/PREVIOUS COUNTER
006114  012677  173256          MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
006120  005737  001126          TST     $BDDAT          ;/RESTORE CSR
006124  001004                  BNE     2$              ;/YES - NEXT TEST.
006126  105766  177776          TSTB    -2(6)           ;/AT HIGH RATE MAY HAVE HAD OVERFLOW
                                                        ;/NOTE: CSR HAD BEEN PUT ON STACK.
006132  100401                  BMI     2$              ;/NEXT TEST IF OVERFLOW.

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

006134  104006                          ERROR   6               ;/CLOCK FAILED TO COUNT AT
                                                                ;/RATE:10KHZ

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


006136  005077  173234          2$:     CLR     @ASR            ;/CLEAR THE CLOCK.
```

```
812
(5)                                     ;;*********************************************************************
(4)                                     ;*TEST 37        *TEST THE ABILITY OF CLOCK TO COUNT AT 1KHZ RATE
(5)
(5)                                     ;*
(5)                                     ;*THIS TEST IS DESIGNED TO TEST THE CLOCK'S ABILITY
(5)                                     ;*TO COUNT AT 1KHZ RATE.
(5)                                     ;*
(4)                                     ;;*********************************************************************
(3)  006142  000004                     TST37:  SCOPE
(2)  006144  012737  000005  001160             MOV     #5,$TIMES       ;;DO 5 ITERATIONS
(1)
(1)
(1)  006152  005077  173220             CLR     @ASR            ;/CLEAR CLOCK
(1)  006156  005077  173216             CLR     @ABR            ;/CLEAR PRESET BUFFER
(1)  006162  012777  000041  173206     MOV     #BIT0!40,@ASR   ;/START CLOCK, MODE0, RATE:1KHZ
(1)  006170  005000                     CLR     R0              ;/NOW WE'LL DO A LITTLE DELAY. THIS DELAY
(1)
(1)  006172  005200                     1$:     INC     R0              ;/WILL AMOUNT TO APPROXIMATELY
(1)  006174  001376                     BNE     1$              ;/369 MS.
(1)
(2)  006176  017746  173174             MOV     @ASR,-(6)       ;/SAVE CSR
(2)  006202  011637  001424             MOV     (6),$TMP3       ;/GET CSR.
(2)  006206  042737  177707  001424     BIC     #177707,$TMP3   ;/SAVE RATE BITS.
(2)  006214  052737  004005  001424     BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
(2)  006222  013777  001424  173146     MOV     $TMP3,@ASR              ;/LOAD CSR.
(2)                                                             ;/THIS MUST BE DONE IN
(2)                                                             ;/ORDER TO XFERR COUNTER
(2)                                                             ;/TO BUFFER ON ST2.
(2)  006230  052777  001000  173140     BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
(2)  006236  017737  173136  001126     MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
(2)                                                             ;/PREVIOUS COUNTER
(2)  006244  012677  173126             MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
(2)  006250  005737  001126             TST     $BDDAT          ;/RESTORE CSR
(1)  006254  001004                     BNE     2$              ;/YES - NEXT TEST.
(1)  006256  105766  177776             TSTB    -2(6)           ;/AT HIGH RATE MAY HAVE HAD OVERFLOW
(1)                                                             ;/NOTE: CSR HAD BEEN PUT ON STACK.
(1)  006262  100401                     BMI     2$              ;/NEXT TEST IF OVERFLOW.
(1)
(2)

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

(1)  006264  104006                             ERROR   6               ;/CLOCK FAILED TO COUNT AT
(1)                                                                     ;/RATE:1KHZ
(2)

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

(1)
(1)  006266  005077  173104             2$:     CLR     @ASR            ;/CLEAR THE CLOCK.
(1)
(1)
```

```
814
(5)                                     ;;*********************************************************************
(4)                                     ;*TEST 40        *TEST THE ABILITY OF CLOCK TO COUNT AT 100HZ RATE
(5)
(5)                                     ;*
(5)                                     ;*THIS TEST IS DESIGNED TO TEST THE CLOCK'S ABILITY
(5)                                     ;*TO COUNT AT 100HZ RATE.
(5)                                     ;*
(4)                                     ;*********************************************************************
(3)  006272  000004                     TST40:  SCOPE
(2)  006274  012737  000005  001160             MOV     #5,$TIMES       ;;DO 5 ITERATIONS
(1)
(1)
(1)  006302  005077  173070             .        CLR     @ASR            ;/CLEAR CLOCK
(1)  006306  005077  173066                      CLR     @ABR            ;/CLEAR PRESET BUFFER
(1)  006312  012777  000051  173056.             MOV     #BIT0!50,@ASR   ;/START CLOCK, MODE0, RATE:100HZ
(1)  006320  005000                              CLR     R0              ;/NOW WE'LL DO A LITTLE DELAY. THIS DELAY
(1)
(1)  006322  005200                     1$:      INC     R0              ;/WILL AMOUNT TO APPROXIMATELY
(1)  006324  001376                              BNE     1$              ;/369 MS.
(1)
(2)  006326  017746  173044                      MOV     @ASR,-(6)       ;/SAVE CSR
(2)  006332  011637  001424                      MOV     (6),$TMP3       ;/GET CSR.
(2)  006336  042737  177707  001424             BIC     #177707,$TMP3   ;/SAVE RATE BITS.
(2)  006344  052737  004005  001424             BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
(2)  006352  013777  001424  173016             MOV     $TMP3,@ASR              ;/LOAD CSR.
(2)                                                                     ;/THIS MUST BE DONE IN
(2)                                                                     ;/ORDER TO XFERR COUNTER
(2)                                                                     ;/TO BUFFER ON ST2.
(2)  006360  052777  001000  173010             BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
(2)  006366  017737  173006  001126             MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
(2)                                                                     ;/PREVIOUS COUNTER
(2)  006374  012677  172776                      MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
(2)  006400  005737  001126                      TST     $BDDAT          ;/RESTORE CSR
(1)  006404  001004                              BNE     2$              ;/YES - NEXT TEST.
(1)  006406  105766  177776                      TSTB    -2(6)           ;/AT HIGH RATE MAY HAVE HAD OVERFLOW
(1)                                                                     ;/NOTE: CSR HAD BEEN PUT ON STACK.
(1)  006412  100401                              BMI     2$              ;/NEXT TEST IF OVERFLOW.
(1)
(2)

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

(1)  006414  104006                              ERROR   6               ;/CLOCK FAILED TO COUNT AT
(1)                                                                     ;/RATE:100HZ
(2)

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

(1)
(1)  006416  005077  172754             2$:      CLR     @ASR            ;/CLEAR THE CLOCK.
(1)
(1)
```

```
816
(5)                                     ;;**********************************************************************
(4)                                     ;*TEST 41        *TEST THE ABILITY OF CLOCK TO COUNT AT LINEFREQ RATE
(5)                                     ;*
(5)                                     ;*THIS TEST IS DESIGNED TO TEST THE CLOCK'S ABILITY
(5)                                     ;*TO COUNT AT LINEFREQ RATE.
(5)                                     ;*
(4)                                     ;;**********************************************************************
(3) 006422 000004                       TST41:  SCOPE
(2) 006424 012737 000005 001160                 MOV     #5,$TIMES       ;;DO 5 ITERATIONS
(1)
(1) 006432 032777 010000 172500                 BIT     #BIT12,@SWR     ;SHALL WE TEST
(1)                                                                     ;LINE FREQ?
(1)                                                                     ;SW12=1
(3) 006440 001450                               BEQ     TST42           ;;
(1)
(1) 006442 005077 172730                        CLR     @ASR            ;/CLEAR CLOCK
(1) 006446 005077 172726                        CLR     @ABR            ;/CLEAR PRESET BUFFER
(1) 006452 012777 000071 172716                 MOV     #BIT0!70,@ASR   ;/START CLOCK, MODE0, RATE:LINEFREQ
(1) 006460 005000                               CLR     R0              ;/NOW WE'LL DO A LITTLE DELAY. THIS DELAY
(1)
(1) 006462 005200                       1$:     INC     R0              ;/WILL AMOUNT TO APPROXIMATELY
(1) 006464 001376                               BNE     1$              ;/369 MS.
(1)
(2) 006466 017746 172704                        MOV     @ASR,-(6)       ;/SAVE CSR
(2) 006472 011637 001424                        MOV     (6),$TMP3       ;/GET CSR.
(2) 006476 042737 177707 001424                 BIC     #177707,$TMP3   ;/SAVE RATE BITS.
(2) 006504 052737 004005 001424                 BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
(2) 006512 013777 001424 172656                 MOV     $TMP3,@ASR              ;/LOAD CSR.
(2)                                                                     ;/THIS MUST BE DONE IN
(2)                                                                     ;/ORDER TO XFERR COUNTER
(2)                                                                     ;/TO BUFFER ON ST2.
(2) 006520 052777 001000 172650                 BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
(2) 006526 017737 172646 001126                 MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
(2)                                                                     ;/PREVIOUS COUNTER
(2) 006534 012677 172636                        MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
(2) 006540 005737 001126                        TST     $BDDAT          ;/RESTORE CSR
(1) 006544 001004                               BNE     2$              ;/YES - NEXT TEST.
(1) 006546 105766 177776                        TSTB    -2(6)           ;/AT HIGH RATE MAY HAVE HAD OVERFLOW
(1)                                                                     ;/NOTE: CSR HAD BEEN PUT ON STACK.
(1) 006552 100401                               BMI     2$              ;/NEXT TEST IF OVERFLOW.
(1)
(2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1) 006554 104006                               ERROR   6               ;/CLOCK FAILED TO COUNT AT
(1)                                                                     ;/RATE:LINEFREQ
(2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)
(1) 006556 005077 172614        2$:     CLR     @ASR            ;/CLEAR THE CLOCK.
(1)
(1)
817
```

```
819                                     ;;*********************************************************************
(3)                                     ;*TEST 42        *TEST THAT COUNTER DOESN'T COUNT WHEN ''SLAVE IN'' RATE IS SELECTED
(3)                                     ;;*********************************************************************
(2)  006562  000004                     TST42:  SCOPE
(1)  006564  012737  000020  001160             MOV     #20,$TIMES      ;;DO 20 ITERATIONS
820
821  006572  005077  172600                    CLR     @ASR            ;CLEAR CSR.
822  006576  005077  172576                    CLR     @ABR            ;CLEAR PRESET BUFFER.
823
824  006602  052777  000001  172566             BIS     #BIT0,@ASR      ;SET GO BIT, RATE:''SLAVE IN''
825
826  006610  052777  000400  172560             BIS     #BIT8,@ASR      ;GENERATE A MAINTENANCE ST1.
827  006616  005000                             CLR     R0
828
829  006620  005200                     1$:     INC     R0              ;NOW DO A SHORT DELAY.
830  006622  001376                             BNE     1$
831
832  006624  017746  172546                    MOV     @ASR,-(6)       ;/SAVE CSR
(1)  006630  011637  001424                    MOV     (6),$TMP3       ;/GET CSR.
(1)  006634  042737  177707  001424             BIC     #177707,$TMP3   ;/SAVE RATE BITS.
(1)  006642  052737  004005  001424             BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
(1)  006650  013777  001424  172520             MOV     $TMP3,@ASR              ;/LOAD CSR.
(1)                                                                     ;/THIS MUST BE DONE IN
(1)                                                                     ;/ORDER TO XFERR COUNTER
(1)                                                                     ;/TO BUFFER ON ST2.
(1)  006656  052777  001000  172512             BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
(1)  006664  017737  172510  001126             MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
(1)                                                                     ;/PREVIOUS COUNTER
(1)  006672  012677  172500                    MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
(1)  006676  005737  001126                    TST     $BDDAT          ;/RESTORE CSR
833  006702  001003                             BNE     2$              ;IF ANY COUNT, ERROR.
834  006704  105777  172466                    TSTB    @ASR            ;CHECK FOR OVERFLOW.
835  006710  100003                             BPL     3$              ;IF NO OVERFLOW, NEXT TEST
836
837  006712  005037  001124             2$:     CLR     $GDDAT          ;CLEAR FOR TYPEOUT, EXP'D ZERO.
838
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

839  006716  104011                             ERROR   11              ;CLOCK COUNTED WHEN RATE:''SLAVE IN''
840                                                                     ;WAS SELECTED.
841
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

842  006720  005077  172452             3$:     CLR     @ASR            ;CLEAR CLOCK'S CSR.
843
844
845                                     ;;*********************************************************************
(3)                                     ;*TEST 43        *TEST THAT THE CLOCK WILL COUNT IN MODE 1
(3)                                     ;;*********************************************************************
(2)  006724  000004                     TST43:  SCOPE
846
847  006726  005077  172444                    CLR     @ASR            ;CLEAR THE CSR.
848  006732  012777  177777  172440             MOV     #-1,@ABR        ;PRESET BUFFER REG.
849  006740  052777  000013  172430             BIS     #13,@ASR        ;START CLOCK,RATE:1 MHZ,MODE 1.
850  006746  005000                             CLR     R0
851  006750  105200                     1$:     INCB    R0              ;NOW DO A SHORT DELAY SO THAT THE
```

```
                                                J 5
 852  006752  001376                          BNE     1$              ;CLOCK CAN COUNT TO OVERFLOW.
 853
 854  006754  105777  172416                  TSTB    @ASR            ;OVERFLOW SHOULD HAVE SET.
 855  006760  100401                          BMI     2$              ;GOTO NEXT TEST IF SO.
 856
 857
                                              ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
 858  006762  104006                          ERROR   6               ;CLOCK FAILED TO COUNT UP AND SET
 859                                                                  ;OVERFLOW IN MODE 1.
 860
                                              ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
 861  006764                                  2$:
 862
 863
 864                                          .SBTTL  *
 865                                          .SBTTL  *PHASE 4 CLOCK INTERRUPT TEST.
 866                                          .SBTTL  *
 867
 875
 876                                          ;;**********************************************************************
 (3)                                          ;*TEST 44        *TEST THAT THE CLOCK WILL INTERRUPT ON OVERFLOW
 (3)                                          ;;**********************************************************************
 (2)  006764  000004                          TST44:  SCOPE
 (1)  006766  012737  000020  001160          MOV     #20,$TIMES      ;;DO 20 ITERATIONS
 877
 878
 (1)  006774  012746  000340                  MOV     #340,-(SP)      ;PUT PRIORITY ON STACK.
 (1)  007000  012746  007006                  MOV     #64$,-(SP)      ;PUT RETURN ADDRESS ON STACK
 (1)  007004  000002                          RTI                     ;DO AN RTI, PUTS PRIORITY IN CPU.
 (1)  007006                                  64$:
 879
 880  007006  005077  172364                  CLR     @ASR            ;CLEAR CLOCK'S CSR.
 881  007012  012777  177777  172360          MOV     #-1,@ABR        ;SET PRESET BUFFER TO ALL ONES.
 882
 883  007020  012777  000161  172350          MOV     #161,@ASR       ;START CLOCK, RATE:ST1.
 884  007026  052777  000400  172342          BIS     #BIT8,@ASR      ;GENERATE A MAINTENANCE ST1.
 885  007034  012777  007074  172340          MOV     #1$,@VECT1      ;SET INTERRUPT ADDR.
 886
 (1)  007042  012746  000000                  MOV     #0,-(SP)        ;PUT PRIORITY ON STACK.
 (1)  007046  012746  007054                  MOV     #65$,-(SP)      ;PUT RETURN ADDRESS ON STACK
 (1)  007052  000002                          RTI                     ;DO AN RTI, PUTS PRIORITY IN CPU.
 (1)  007054                                  65$:
 887
 888  007054  000240                          NOP                     ;STALL TIME
 889
 (1)  007056  012746  000340                  MOV     #340,-(SP)      ;PUT PRIORITY ON STACK.
 (1)  007062  012746  007070                  MOV     #66$,-(SP)      ;PUT RETURN ADDRESS ON STACK
 (1)  007066  000002                          RTI                     ;DO AN RTI, PUTS PRIORITY IN CPU.
 (1)  007070                                  66$:
 890
                                              ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
 891  007070  104007                          ERROR   7               ;CLOCK FAILED TO INTERRUPT.
 892
```

```
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
  893  007072  000402                          BR      2$
  894  007074                        1$:
  (1)  007074  062706  000004                  ADD     #4,SP                   ;/ADD #4 TO STACK POINTER.
  895  007100  005077  172272        2$:       CLR     @ASR            ;CLEAR THE CLOCK.
  896
  897                                   ;;*****************************************************************
  (3)                                   ;*TEST 45       *TEST THAT ST2 WILL CAUSE AN INTERRUPT
  (3)                                   ;;*****************************************************************
  (2)  007104  000004                   TST45:  SCOPE
  898
  899
  (1)  007106  012746  000340                   MOV     #340,-(SP)      ;PUT PRIORITY ON STACK.
  (1)  007112  012746  007120                   MOV     #64$,-(SP)      ;PUT RETURN ADDRESS ON STACK
  (1)  007116  000002                           RTI                     ;DO AN RTI, PUTS PRIORITY IN CPU.
  (1)  007120                           64$:
  900
  901  007120  005077  172252                   CLR     @ASR            ;CLEAR CLOCKS CSR.
  902  007124  012777  007200  172254           MOV     #1$,@VECT2      ;SET UP INTERRUPT VECTOR.
  903  007132  012777  040001  172236           MOV     #BIT14!BIT0,@ASR;SET ''INT2'',AND GO BIT.
  904  007140  052777  001000  172230           BIS     #BIT9,@ASR      ;GENERATE A MAINTENANCE ST2.
  905
  (1)  007146  012746  000000                   MOV     #0,-(SP)        ;PUT PRIORITY ON STACK.
  (1)  007152  012746  007160                   MOV     #65$,-(SP)      ;PUT RETURN ADDRESS ON STACK
  (1)  007156  000002                           RTI                     ;DO AN RTI, PUTS PRIORITY IN CPU.
  (1)  007160                           65$:
  906
  907  007160  000240                          NOP                     ;STALL TIME
  908
  (1)  007162  012746  000340                   MOV     #340,-(SP)      ;PUT PRIORITY ON STACK.
  (1)  007166  012746  007174                   MOV     #66$,-(SP)      ;PUT RETURN ADDRESS ON STACK
  (1)  007172  000002                           RTI                     ;DO AN RTI, PUTS PRIORITY IN CPU.
  (1)  007174                           66$:
  909
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

  910  007174  104007                          ERROR   7               ;CLOCK FAILED TO INTERRUPT ON ST2.
  911
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

  912  007176  000402                          BR      2$
  913  007200                        1$:
  (1)  007200  062706  000004                  ADD     #4,SP                   ;/ADD #4 TO STACK POINTER.
  914  007204  005077  172166        2$:       CLR     @ASR            ;CLEAR CLOCK'S CSR.
  915
  916                                           .SBTTL  *
  917                                           .SBTTL  *PHASE 5 ADVANCED TESTING
  918                                           .SBTTL  *
  919
```

```
1000                                    ;;*****************************************************************
 (3)                                    ;*TEST 46       *TEST THAT THE 'FOR' BIT WILL SET ON 2 ST2'S
 (3)                                    ;;*****************************************************************
 (2)  007210  000004                    TST46:  SCOPE
1001
1002  007212  005077  172160                    CLR     @ASR            ;START WITH CSR CLEAR.
1003  007216  005277  172154                    INC     @ASR            ;SET GO BIT.
1004  007222  052777  001000  172146           BIS #BIT9,@ASR          ; GENERATE THE 1ST ST2 PULSE.
1005  007230  052777  001000  172140           BIS     #BIT9,@ASR      ;GENERATE 2ND ST2 PULSE.
1006                                                                    ;THIS SHOULD CAUSE FOR BIT TO SET.
1007  007236  032777  010000  172132           BIT     #BIT12,@ASR     ;DID FOR BIT SET?
1008  007244  001007                          BNE     1$              ;YES-THEN NEXT TEST.
1009
1010  007246  017737  172124  001126           MOV     @ASR,$BDDAT     ;RECORD CSR.
1011  007254  012737  110001  001124           MOV     #BIT15!BIT12!BIT0,$GDDAT        ;RECORD S/B.
1012
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

1013  007262  104001                          ERROR   1               ;ERROR-'FOR' BIT FAILED TO SET ON
1014                                                                    ;ON TWO SUCCESIVE ST2 PULSES.
1015
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

1016  007264                            1$:
1017
1018                                    ;;*****************************************************************
 (3)                                    ;*TEST 47       *TEST THAT FOR BIT WILL SET ON TWO OVERFLOWS
 (3)                                    ;;*****************************************************************
 (2)  007264  000004                    TST47:  SCOPE
 (1)  007266  012737  000002  001160            MOV     #2,$TIMES       ;;DO 2 ITERATIONS
1019
1020  007274  005077  172076                    CLR     @ASR            ;START WITH CSR CLEAR.
1021  007300  012777  177777  172072           MOV     #-1,@ABR        ;PRELOAD BUFFER REG.
1022  007306  052777  000013  172062           BIS     #13,@ASR        ;START CLOCK,MODE 1,1MHZ.
1023  007314  005000                          CLR     R0              ;NOW DO A SHORT DELAY.
1024  007316  105200                  1$:     INCB    R0              ;DURING THIS DELAY, THE CLOCK WILL
1025  007320  001376                          BNE     1$              ;HAVE OVERFLOWED TWICE-
1026                                                                    ;THIS SHOULD CAUSE THE FOR BIT TO SET.
1027
1028  007322  032777  010000  172046           BIT     #BIT12,@ASR     ;DID FOR SET?
1029  007330  001007                          BNE     2$              ;YES-NEXT TEST.
1030
1031  007332  017737  172040  001126           MOV     @ASR,$BDDAT     ;NO-RECORD THE CSR.
1032  007340  012737  010213  001124           MOV     #010213,$GDDAT  ;RECORD S/B.
1033
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

1034  007346  104001                          ERROR   1               ;ERROR FOR BIT FAILED TO SET ON
1035                                                                    ; TWO SUCCESSIVE OVERFLOWS.
1036
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

1037  007350                            2$:
1038                                    ;;*****************************************************************
 (3)                                    ;*TEST 50       *TEST THAT FOR BIT WILL CLEAR IF GO BIT IS SET
 (3)                                    ;;*****************************************************************
```

```
  (2)  007350  000004                    TST50:  SCOPE
1039
1040   007352  012777  000001  172016            MOV     #BIT0,@ASR      ;CLEAR CSR,SET GO BIT.
1041   007360  052777  001000  172010            BIS     #BIT9,@ASR      ;SET 1ST ST2 PULSE.
1042   007366  052777  001000  172002            BIS     #BIT9,@ASR      ;GENERATE 2ND ST2 PULSE.
1043                                                                     ;FOR BIT SETS HERE.
1044   007374  042777  000001  171774            BIC     #BIT0,@ASR      ;CLEAR GO BIT.
1045
1046   007402  052777  000001  171766            BIS     #BIT0,@ASR      ;SET THE ''GO'' BIT AGAIN -
1047                                                                     ;SHOULD CLEAR FOR BIT.
1048
1049   007410  017737  171762  001126            MOV     @ASR,$BDDAT     ;READ THE CSR.
1050
1051   007416  012737  100001  001124            MOV     #100001,$GDDAT  ;RECORD WHAT CSR S/B.
1052   007424  032737  010000  001126            BIT     #BIT12,$BDDAT   ;DID FOR BIT CLEAR?
1053   007432  001401                            BEQ     1$              ;YES NEXT TEST.
1054
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

1055   007434  104001                            ERROR   1               ;FOR BIT FALED TO CLEAR
1056                                                                    ;WHEN ''GO'' BIT WAS SET.
1057
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

1058   007436  005077  171734            1$:     CLR     @ASR            ;CLEAR THE CSR.
1059
1060                                    ;;**********************************************************************
  (3)                                   ;*TEST 51       *TEST THAT WE CAN DISABLE THE INTERNAL OSC
  (3)                                   ;;**********************************************************************
  (2)  007442  000004                   TST51:  SCOPE
  (1)  007444  012737  000005  001160           MOV     #5,$TIMES       ;;DO 5 ITERATIONS
1061
1062   007452  005077  171720            1$:     CLR     @ASR            ;CLEAR THE CSR
1063   007456  005077  171716                    CLR     @ABR            ;CLEAR THE PRESET BUFFER
1064   007462  005037  001124                    CLR     $GDDAT          ;CLEAR EXPED.
1065
1066   007466  012777  004000  171702            MOV     #BIT11,@ASR     ;DISABLE THE INTERNAL OSC.
1067   007474  052777  000011  171674            BIS     #BIT3!BIT0,@ASR ;START CLOCK:RATE 1MHZ.
1068   007502  005000                            CLR     R0
1069   007504  105200                    1$:     INCB    R0              ;DELAY A SHORT TIME.
1070   007506  001376                            BNE     1$
1071   007510  017746  171662                    MOV     @ASR,-(6)       ;/SAVE CSR
  (1)  007514  011637  001424                    MOV     (6),$TMP3       ;/GET CSR.
  (1)  007520  042737  177707  001424            BIC     #177707,$TMP3   ;/SAVE RATE BITS.
  (1)  007526  052737  004005  001424            BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
  (1)  007534  013777  001424  171634            MOV     $TMP3,@ASR              ;/LOAD CSR.
  (1)                                                                   ;/THIS MUST BE DONE IN
  (1)                                                                   ;/ORDER TO XFERR COUNTER
  (1)                                                                   ;/TO BUFFER ON ST2.
  (1)  007542  052777  001000  171626            BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
  (1)  007550  017737  171624  001126            MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
  (1)                                                                   ;/PREVIOUS COUNTER
  (1)  007556  012677  171614                    MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
  (1)  007562  005737  001126                    TST     $BDDAT          ;/RESTORE CSR
1072   007566  001401                            BEQ     2$              ;NO - GOOD - NEXT TEST.
1073
```

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

1074  007570  104011                    ERROR   11              ;CLOCK DISABLE INTERNAL
1075                                                          ;OSC. DID NOT WORK.
1076
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

1077  007572  005077  171600            2$:     CLR     @ASR            ;CLEAR THE CSR.
1078
1079                                    ;;*****************************************************************
 (3)                                    ;*TEST 52       *TEST THAT CLOCK CAN BE COUNTED USING MAINTENANCE OSC
 (3)                                    ;;*****************************************************************
 (2)  007576  000004                    TST52:  SCOPE
1080
1081  007600  005077  171572            CLR     @ASR            ;CLEAR THE CSR.
1082  007604  005077  171570            CLR     @ABR            ;CLEAR THE PRESET BUFFER.
1083  007610  052777  004000  171560    BIS     #BIT11,@ASR     ;DISABLE THE INTERNAL OSC.
1084  007616  052777  000011  171552    BIS     #BIT3!BIT0,@ASR ;START CLOCK, 1MHZ, GO.
1085  007624  012700  177754            MOV     #-20.,R0        ;SET TO COUNT 20 TIMES
1086
1087  007630  052777  002000  171540  1$:     BIS     #BIT10,@ASR     ;GENERATE 1 MAINTENANCE OSC.
1088                                                          ;NOTE: AT 1MHZ, IT TAKES 10
1089                                                          ;MAINT. OSC TO EQUAL 1 COUNT
1090  007636  005200                    INC     R0              ;DO 20 MAINTENANCE OSC.
1091  007640  001373                    BNE     1$
1092
1093  007642  017746  171530            MOV     @ASR,-(6)       ;/SAVE CSR
 (1)  007646  011637  001424            MOV     (6),$TMP3       ;/GET CSR.
 (1)  007652  042737  177707  001424    BIC     #177707,$TMP3   ;/SAVE RATE BITS.
 (1)  007660  052737  004005  001424    BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
 (1)  007666  013777  001424  171502    MOV     $TMP3,@ASR              ;/LOAD CSR.
 (1)                                                          ;/THIS MUST BE DONE IN
 (1)                                                          ;/ORDER TO XFERR COUNTER
 (1)                                                          ;/TO BUFFER ON ST2.
 (1)  007674  052777  001000  171474    BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
 (1)  007702  017737  171472  001126    MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
 (1)                                                          ;/PREVIOUS COUNTER
 (1)  007710  012677  171462            MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
 (1)  007714  005737  001126            TST     $BDDAT          ;/RESTORE CSR
1094  007720  001001                    BNE     2$              ;YES - NEXT TEST.
1095
1096
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

1097  007722  104011                    ERROR   11              ;ERROR COULD NOT COUNT USING
1098                                                          ;MAINTENANCE OSC.
1099
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

1100  007724  005077  171446            2$:     CLR     @ASR            ;CLEAR THE CSR.
1101
```

```
 1171
 1172
  (1)
  (5)                                   ;;*****************************************************************
  (4)                                   ;*TEST 53        *TEST THE CLOCK'S 1MHZ DIVIDER
  (4)                                   ;;*****************************************************************
  (3) 007730  000004                    TST53:  SCOPE
  (2) 007732  012737  000005  001160            MOV     #5,$TIMES       ;;DO 5 ITERATIONS
  (1)
  (1) 007740  005077  171432                    CLR     @ASR            ;/CLEAR THE CSR.
  (1) 007744  005077  171430                    CLR     @ABR            ;/CLEAR THE PRESET BUFFER.
  (1) 007750  052777  004000  171420            BIS     #BIT11,@ASR     ;/DISABLE THE INTERNAL OSC.
  (1) 007756  052777  000011  171412            BIS     #1!10,@ASR      ;/ENABLE CLOCK, RATE:1MHZ
  (1)
  (1)
  (1) 007764  012700  177766          10$:      MOV     #-10.,R0        ;/SET TO GENERATE 10 OSC PULSES.
  (1)
  (1) 007770  052777  002000  171400  1$:       BIS     #BIT10,@ASR     ;/GENERATE ONE OSC PULSE.
  (1) 007776  005200                            INC     R0              ;/DONE 10 OSC PULSES?
  (1) 010000  001373                            BNE     1$              ;/NO - DO ANOTHER ONE.
  (1)
  (1)
  (1) 010002  012737  000001  001124  2$:       MOV     #1,$GDDAT       ;/SET FOR ERROR TYPEOUT - IF ANY.
  (2) 010010  017746  171362                    MOV     @ASR,-(6)       ;/SAVE CSR
  (2) 010014  011637  001424                    MOV     (6),$TMP3       ;/GET CSR.
  (2) 010020  042737  177707  001424            BIC     #177707,$TMP3   ;/SAVE RATE BITS.
  (2) 010026  052737  004005  001424            BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
  (2) 010034  013777  001424  171334            MOV     $TMP3,@ASR              ;/LOAD CSR.
  (2)                                                                   ;/THIS MUST BE DONE IN
  (2)                                                                   ;/ORDER TO XFERR COUNTER
  (2)                                                                   ;/TO BUFFER ON ST2.
  (2) 010042  052777  001000  171326            BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
  (2) 010050  017737  171324  001126            MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
  (2)                                                                   ;/PREVIOUS COUNTER
  (2) 010056  012677  171314                    MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
  (2) 010062  005737  001126                    TST     $BDDAT          ;/RESTORE CSR
  (1) 010066  013737  001124  001420            MOV     $GDDAT,$TMP0    ;/$TMP0 USED IN ERROR TYPEOUT.
  (1)
  (1) 010074  023737  001124  001126            CMP     $GDDAT,$BDDAT   ;/DID CLOCK ADVANCE ONCE?
  (1)
  (1) 010102  001402                            BEQ     3$              ;/YES - NEXT TEST.
  (2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

  (1) 010104  104011                            ERROR   11              ;/ERROR ON CLOCK1MHZ PULSE
  (1)                                                                   ;/NOT GENERATED WHEN 10
  (1)                                                                   ;/OSC PULSES GENERATED.
  (2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

  (1) 010106  000443                            BR      5$
  (1) 010110  012700  000011          3$:       MOV     #9.,R0  ;/GET THE NUMBER OF MORE OSC PULSES
  (1)                                                                   ;/TO BE GENERATED.
  (1)
  (1) 010114  052777  002000  171254  4$:       BIS     #BIT10,@ASR     ;/GENERATE ANOTHER OSC PULSE.
  (1) 010122  005300                            DEC     R0              ;/WHAT WE WANT TO CHECK
```

```
(1)  010124  001373                        BNE     4$              ;/1MHZ PULSE ON 9 OSC PULSES.
(1)
(1)
(2)  010126  017746  171244                MOV     @ASR,-(6)       ;/SAVE CSR
(2)  010132  011637  001424                MOV     (6),$TMP3       ;/GET CSR.
(2)  010136  042737  177707  001424        BIC     #177707,$TMP3   ;/SAVE RATE BITS.
(2)  010144  052737  004005  001424        BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
(2)  010152  013777  001424  171216        MOV     $TMP3,@ASR              ;/LOAD CSR.
(2)                                                 ;/THIS MUST BE DONE IN
(2)                                                 ;/ORDER TO XFERR COUNTER
(2)                                                 ;/TO BUFFER ON ST2.
(2)  010160  052777  001000  171210        BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
(2)  010166  017737  171206  001126        MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
(2)                                                 ;/PREVIOUS COUNTER
(2)  010174  012677  171176                MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
(2)  010200  005737  001126                TST     $BDDAT          ;/RESTORE CSR
(1)  010204  023737  001124  001126        CMP     $GDDAT,$BDDAT   ;/WAS ANOTHER 1MHZ PULSE GENERATED?
(1)  010212  001401                        BEQ     5$              ;/NO - NEXT TEST.
(2)

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

(1)  010214  104011                        ERROR   11              ;/WE SEEM TO HAVE GENERATED
(1)                                                 ;/ANOTHER 1MHZ PULSE ON
(1)                                                 ;/ONLY 9 MAINTENANCE
(1)                                                 ;/OSC PULSES.
(2)

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

(1)  010216  005077  171154        5$:     CLR     @ASR            ;/CLEAR THE CSR.
(1)
1173
(1)
(5)                                 ;;**********************************************************************
(4)                                 ;*TEST 54        *TEST THE CLOCK'S 100KHZ DIVIDER
(4)                                 ;;**********************************************************************
(3)  010222  000004                TST54:  SCOPE
(2)  010224  012737  000005  001160        MOV     #5,$TIMES       ;;DO 5 ITERATIONS
(1)
(1)  010232  005077  171140                CLR     @ASR            ;/CLEAR THE CSR.
(1)  010236  005077  171136                CLR     @ABR            ;/CLEAR THE PRESET BUFFER.
(1)  010242  052777  004000  171126        BIS     #BIT11,@ASR     ;/DISABLE THE INTERNAL OSC.
(1)  010250  052777  000021  171120        BIS     #1!20,@ASR      ;/ENABLE CLOCK, RATE:100KHZ
(1)
(1)
(1)  010256  012700  177634        10$:    MOV     #-100.,R0       ;/SET TO GENERATE 100 OSC PULSES.
(1)
(1)  010262  052777  002000  171106  1$:    BIS     #BIT10,@ASR     ;/GENERATE ONE OSC PULSE.
(1)  010270  005200                        INC     R0              ;/DONE 100 OSC PULSES?
(1)  010272  001373                        BNE     1$              ;/NO - DO ANOTHER ONE.
(1)
(1)
(1)  010274  012737  000001  001124  2$:    MOV     #1,$GDDAT       ;/SET FOR ERROR TYPEOUT - IF ANY.
(2)  010302  017746  171070                MOV     @ASR,-(6)       ;/SAVE CSR
(2)  010306  011637  001424                MOV     (6),$TMP3       ;/GET CSR.
(2)  010312  042737  177707  001424        BIC     #177707,$TMP3   ;/SAVE RATE BITS.
(2)  010320  052737  004005  001424        BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
```

```
(2)  010326  013777  001424  171042          MOV     $TMP3,@ASR              ;/LOAD CSR.
(2)                                                                          ;/THIS MUST BE DONE IN
(2)                                                                          ;/ORDER TO XFERR COUNTER
(2)                                                                          ;/TO BUFFER ON ST2.
(2)  010334  052777  001000  171034          BIS     #BIT9,@ASR              ;/GENERATE ON ST2 PULSE
(2)  010342  017737  171032  001126          MOV     @ABR,$BDDAT             ;/READ THE PRESET BUFFER,
(2)                                                                          ;/PREVIOUS COUNTER
(2)  010350  012677  171022                  MOV     (6)+,@ASR               ;/CONTENTS ARE IN $BDDAT.
(2)  010354  005737  001126                  TST     $BDDAT                  ;/RESTORE CSR
(1)  010360  013737  001124  001420          MOV     $GDDAT,$TMP0            ;/$TMP0 USED IN ERROR TYPEOUT.
(1)
(1)  010366  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;/DID CLOCK ADVANCE ONCE?
(1)
(1)  010374  001402                          BEQ     3$                      ;/YES - NEXT TEST.
(2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  010376  104011                          ERROR   11                      ;/ERROR ON CLOCK100KHZ PULSE
(1)                                                                          ;/NOT GENERATED WHEN 100
(1)                                                                          ;/OSC PULSES GENERATED.
(2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  010400  000443                          BR      5$
(1)  010402  012700  000143          3$:     MOV     #99.,R0 ;/GET THE NUMBER OF MORE OSC PULSES
(1)                                                                          ;/TO BE GENERATED.
(1)
(1)  010406  052777  002000  170762  4$:     BIS     #BIT10,@ASR             ;/GENERATE ANOTHER OSC PULSE.
(1)  010414  005300                          DEC     R0                      ;/WHAT WE WANT TO CHECK
(1)  010416  001373                          BNE     4$                      ;/100KHZ PULSE ON 99 OSC PULSES.
(1)
(1)
(2)  010420  017746  170752                  MOV     @ASR,-(6)               ;/SAVE CSR
(2)  010424  011637  001424                  MOV     (6),$TMP3               ;/GET CSR.
(2)  010430  042737  177707  001424          BIC     #177707,$TMP3           ;/SAVE RATE BITS.
(2)  010436  052737  004005  001424          BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
(2)  010444  013777  001424  170724          MOV     $TMP3,@ASR              ;/LOAD CSR.
(2)                                                                          ;/THIS MUST BE DONE IN
(2)                                                                          ;/ORDER TO XFERR COUNTER
(2)                                                                          ;/TO BUFFER ON ST2.
(2)  010452  052777  001000  170716          BIS     #BIT9,@ASR              ;/GENERATE ON ST2 PULSE
(2)  010460  017737  170714  001126          MOV     @ABR,$BDDAT             ;/READ THE PRESET BUFFER,
(2)                                                                          ;/PREVIOUS COUNTER
(2)  010466  012677  170704                  MOV     (6)+,@ASR               ;/CONTENTS ARE IN $BDDAT.
(2)  010472  005737  001126                  TST     $BDDAT                  ;/RESTORE CSR
(1)  010476  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;/WAS ANOTHER 100KHZ PULSE GENERATED?
(1)  010504  001401                          BEQ     5$                      ;/NO - NEXT TEST.
(2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  010506  104011                          ERROR   11                      ;/WE SEEM TO HAVE GENERATED
(1)                                                                          ;/ANOTHER 100KHZ PULSE ON
(1)                                                                          ;/ONLY 99 MAINTENANCE
(1)                                                                          ;/OSC PULSES.
(2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

```
  (1)  010510  005077  170662          5$:     CLR     @ASR            ;/CLEAR THE CSR.
  (1)
 1174
  (1)
  (5)                                  ;;*****************************************************************
  (4)                                  ;*TEST 55        *TEST THE CLOCK'S 10KHZ DIVIDER
  (4)                                  ;;*****************************************************************
  (3)  010514  000004                  TST55:  SCOPE
  (2)  010516  012737  000005  001160          MOV     #5,$TIMES       ;;DO 5 ITERATIONS
  (1)
  (1)  010524  005077  170646          CLR     @ASR            ;/CLEAR THE CSR.
  (1)  010530  005077  170644          CLR     @ABR            ;/CLEAR THE PRESET BUFFER.
  (1)  010534  052777  004000  170634  BIS     #BIT11,@ASR     ;/DISABLE THE INTERNAL OSC.
  (1)  010542  052777  000031  170626  BIS     #1!30,@ASR      ;/ENABLE CLOCK, RATE:10KHZ
  (1)
  (1)
  (1)  010550  012700  176030          10$:    MOV     #-1000.,R0      ;/SET TO GENERATE 1000 OSC PULSES.
  (1)
  (1)  010554  052777  002000  170614  1$:     BIS     #BIT10,@ASR     ;/GENERATE ONE OSC PULSE.
  (1)  010562  005200                  INC     R0              ;/DONE 1000 OSC PULSES?
  (1)  010564  001373                  BNE     1$              ;/NO - DO ANOTHER ONE.
  (1)
  (1)
  (1)  010566  012737  000001  001124  2$:     MOV     #1,$GDDAT       ;/SET FOR ERROR TYPEOUT - IF ANY.
  (2)  010574  017746  170576          MOV     @ASR,-(6)       ;/SAVE CSR
  (2)  010600  011637  001424          MOV     (6),$TMP3       ;/GET CSR.
  (2)  010604  042737  177707  001424  BIC     #177707,$TMP3   ;/SAVE RATE BITS.
  (2)  010612  052737  004005  001424  BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
  (2)  010620  013777  001424  170550  MOV     $TMP3,@ASR                      ;/LOAD CSR.
  (2)                                                                  ;/THIS MUST BE DONE IN
  (2)                                                                  ;/ORDER TO XFERR COUNTER
  (2)                                                                  ;/TO BUFFER ON ST2.
  (2)  010626  052777  001000  170542  BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
  (2)  010634  017737  170540  001126  MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
  (2)                                                                  ;/PREVIOUS COUNTER
  (2)  010642  012677  170530          MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
  (2)  010646  005737  001126          TST     $BDDAT          ;/RESTORE CSR
  (1)  010652  013737  001124  001420  MOV     $GDDAT,$TMP0    ;/$TMP0 USED IN ERROR TYPEOUT.
  (1)
  (1)  010660  023737  001124  001126  CMP     $GDDAT,$BDDAT   ;/DID CLOCK ADVANCE ONCE?
  (1)
  (1)  010666  001402                  BEQ     3$              ;/YES - NEXT TEST.
  (2)
                                       ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

  (1)  010670  104011                  ERROR   11              ;/ERROR ON CLOCK10KHZ PULSE
  (1)                                                          ;/NOT GENERATED WHEN 1000
  (1)                                                          ;/OSC PULSES GENERATED.
  (2)
                                       ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

  (1)  010672  000443                  BR      5$
  (1)  010674  012700  001747  3$:     MOV     #999.,R0        ;/GET THE NUMBER OF MORE OSC PULSES
  (1)                                                          ;/TO BE GENERATED.
  (1)
```

```
(1)  010700  052777  002000  170470  4$:     BIS     #BIT10,@ASR     ;/GENERATE ANOTHER OSC PULSE.
(1)  010706  005300                          DEC     R0              ;/WHAT WE WANT TO CHECK
(1)  010710  001373                          BNE     4$              ;/10KHZ PULSE ON 999 OSC PULSES.
(1)
(1)
(2)  010712  017746  170460                  MOV     @ASR,-(6)       ;/SAVE CSR
(2)  010716  011637  001424                  MOV     (6),$TMP3       ;/GET CSR.
(2)  010722  042737  177707  001424          BIC     #177707,$TMP3   ;/SAVE RATE BITS.
(2)  010730  052737  004005  001424          BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
(2)  010736  013777  001424  170432          MOV     $TMP3,@ASR              ;/LOAD CSR.
(2)                                                                  ;/THIS MUST BE DONE IN
(2)                                                                  ;/ORDER TO XFERR COUNTER
(2)                                                                  ;/TO BUFFER ON ST2.
(2)  010744  052777  001000  170424          BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
(2)  010752  017737  170422  001126          MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
(2)                                                                  ;/PREVIOUS COUNTER
(2)  010760  012677  170412                  MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
(2)  010764  005737  001126                  TST     $BDDAT          ;/RESTORE CSR
(1)  010770  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;/WAS ANOTHER 10KHZ PULSE GENERATED?
(1)  010776  001401                          BEQ     5$              ;/NO - NEXT TEST.
(2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  011000  104011                          ERROR   11              ;/WE SEEM TO HAVE GENERATED
(1)                                                                  ;/ANOTHER 10KHZ PULSE ON
(1)                                                                  ;/ONLY 999 MAINTENANCE
(1)                                                                  ;/OSC PULSES.
(2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  011002  005077  170370          5$:     CLR     @ASR            ;/CLEAR THE CSR.
(1)
1175
1176
(1)
(5)                                     ;;**************************************************************************
(4)                                     ;*TEST 56        *TEST THE CLOCK'S 1KHZ DIVIDER
(4)                                     ;;**************************************************************************
(3)  011006  000004                  TST56:  SCOPE
(2)  011010  012737  000005  001160          MOV     #5,$TIMES       ;;DO 5 ITERATIONS
(1)
(1)  011016  005077  170354                  CLR     @ASR            ;/CLEAR THE CSR.
(1)  011022  005077  170352                  CLR     @ABR            ;/CLEAR THE PRESET BUFFER.
(1)  011026  052777  004000  170342          BIS     #BIT11,@ASR     ;/DISABLE THE INTERNAL OSC.
(1)  011034  052777  000041  170334          BIS     #1!40,@ASR      ;/ENABLE CLOCK, RATE:1KHZ
(1)
(1)
(1)  011042  012700  154360          10$:    MOV     #-10000.,R0     ;/SET TO GENERATE 10000 OSC PULSES.
(1)
(1)  011046  052777  002000  170322  1$:     BIS     #BIT10,@ASR     ;/GENERATE ONE OSC PULSE.
(1)  011054  005200                          INC     R0              ;/DONE 10000 OSC PULSES?
(1)  011056  001373                          BNE     1$              ;/NO - DO ANOTHER ONE.
(1)
(1)
(1)  011060  012737  000001  001124  2$:     MOV     #1,$GDDAT       ;/SET FOR ERROR TYPEOUT - IF ANY.
(2)  011066  017746  170304                  MOV     @ASR,-(6)       ;/SAVE CSR
```

```
(2)  011072  011637  001424                 MOV     (6),$TMP3       ;/GET CSR.
(2)  011076  042737  177707  001424         BIC     #177707,$TMP3   ;/SAVE RATE BITS.
(2)  011104  052737  004005  001424         BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
(2)  011112  013777  001424  170256         MOV     $TMP3,@ASR              ;/LOAD CSR.
(2)                                                                         ;/THIS MUST BE DONE IN
(2)                                                                         ;/ORDER TO XFERR COUNTER
(2)                                                                         ;/TO BUFFER ON ST2.
(2)  011120  052777  001000  170250         BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
(2)  011126  017737  170246  001126         MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
(2)                                                                         ;/PREVIOUS COUNTER
(2)  011134  012677  170236                 MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
(2)  011140  005737  001126                 TST     $BDDAT          ;/RESTORE CSR
(1)  011144  013737  001124  001420         MOV     $GDDAT,$TMP0    ;/$TMP0 USED IN ERROR TYPEOUT.
(1)
(1)  011152  023737  001124  001126         CMP     $GDDAT,$BDDAT   ;/DID CLOCK ADVANCE ONCE?
(1)
(1)  011160  001402                         BEQ     3$              ;/YES - NEXT TEST.
(2)

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

(1)  011162  104011                         ERROR   11              ;/ERROR ON CLOCK1KHZ PULSE
(1)                                                                 ;/NOT GENERATED WHEN 10000
(1)                                                                 ;/OSC PULSES GENERATED.
(2)

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

(1)  011164  000443                         BR      5$
(1)  011166  012700  023417         3$:     MOV     #9999.,R0       ;/GET THE NUMBER OF MORE OSC PULSES
(1)                                                                 ;/TO BE GENERATED.
(1)
(1)  011172  052777  002000  170176 4$:     BIS     #BIT10,@ASR     ;/GENERATE ANOTHER OSC PULSE.
(1)  011200  005300                         DEC     R0              ;/WHAT WE WANT TO CHECK
(1)  011202  001373                         BNE     4$              ;/1KHZ PULSE ON 9999 OSC PULSES.
(1)
(1)
(2)  011204  017746  170166                 MOV     @ASR,-(6)       ;/SAVE CSR
(2)  011210  011637  001424                 MOV     (6),$TMP3       ;/GET CSR.
(2)  011214  042737  177707  001424         BIC     #177707,$TMP3   ;/SAVE RATE BITS.
(2)  011222  052737  004005  001424         BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
(2)  011230  013777  001424  170140         MOV     $TMP3,@ASR              ;/LOAD CSR.
(2)                                                                         ;/THIS MUST BE DONE IN
(2)                                                                         ;/ORDER TO XFERR COUNTER
(2)                                                                         ;/TO BUFFER ON ST2.
(2)  011236  052777  001000  170132         BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
(2)  011244  017737  170130  001126         MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
(2)                                                                         ;/PREVIOUS COUNTER
(2)  011252  012677  170120                 MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
(2)  011256  005737  001126                 TST     $BDDAT          ;/RESTORE CSR
(1)  011262  023737  001124  001126         CMP     $GDDAT,$BDDAT   ;/WAS ANOTHER 1KHZ PULSE GENERATED?
(1)  011270  001401                         BEQ     5$              ;/NO - NEXT TEST.
(2)

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

(1)  011272  104011                         ERROR   11              ;/WE SEEM TO HAVE GENERATED
(1)                                                                 ;/ANOTHER 1KHZ PULSE ON
(1)                                                                 ;/ONLY 9999 MAINTENANCE
```

```
(1)                                                     ;/OSC PULSES.
(2)
                                                ;;$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  011274  005077  170076           5$:     CLR     @ASR            ;/CLEAR THE CSR.
(1)
1177
(1)
(5)                                             ;;**********************************************************************
(4)                                             ;*TEST 57        *TEST THE CLOCK'S 100HZ DIVIDER
(4)                                             ;;**********************************************************************
(3)  011300  000004                     TST57:  SCOPE
(2)  011302  012737  000005  001160            MOV     #5,$TIMES       ;;DO 5 ITERATIONS
(1)
(1)  011310  005077  170062                     CLR     @ASR            ;/CLEAR THE CSR.
(1)  011314  005077  170060                     CLR     @ABR            ;/CLEAR THE PRESET BUFFER.
(1)  011320  052777  004000  170050            BIS     #BIT11,@ASR     ;/DISABLE THE INTERNAL OSC.
(1)  011326  052777  000051  170042            BIS     #1!50,@ASR      ;/ENABLE CLOCK, RATE:100HZ
(1)
(1)  011334  012701  000012                     MOV     #10.,R1
(1)
(1)  011340  012700  154360           10$:     MOV     #-10000.,R0     ;/SET TO GENERATE 10000 OSC PULSES.
(1)
(1)  011344  052777  002000  170024   1$:     BIS     #BIT10,@ASR     ;/GENERATE ONE OSC PULSE.
(1)  011352  005200                            INC     R0              ;/DONE 10000 OSC PULSES?
(1)  011354  001373                            BNE     1$              ;/NO - DO ANOTHER ONE.
(1)
(1)  011356  005301                            DEC     R1
(1)  011360  001367                            BNE     10$
(1)
(1)  011362  012737  000001  001124   2$:     MOV     #1,$GDDAT       ;/SET FOR ERROR TYPEOUT - IF ANY.
(2)  011370  017746  170002                    MOV     @ASR,-(6)       ;/SAVE CSR
(2)  011374  011637  001424                    MOV     (6),$TMP3       ;/GET CSR.
(2)  011400  042737  177707  001424            BIC     #177707,$TMP3   ;/SAVE RATE BITS.
(2)  011406  052737  004005  001424            BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
(2)  011414  013777  001424  167754            MOV     $TMP3,@ASR              ;/LOAD CSR.
(2)                                                     ;/THIS MUST BE DONE IN
(2)                                                     ;/ORDER TO XFERR COUNTER
(2)                                                     ;/TO BUFFER ON ST2.
(2)  011422  052777  001000  167746            BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
(2)  011430  017737  167744  001126            MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
(2)                                                     ;/PREVIOUS COUNTER
(2)  011436  012677  167734                    MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
(2)  011442  005737  001126                    TST     $BDDAT          ;/RESTORE CSR
(1)  011446  013737  001124  001420            MOV     $GDDAT,$TMP0    ;/$TMP0 USED IN ERROR TYPEOUT.
(1)
(1)  011454  023737  001124  001126            CMP     $GDDAT,$BDDAT   ;/DID CLOCK ADVANCE ONCE?
(1)
(1)  011462  001402                            BEQ     66$             ;/YES - NEXT TEST
(2)
                                                ;;$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  011464  104011                            ERROR   11              ;/ERROR ON CLOCK100HZ PULSE
(1)                                                     ;/NOT GENERATED WHEN 10000
(1)                                                     ;/OSC PULSES GENERATED.
(2)
```

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

(1)  011466  000447                    BR      5$
(1)  011470  012701  000012    66$:     MOV     #10.,R1
(1)  011474  012700  023417    3$:      MOV     #9999.,R0       ;/GET THE NUMBER OF MORE OSC PULSES
(1)                                                             ;/TO BE GENERATED.
(1)
(1)  011500  052777  002000  167670 4$:      BIS     #BIT10,@ASR     ;/GENERATE ANOTHER OSC PULSE.
(1)  011506  005300                    DEC     R0              ;/WHAT WE WANT TO CHECK
(1)  011510  001373                    BNE     4$              ;/100HZ PULSE ON 9999 OSC PULSES.
(1)  011512  005301                    DEC     R1
(1)  011514  001367                    BNE     3$
(1)
(1)
(2)  011516  017746  167654            MOV     @ASR,-(6)       ;/SAVE CSR
(2)  011522  011637  001424            MOV     (6),$TMP3       ;/GET CSR.
(2)  011526  042737  177707  001424    BIC     #177707,$TMP3   ;/SAVE RATE BITS.
(2)  011534  052737  004005  001424    BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
(2)  011542  013777  001424  167626    MOV     $TMP3,@ASR              ;/LOAD CSR.
(2)                                                             ;/THIS MUST BE DONE IN
(2)                                                             ;/ORDER TO XFERR COUNTER
(2)                                                             ;/TO BUFFER ON ST2.
(2)  011550  052777  001000  167620    BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
(2)  011556  017737  167616  001126    MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
(2)                                                             ;/PREVIOUS COUNTER
(2)  011564  012677  167606            MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
(2)  011570  005737  001126            TST     $BDDAT          ;/RESTORE CSR
(1)  011574  023737  001124  001126    CMP     $GDDAT,$BDDAT   ;/WAS ANOTHER 100HZ PULSE GENERATED?
(1)  011602  001401                    BEQ     5$              ;/NO - NEXT TEST.
(2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  011604  104011                            ERROR   11              ;/WE SEEM TO HAVE GENERATED
(1)                                                                     ;/ANOTHER 100HZ PULSE ON
(1)                                                                     ;/ONLY 9999 MAINTENANCE
(1)                                                                     ;/OSC PULSES.
(2)
                                        ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

(1)  011606  005077  167564    5$:      CLR     @ASR            ;/CLEAR THE CSR.
(1)
1178
1196
```

```
                                ;;*****************************************************************
                                ;*TEST 60        *TEST THE CLOCK'S MODE 2 OPERATION
                                ;*
                                ;*IN THIS TEST WE'LL CHECK MODE 2 OPERATION
                                ;*MODE 2: EXTERNAL EVENTS TIMING MODE
                                ;*SETTING THE GO BIT CAUSES THE COUNTER TO BEGIN COUNTING FROM
                                ;*ZERO AND TO FREE-RUN UNTIL THE GO BIT IS WRITTEN
                                ;*TO A ZERO THE COUNTER WILL CONTINUE COUNTING AFTER
                                ;*OVERFLOW. AN EXTERNAL PULSE FROM SCHMITZ TRIGGER 2
                                ;*(WHEN ST2 GO ENABLE IS A '0') CAUSES DATA TO
                                ;*TRANSFER FROM THE COUNTER TO THE BUFFER/PRESET REG.
                                ;*WHILE THE COUNTER CONTINUES TO RUN.
                                ;*
                                ;*TO TEST THIS MODE, WE'LL DISABLE THE INTERNAL OSC AND USE
                                ;*MAINTENANCE OSC PULSES AS WELL AS A MAINTENANCE
                                ;*ST2.
                                ;*
                                ;;*****************************************************************
011612  000004                  TST60:  SCOPE
011614  012737  000020  001160          MOV     #20,$TIMES      ;;DO 20 ITERATIONS
1199
1200    011622  005077  167550          CLR     @ASR            ;CLEAR THE CSR.
1201    011626  005077  167546          CLR     @ABR            ;CLEAR THE PRESET REG.
1202    011632  012777  004015  167536  MOV     #004015,@ASR    ;START CLOCK.
1203
1204    011640  012700  177754  1$:     MOV     #-20.,R0        ;SET TO GIVE 20 MAINTENANCE OSC.
1205    011644  052777  002000  167524  2$:     BIS     #BIT10,@ASR     ;GENERATE A MAINTENANCE OSC.
1206    011652  005200                  INC     R0
1207    011654  001373                  BNE     2$              ;IF NOT DONE 20 TIMES, LOOP.
1208
1209    011656  052777  001000  167512  3$:     BIS     #BIT9,@ASR      ;HERE'S THE BIGGIE! AN ST2 HAS BEEN GENERATED
1210    011664  012737  000002  001124          MOV     #2,$GDDAT       ;THE PRESET BUFFER SHOULD BE 2.
1211    011672  017737  167502  001126          MOV     @ABR,$BDDAT     ;READ THE PRESET BUFFER.
1212    011700  023737  001126  001124          CMP     $BDDAT,$GDDAT   ;DID A COUNTER TO PRESET BUFFER OCCUR?
1213    011706  001402                  BEQ     4$              ;YES - NEXT SUBTEST.
1214
                                ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

1215    011710  104005                  ERROR   5               ;A COUNTER TO PRESET BUFFER DID NOT
1216                                                            ;HAPPEN PROPERLY.
1217
                                ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

1218    011712  000434                  BR      5$
1219
1220    011714                  4$:
 (1)    011714  017746  167456          MOV     @ASR,-(6)       ;/SAVE CSR
 (1)    011720  011637  001424          MOV     (6),$TMP3       ;/GET CSR.
 (1)    011724  042737  177707  001424  BIC     #177707,$TMP3   ;/SAVE RATE BITS.
 (1)    011732  052737  004005  001424  BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
 (1)    011740  013777  001424  167430  MOV     $TMP3,@ASR              ;/LOAD CSR.
 (1)                                                            ;/THIS MUST BE DONE IN
 (1)                                                            ;/ORDER TO XFERR COUNTER
 (1)                                                            ;/TO BUFFER ON ST2.
 (1)    011746  052777  001000  167422  BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
 (1)    011754  017737  167420  001126  MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
```

```
  (1)                                                   ;/PREVIOUS COUNTER
  (1)  011762  012677  167410                 MOV      (6)+,@ASR          ;/CONTENTS ARE IN $BDDAT.
  (1)  011766  005737  001126                 TST      $BDDAT             ;/RESTORE CSR
 1221  011772  023737  001124  001126         CMP      $GDDAT,$BDDAT      ;WAS THE COUNTER ACCIDENTLY ZEROED?
 1222  012000  001401                         BEQ      5$                 ;NO - NEXT TEST.
 1223
                                             ;;$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 1224  012002  104005                         ERROR    5                  ;THE COUNT REGISTER SHOULD NOT
 1225                                                                     ;HAVE BEEN EFFECTED BY THE ST2
 1226                                                                     ;IN MODE 2.
 1227
                                             ;;$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 1228  012004                                 5$:
 1229
 1237
 1238                                        ;;*****************************************************************
  (3)                                        ;*TEST 61        *TEST THE CLOCK'S MODE 3 OPERATION
  (4)                                        ;*
  (4)                                        ;*IN THIS TEST WE'LL CHECK MODE 3 OPERATION.
  (4)                                        ;*MODE 3 IS JUST LIKE MODE 2 EXCEPT THAT THE COUNT
  (4)                                        ;*REG IS ZEROED AFTER AN ST2.
  (4)                                        ;*
  (3)                                        ;*****************************************************************
  (2)  012004  000004                        TST61:   SCOPE
  (1)  012006  012737  000020  001160                 MOV      #20,$TIMES         ;;DO 20 ITERATIONS
 1239
 1240  012014  005077  167356                         CLR      @ASR               ;CLEAR THE CSR.
 1241  012020  005077  167354                         CLR      @ABR               ;CLEAR THE BUFFER REG.
 1242  012024  012777  004017  167344                 MOV      #4017,@ASR         ;START CLOCK.
 1243
 1244  012032  012700  177754              1$:        MOV      #-20.,R0           ;SET TO GIVE 20 MAINTENANCE OSC.
 1245  012036  052777  002000  167332      2$:        BIS      #BIT10,@ASR        ;GENERATE A MAINTENANCE OSC.
 1246  012044  005200                               INC      R0
 1247  012046  001373                               BNE      2$                 ;IF NOT DONE 20 TIMES, LOOP.
 1248
 1249  012050  052777  001000  167320      3$:        BIS      #BIT9,@ASR         ;HERE'S THE BIGGIE! AN ST2 HAS BEEN GENERATED
 1250  012056  012737  000002  001124                 MOV      #2,$GDDAT          ;THE PRESET BUFFER SHOULD BE 2.
 1251  012064  017737  167310  001126                 MOV      @ABR,$BDDAT        ;READ THE PRESET BUFFER.
 1252  012072  023737  001126  001124                 CMP      $BDDAT,$GDDAT      ;DID A COUNTER TO PRESET BUFFER OCCUR?
 1253  012100  001402                               BEQ      4$                 ;YES - NEXT SUBTEST.
 1254
                                             ;;$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 1255  012102  104005                         ERROR    5                  ;A COUNTER TO PRESET BUFFER DID NOT
 1256                                                                     ;HAPPEN PROPERLY.
 1257
                                             ;;$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 1258  012104  000445                         BR       TST62              ;;
 1259
 1260  012106  005037  001124              4$:        CLR      $GDDAT             ;EXPECT ZERO BACK FROM COUNT REG.
 1261  012112  017746  167260                         MOV      @ASR,-(6)          ;/SAVE CSR
  (1)  012116  011637  001424                         MOV      (6),$TMP3          ;/GET CSR.
  (1)  012122  042737  177707  001424               BIC      #177707,$TMP3      ;/SAVE RATE BITS.
```

```
(1) 012130 052737 004005 001424          BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
(1) 012136 013777 001424 167232          MOV     $TMP3,@ASR              ;/LOAD CSR.
(1)                                                                      ;/THIS MUST BE DONE IN
(1)                                                                      ;/ORDER TO XFERR COUNTER
(1)                                                                      ;/TO BUFFER ON ST2.
(1) 012144 052777 001000 167224          BIS     #BIT9,@ASR              ;/GENERATE ON ST2 PULSE
(1) 012152 017737 167222 001126          MOV     @ABR,$BDDAT             ;/READ THE PRESET BUFFER,
(1)                                                                      ;/PREVIOUS COUNTER
(1) 012160 012677 167212                 MOV     (6)+,@ASR               ;/CONTENTS ARE IN $BDDAT.
(1) 012164 005737 001126                 TST     $BDDAT                  ;/RESTORE CSR
1262 012170 001402                       BEQ     5$                      ;IF SO - NEXT TEST.
1263

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

1264 012172 104005                       ERROR   5                       ;THE CLOCK FORGOT TO ZERO THE COUNT
1265                                                                     ;REG. AFTER AN ST2 OCCURRED ON
1266                                                                     ;A MODE 3 COUNT.
1267

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

1268 012174 000411                       BR      TST62                   ;;
1269 012176                      5$:
1270 012176 005077 167174                CLR     @ASR                    ;NOW TRY CLEARING THE CSR.
1271 012202 017737 167170 001126         MOV     @ASR,$BDDAT             ;READ THE CSR - DID IT CLEAR?
1272 012210 001403                       BEQ     TST62                   ;;
1273 012212 005037 001124                CLR     $GDDAT                  ;NO - RECORD S/B.
1274

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

1275 012216 104002                       ERROR   2                       ;CSR FAILED TO CLEAR
1276

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

1277
1278                             ;;*****************************************************************
 (3)                             ;*TEST 62        *IF ENABLED,CHECK THRESHOLD ST1 FROM TESTOR
 (3)                             ;;*****************************************************************
 (2) 012220 000004               TST62:  SCOPE
1279
1280 012222 012737 000002 001160         MOV     #2,$TIMES               ;;DO 2 ITERATIONS.
1281 012230 005737 001440                TST     EXS                     ;OPERATING IN TESTOR MODE?
1282 012234 001002                       BNE     4$                      ;YES DO THIS TEST.
1283 012236 000137 013056                JMP     ENDP                    ;NO-END PASS
1284 012242                      4$:
1285
1286 012242 005077 167130                CLR     @ASR                    ;YES-CLEAR CSR.
1287 012246 012777 177775 167124         MOV     #-3,@ABR                ;SET TO COUNT THREE TIMES.
1288 012254 012777 000061 167114         MOV     #61,@ASR                ;SET RATE: ST1,MODE 1 GO.
1289 012262 104401 012270                TYPE    ,65$                    ;;TYPE ASCIZ STRING
 (1) 012266 000432                       BR      64$                     ;;GET OVER THE ASCIZ
 (1)                             ;;65$:  .ASCIZ  <200><7><7>#SET ST1 THRESHOLD POT FULLY COUNTERCLOCKWIZE..#<7>
 (1) 012354                      64$:
1290 012354 004737 013740                JSR     PC,ANYKEY               ;TYPE LAST MESSAGE,AND WAIT FOR OPERATER.
1291 012360 012737 177775 001124         MOV     #-3,$GDDAT              ;DON'T EXPECT COUNT TO CHANGE.
1292 012366 017746 167004                MOV     @ASR,-(6)               ;/SAVE CSR
 (1) 012372 011637 001424                MOV     (6),$TMP3               ;/GET CSR.
```

```
  (1)  012376  042737  177707  001424          BIC     #177707,$TMP3   ;/SAVE RATE BITS.
  (1)  012404  052737  004005  001424          BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
  (1)  012412  013777  001424  166756          MOV     $TMP3,@ASR              ;/LOAD CSR.
  (1)                                                                  ;/THIS MUST BE DONE IN
  (1)                                                                  ;/ORDER TO XFERR COUNTER
  (1)                                                                  ;/TO BUFFER ON ST2.
  (1)  012420  052777  001000  166750          BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
  (1)  012426  017737  166746  001126          MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
  (1)                                                                  ;/PREVIOUS COUNTER
  (1)  012434  012677  166736                  MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
  (1)  012440  005737  001126                  TST     $BDDAT          ;/RESTORE CSR
 1293  012444  001002                          BNE     2$      ;IF NON-ZERO,WE'RE OK.
 1294
                                      ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 1295  012446  104002                          ERROR   2               ;COUNTERCLOCKWIZE THRESHOLD SETTING
 1296                                                                 ;OF ST1 SHOULD HAVE INHIBITED
 1297                                                                 ;ST1 FROM CAUSEING CLOCK TO COUNT.
 1298
                                      ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 1299  012450  000463                          BR      TST63           ;;
 1300
 1301  012452                              2$:
 1302  012452  104401  012460                TYPE    ,67$            ;;TYPE ASCIZ STRING
  (1)  012456  000425                        BR      66$             ;;GET OVER THE ASCIZ
  (1)                                 ;;67$:  .ASCIZ  <200><7><7>#SET ST1 THRESHOLD POT FULLY CLOCKWIZE#
  (1)  012532                         66$:
 1303  012532  004737  013740          3$:     JSR     PC,ANYKEY       ;TYPE LAST MESS. AND WAIT FOR OPERATOR.
 1304  012536  017746  166634                  MOV     @ASR,-(6)       ;/SAVE CSR
  (1)  012542  011637  001424                  MOV     (6),$TMP3       ;/GET CSR.
  (1)  012546  042737  177707  001424          BIC     #177707,$TMP3   ;/SAVE RATE BITS.
  (1)  012554  052737  004005  001424          BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
  (1)  012562  013777  001424  166606          MOV     $TMP3,@ASR              ;/LOAD CSR.
  (1)                                                                  ;/THIS MUST BE DONE IN
  (1)                                                                  ;/ORDER TO XFERR COUNTER
  (1)                                                                  ;/TO BUFFER ON ST2.
  (1)  012570  052777  001000  166600          BIS     #BIT9,@ASR      ;/GENERATE ON ST2 PULSE
  (1)  012576  017737  166576  001126          MOV     @ABR,$BDDAT     ;/READ THE PRESET BUFFER,
  (1)                                                                  ;/PREVIOUS COUNTER
  (1)  012604  012677  166566                  MOV     (6)+,@ASR       ;/CONTENTS ARE IN $BDDAT.
  (1)  012610  005737  001126                  TST     $BDDAT          ;/RESTORE CSR
 1305  012614  001001                          BNE     TST63           ;;
 1306
                                      ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 1307  012616  104002                          ERROR   2               ;CLOCKWIZE THRESHOLD SETTING OF
 1308                                                                 ;ST1 SHOULD HAVE INHIBITED CLOCK
 1309                                                                 ;FROM COUNTING.
 1310
                                      ;;$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 1311
 1312                                 ;;**********************************************************************
  (3)                                 ;*TEST 63        *IF ENABLED, CHECK ST1,ST2 IN FROM TESTOR
  (3)                                 ;;**********************************************************************
```

```
  (2)  012620  000004
  (1)  012622  012737  000002  001160  TST63:  SCOPE
                                              MOV     #2,$TIMES          ;;DO 2 ITERATIONS
1313
1314   012630  005737  001440                TST     EXS                ;OPERATING IN TESTOR MODE?
1315   012634  001510                        BEQ     ENDP               ;NO-EXIT THIS TEST.
1316
1317   012636  005077  166534                CLR     @ASR               ;CLEAR THE CSR
1318   012642  012777  177775  166530        MOV     #-3,@ABR           ;SET TO COUNT THREE TIMES
1319   012650  012777  000061  166520        MOV     #61,@ASR           ;SET RATE:ST1, MODE1, GO
1320   012656  104401  012664                TYPE    ,65$               ;;TYPE ASCIZ STRING
  (1)  012662  000433                        BR      64$                ;;GET OVER THE ASCIZ
  (1)                                 ;;65$:  .ASCIZ  <200><7><7>#SET ST1 THRESHOLD POT IN THE MIDDLE OF ITS RANGE.#
  (1)  012752                         64$:
1321   012752  004737  013740                JSR     PC,ANYKEY          ;TYPE LAST MESS WAITE FOR OPERATOR.
1322   012756  005037  001124                CLR     $GDDAT             ;EXPECT COUNTER TO BE CLEAR
1323   012762  017746  166410                MOV     @ASR,-(6)          ;/SAVE CSR
  (1)  012766  011637  001424                MOV     (6),$TMP3          ;/GET CSR.
  (1)  012772  042737  177707  001424        BIC     #177707,$TMP3      ;/SAVE RATE BITS.
  (1)  013000  052737  004005  001424        BIS     #BIT11!BIT2!BIT0,$TMP3  ;/SET MODE 2, NO RATE,DISABLE INTERNAL OSC
  (1)  013006  013777  001424  166362        MOV     $TMP3,@ASR                 ;/LOAD CSR.
  (1)                                                                        ;/THIS MUST BE DONE IN
  (1)                                                                        ;/ORDER TO XFERR COUNTER
  (1)                                                                        ;/TO BUFFER ON ST2.
  (1)  013014  052777  001000  166354        BIS     #BIT9,@ASR         ;/GENERATE ON ST2 PULSE
  (1)  013022  017737  166352  001126        MOV     @ABR,$BDDAT        ;/READ THE PRESET BUFFER,
  (1)                                                                        ;/PREVIOUS COUNTER
  (1)  013030  012677  166342                MOV     (6)+,@ASR          ;/CONTENTS ARE IN $BDDAT.
  (1)  013034  005737  001126                TST     $BDDAT             ;/RESTORE CSR
1324                                                                         ;/OF NON-ZERO - REPORT ERROR
1325   013040  001402                        BEQ     2$
1326

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

1327   013042  104002                                ERROR   2                  ;INCORRECT # OF ST1'S
1328                                                                            ;RECEIVED BY CLOCK
1329

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

1330   013044  000404                        BR      ENDP
1331
1332   013046                      2$:
1333   013046  005777  166324                TST     @ASR               ;DID ST2 FLG SET?
1334   013052  100401                        BMI     ENDP
1335

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

1336   013054  104006                                ERROR   6                  ;ST2 FLAG DID SET EVEN THOUGH
1337                                                                            ;SWITCH WAS TOGGLED.
1338

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

1339
1340                                                ;
1341                                                ;NOW WE'LL DETERMINE IF WE ARE ALLOWED TO AUTO-SIZE.
1342                                                ;IF SO, WE'LL FIND OUT IF THERE ARE OTHER CLOCKS OUT THERE
1343                                                ;TO TEST.
```

```
1344                                            ;
1345
1346  013056  000004                ENDP:   SCOPE
1347
1348
1349  013060  105737  001215                 TSTB    $ENVM           ;SEE IF APT WILL LET UP AUTO-SIZE.
1350  013064  100537                         BMI     2$              ;NO - EXIT.
1351
1352
1353  013066  023737  001434  001436         CMP     MDEVCT,TSTCNT   ;TESTED MAX. UNITS?
1354  013074  001507                         BEQ     4$              ;YES EXIT.
1355  013076  006337  001426                 ASL     ROTATE          ;POINT NEXT UNIT.
1356  013102  005237  001434                 INC     MDEVCT
1357
1358  013106  062737  000004  001376         ADD     #4,ASR          ;YES, ADD TO BASE ADDR.
1359  013114  013746  000004                 MOV     ERRVEC,-(6)     ;SAVE CONTENTS OF LOC 4.
1360  013120  012737  013274  000004         MOV     #1$,ERRVEC      ;SET UP IN CASE NO MORE CLOCKS.
1361
1362  013126  005777  166244                 TST     @ASR            ;TIME OUT HERE IF NO MORE CLOCKS.
1363
1364                                                                  ;IF HERE, ANOTHER CLOCK FOUND.
1365  013132  005737  001202                 TST     $PASS           ;IS THIS 1ST PASS?
1366  013136  001003                         BNE     3$              ;NO-GET OUT.
1367  013140  053737  001426  001430         BIS     ROTATE,UTEST    ;YES-RECORD THIS UNIT.
1368  013146                        3$:
1369  013146  104401  013154                 TYPE    ,65$            ;;TYPE ASCIZ STRING
 (1)  013152  000405                         BR      64$             ;;GET OVER THE ASCIZ
 (1)                            ;;65$:   .ASCIZ  <15><12>'UNIT #'
 (1)  013166                    64$:
1370  013166  013746  001204                 MOV     $DEVCT,-(SP)    ;;SAVE $DEVCT FOR TYPEOUT
 (1)  013172  104402                         TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
1371  013174  104401  013202                 TYPE    ,67$            ;;TYPE ASCIZ STRING
 (1)  013200  000406                         BR      66$             ;;GET OVER THE ASCIZ
 (1)                            ;;67$:   .ASCIZ  '' COMPLETED ''
 (1)  013216                    66$:
1372  013216  005237  001204                 INC     $DEVCT
1373  013222  104401  013230                 TYPE    ,69$            ;;TYPE ASCIZ STRING
 (1)  013226  000410                         BR      68$             ;;GET OVER THE ASCIZ
 (1)                            ;;69$:   .ASCIZ  '' TESTING UNIT #''
 (1)  013250                    68$:
1374  013250  013746  001204                 MOV     $DEVCT,-(SP)    ;;SAVE $DEVCT FOR TYPEOUT
 (1)  013254  104402                         TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
1375  013256  012637  000004                 MOV     (6)+,ERRVEC     ;RESETORE LOC 4.
1376  013262  062737  000010  001402         ADD     #10,VECT1       ;UPDATE VECTOR ADDR.
1377  013270  000137  002334                 JMP     LOOP            ;TEST NEW UNIT.
1378
1379  013274                        1$:
 (1)  013274  062706  000004                 ADD     #4,SP                   ;/ADD #4 TO STACK POINTER.
1380  013300  012637  000004                 MOV     (6)+,ERRVEC     ;RESTORE LOC 4
1381  013304  022737  000000  001204         CMP     #0,$DEVCT       ;TESTED ONLY ONE UNIT?
1382  013312  001424                         BEQ     2$              ;YES - NO NEED FOR TYPEOUT.
1383
1384  013314                        4$:
1385  013314  104401  013322                 TYPE    ,71$            ;;TYPE ASCIZ STRING
 (1)  013320  000405                         BR      70$             ;;GET OVER THE ASCIZ
 (1)                            ;;71$:   .ASCIZ  <15><12>'UNIT #''
```

```
                                T63     *IF ENABLED, CHECK ST1,ST2 IN FROM TESTOR                                SEQ 0080

  (1)  013334                           70$:
 1386  013334  013746  001204                    MOV     $DEVCT,-(SP)    ;;SAVE $DEVCT FOR TYPEOUT
  (1)  013340  104402                            TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 1387  013342  104401  013350                    TYPE    ,73$            ;;TYPE ASCIZ STRING
  (1)  013346  000406                            BR      72$             ;;GET OVER THE ASCIZ
  (1)                           ;;73$:  .ASCIZ  '' COMPLETED ''
  (1)  013364                   72$:
 1388
 1389  013364  013737  001250  001376  2$:     MOV     $BASE,ASR
 1390  013372  013737  001244  001402          MOV     $VECT1,VECT1
 1391  013400  013737  001204  001442          MOV     $DEVCT,LCNT
 1392  013406  005237  001442                  INC     LCNT
 1393  013412  012737  000000  001204          MOV     #0,$DEVCT
 1394
 1395  013420  005037  001434                  CLR     MDEVCT          ;BEGIN TESTING 1ST UNIT.
 1396  013424  012737  000001  001426          MOV     #1,ROTATE       ;POINT TO IT.
 1408
 1409
 1410                           .SBTTL  END OF PASS ROUTINE
  (1)
  (2)                           ;;**************************************************************************
  (1)                           ;*INCREMENT THE PASS NUMBER ($PASS)
  (1)                           ;*IF THERES A MONITOR GO TO IT
  (1)                           ;*IF THERE ISN'T JUMP TO LOOP
  (1)
  (1)  013432                   $EOP:
  (2)  013432  000240                    NOP
  (1)  013434  005037  001102            CLR     $TSTNM          ;;ZERO THE TEST NUMBER
  (1)  013440  005037  001160            CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
  (1)  013444  005237  001202            INC     $PASS           ;;INCREMENT THE PASS NUMBER
  (1)  013450  042737  100000  001202    BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
  (1)  013456  005327                    DEC     (PC)+           ;;LOOP?
  (1)  013460  000001           $EOPCT: .WORD   1
  (1)  013462  003122                    BGT     $DOAGN          ;;YES
  (1)  013464  012737                    MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
  (1)  013466  000001           $ENDCT: .WORD   1
  (1)  013470  013460                    $EOPCT
  (3)  013472  104401  013500            TYPE    ,65$            ;;TYPE ASCIZ STRING
  (3)  013476  000406                    BR      64$             ;;GET OVER THE ASCIZ
  (3)                           ;;65$:  .ASCIZ  <15><12>#ENDPASS  #
  (3)  013514                   64$:
  (3)  013514  013746  001202            MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
  (3)  013520  104402                    TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
  (3)  013522  104401  013530            TYPE    ,67$            ;;TYPE ASCIZ STRING
  (3)  013526  000411                    BR      66$             ;;GET OVER THE ASCIZ
  (3)                           ;;67$:  .ASCIZ  #  TOTAL ERRORS  #
  (3)  013552                   66$:
  (3)  013552  013746  001432            MOV     ERCNT,-(SP)     ;;SAVE ERCNT FOR TYPEOUT
  (3)  013556  104402                    TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
  (3)  013560  104401  013566            TYPE    ,69$            ;;TYPE ASCIZ STRING
  (3)  013564  000407                    BR      68$             ;;GET OVER THE ASCIZ
  (3)                           ;;69$:  .ASCIZ  #;  THERE ARE #
  (3)  013604                   68$:
  (3)  013604  013746  001442            MOV     LCNT,-(SP)      ;;SAVE LCNT FOR TYPEOUT
  (3)  013610  104402                    TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
  (3)  013612  104401  013620            TYPE    ,71$            ;;TYPE ASCIZ STRING
```

```
  (3)  013616  000411                        BR      70$             ;;GET OVER THE ASCIZ
  (3)                                 ;;71$:  .ASCIZ  #  (OCTAL) UNITS.#
  (3)  013642                         70$:
  (3)  013642  104401  013650                TYPE    ,73$            ;;TYPE ASCIZ STRING
  (3)  013646  000415                        BR      72$             ;;GET OVER THE ASCIZ
  (3)                                 ;;73$:  .ASCIZ  <200>#THE GOOD UNITS (L TO R) #
  (3)  013702                         72$:
  (3)  013702  013746  001430                MOV     UTEST,-(SP)     ;;SAVE UTEST FOR TYPEOUT
  (3)  013706  104405                        TYPBN                   ;;GO TYPE--BINARY ASCII
  (1)  013710  013700  000042        $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
  (1)  013714  001405                        BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
  (1)  013716  000005                        RESET                   ;;CLEAR THE WORLD
  (1)  013720  004710                $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
  (1)  013722  000240                        NOP                     ;;SAVE ROOM
  (1)  013724  000240                        NOP                     ;;FOR
  (1)  013726  000240                        NOP                     ;;ACT11
  (1)  013730                        $DOAGN:
  (1)  013730  000137                        JMP     @(PC)+          ;;RETURN
  (1)  013732  002334                $RTNAD: .WORD   LOOP
  (1)  013734     377      377  000  $ENULL: .BYTE   -1,-1,0         ;;NULL CHARACTER STRING
  (1)          013740                        .EVEN
 1411
 1412                                        ;
 1413                                        ;THIS ROUTINE TYPES LAST MESSAGE AND WAITS FOR AN  OPERATOR
 1414                                        ;RESPONCE.
 1415                                        ;
 1416
 1417  013740  105777  165202        ANYKEY: TSTB    @$TKB           ;CLEAR TTY READY FLAG.
 1418  013744  104401  013752                TYPE    ,65$            ;;TYPE ASCIZ STRING
  (1)  013750  000430                        BR      64$             ;;GET OVER THE ASCIZ
  (1)                                 ;;65$:  .ASCIZ  <200><7>#SWITCH ST1 3 TIMES,TYPE ANY KEY WHEN DONE..#<7>
  (1)  014032                         64$:
 1419
 1420  014032  105777  165106        1$:     TSTB    @$TKS           ;WAIT FOR OPERATOR.
 1421  014036  100375                        BPL     1$
 1422  014040  105777  165102                TSTB    @$TKB           ;CLEAR TTY READY FLAG.
 1423  014044  000207                        RTS PC
 1443
```

```
1445                                    ;
1446                            .SBTTL  ;I/O SIGNAL TEST #1 ST1 IN AND ST2 OUT IN AND OUT
1447                                    ;
1448
 (1)                                    ;
 (1)                                    ;SWITCH PACK S2 MUST BE SET UP AS FOLLOWS:
 (1)                                    ;
 (1)                                    ;       SWITCH  1 - OFF
 (1)                                    ;               2 - ON
 (1)                                    ;               3 - OFF
 (1)                                    ;               4 - OFF
 (1)                                    ;               5 - ON
 (1)                                    ;               6 - ON
 (1)                                    ;               7 - NOT USED
 (1)                                    ;
 (1)                                    ; THIS SELECTS TTL THRESHOLDS AND POSITIVE SLOPE FOR
 (1)                                    ; SCHMITT TRIGGER 1.
 (1)                                    ;
 (1)                                    ;PLEASE REMOVE ANY PREVIOUS JUMPER.
 (1)                                    ;
1449                                    ;JUMPER THE FOLLOWING PINS TOGETHER:
1450                                    ;
1451                                    ;       J1 - SS (ST2 OUT)       TO J1 - VV (ST1-IN)
1452                                    ;
1453                                    ;LOAD AND START AT LOCATION 210
1454                                    ;END PASSES OCCUR IMMEDIATELY AND ARE NOT REPORTED
1455                                    ;ERRORS ARE REPORTED AS IN THE REGULAR LOGIC TEST AND
1456                                    ;THEIR PRINTOUT MAY BE INHIBITED
1457
1458  014046  012737  014060  001420  OITST1: MOV     #IOTST1,$TMP0   ;LOAD RETURN ADDRESS
1459  014054  000137  001526                  JMP     INIT            ;PRIME THE PROGRAM VECTOR SPACE
1460  014060  104407                  IOTST1: CKSWR                   ;CHECK THE SWR
1461  014062  005077  165310          1$:     CLR     @ASR            ;CLEAR THE CSR
1462  014066  005077  165306                  CLR     @ABR            ;CLEAR THE BUFFER REG.
1463  014072  012777  000061  165276          MOV     #61,@ASR        ;RATE ST1, MODE 0, GO.
1464  014100  052777  001000  165270          BIS     #BIT9,@ASR      ;GENERATE A MAINTENANCE ST2.
1465  014106  012777  000005  165262          MOV     #5,@ASR         ;NOW SET TO READ COUNT REG
1466  014114  052777  001000  165254          BIS     #BIT9,@ASR      ;FORCE COUNT -> BUFFER REG.
1467  014122  027727  165252  000001          CMP     @ABR,#1         ;DID COUNT REG ADVANCE ONCE?
1468  014130  001753                          BEQ     IOTST1          ;YES - LOOP.
1469  014132  104000                          ERROR                   ;ST2 OUT TO ST1 IN FAILED.
1470  014134  000751                          BR      IOTST1
1471
```

```
                                        .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 - OFF
                                                        ;               6 - ON
                                                        ;               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.
                                                        ;
014136  012737  014150  001420  OITST2: MOV     #IOTST2,$TMP0   ;LOAD RETURN ADDRESS
014144  000137  001526                JMP     INIT            ;PRIME THE PROGRAM VECTOR SPACE
014150  104407                IOTST2: CKSWR                   ;CHECK THE SWR.
014152  005077  165220                CLR     @ASR            ;CLEAR THE CSR.
014156  012777  177777  165214        MOV     #-1,@ABR        ;PRELOAD PRESET BUFFER.
014164  012777  000063  165204        MOV     #63,@ASR        ;RATE ST1, MODE 1, GO.
014172  052777  000400  165176        BIS     #BIT8,@ASR      ;GENERATE A MAIN. ST1.
014200  000240                        NOP
014202  000240                        NOP
014204  005777  165166                TST     @ASR            ;DID OVERFLOW SET ST2 FLAG?
014210  100757                        BMI     IOTST2          ;YES - LOOP
014212  104000                        ERROR                   ;CLK OV OUT TO ST2 IN FAILED.
014214  000755                        BR      IOTST2          ;LOOP
```

```
1501                                                    ;
1502                                    .SBTTL          ;I/O SIGNAL TEST #3 ST1 OUT AND ST2 IN
1503                                                    ;
1504
1505                                                    ;
1506                                                    ;SWITCH PACK S2 MUST BE SET UP AS FOLLOWS:
1507                                                    ;       SWITCH  1 - OFF
1508                                                    ;               2 - OFF
1509                                                    ;               3 - OFF
1510                                                    ;               4 - ON
1511                                                    ;               5 - ON
1512                                                    ;               6 - ON
1513                                                    ;               7 - NOT USED
1514                                                    ; THIS SELECTS TTL THRESHOLD AND POSITIVE SLOPE FOR
1515                                                    ; SCHMITT TRIGGER 2.
1516                                                    ;
1517                                                    ;PLEASE REMOVE ANY PREVIOUS JUMPERS.
1518                                                    ;
1519                                                    ;JUMPER THE FOLLOWING PINS TOGETHER:
1520                                                    ;       J1 - UU (ST1 OUT)       TO J1 - TT (ST2-IN)
1521                                                    ;
1522                                                    ;LOAD AND START AT LOCATION 220
1523                                                    ;END PASSES OCCUR IMMEDIATELY AND ARE NOT REPORTED
1524                                                    ;ERRORS ARE REPORTED AS IN THE REGULAR LOGIC TEST AND
1525                                                    ;THEIR PRINTOUT MAY BE INHIBITED
1526
1527  014216  012737  014230  001420  OITST3: MOV     #IOTST3,$TMP0   ;LOAD RETURN ADDRESS
1528  014224  000137  001526            JMP     INIT                ;PRIME THE PROGRAM VECTOR SPACE
1529  014230  104407                    IOTST3: CKSWR               ;CHECK THE SWR
1530  014232  012777  000001  165136            MOV     #1,@ASR     ;SET GO BIT.
1531  014240  052777  000400  165130            BIS     #BIT8,@ASR  ;GENERATE A MAIN. ST1.
1532  014246  005777  165124                    TST     @ASR        ;DID ST2 FLAG SET?
1533  014252  100401                            BMI     1$
1534  014254  104000                            ERROR               ;ST1 OUT TO ST2 IN FAILED
1535
1536  014256  032777  010000  165112  1$:       BIT     #BIT12,@ASR ;DID "FOR" BIT SET?
1537  014264  001761                            BEQ     IOTST3      ;NO - GOOD!
1538  014266  104000                            ERROR               ;"FOR" BIT SET ON ONLY 1 ST2.
1539  014270  000757                            BR      IOTST3      ;LOOP
```

```
1541
1542                                    .SBTTL
1543                                    .SBTTL  *SYSMAC ROUTINES
1544                                    .SBTTL
1545
1546                                .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
 (1)
 (2)                                ;;**********************************************************************
 (1)                                ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
 (1)                                ;*OCTAL (ASCII) NUMBER AND TYPE IT.
 (1)                                ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
 (1)                                ;*CALL:
 (1)                                ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
 (1)                                ;*      TYPOS                   ;;CALL FOR TYPEOUT
 (1)                                ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
 (1)                                ;*      .BYTE   M               ;;M=1 OR 0
 (1)                                ;*                                      ;;1=TYPE LEADING ZEROS
 (1)                                ;*                                      ;;0=SUPPRESS LEADING ZEROS
 (1)                                ;*
 (1)                                ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
 (1)                                ;*$TYPOS OR $TYPOC
 (1)                                ;*CALL.
 (1)                                ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
 (1)                                ;*      TYPON                   ;;CALL FOR TYPEOUT
 (1)                                ;*
 (1)                                ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
 (1)                                ;*CALL:
 (1)                                ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
 (1)                                ;*      TYPOC                   ;;CALL FOR TYPEOUT
 (1)
 (1)  014272  017646  000000        $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
 (1)  014276  116637  000001  014515        MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
 (1)  014304  112637  014517                MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
 (1)  014310  062716  000002                ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
 (1)  014314  000406                        BR      $TYPON
 (1)  014316  112737  000001  014515  $TYPOC: MOVB   #1,$OFILL       ;;SET THE ZERO FILL SWITCH
 (1)  014324  112737  000006  014517        MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
 (1)  014332  112737  000005  014514  $TYPON: MOVB   #5,$OCNT        ;;SET THE ITERATION COUNT
 (1)  014340  010346                        MOV     R3,-(SP)        ;;SAVE R3
 (1)  014342  010446                        MOV     R4,-(SP)        ;;SAVE R4
 (1)  014344  010546                        MOV     R5,-(SP)        ;;SAVE R5
 (1)  014346  113704  014517                MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
 (1)  014352  005404                        NEG     R4
 (1)  014354  062704  000006                ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
 (1)  014360  110437  014516                MOVB    R4,$OMODE       ;;SAVE IT FOR USE
 (1)  014364  113704  014515                MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
 (1)  014370  016605  000012                MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
 (1)  014374  005003                        CLR     R3              ;;CLEAR THE OUTPUT WORD
 (1)  014376  006105                1$:     ROL     R5              ;;ROTATE MSB INTO "C"
 (1)  014400  000404                        BR      3$              ;;GO DO MSB
 (1)  014402  006105                2$:     ROL     R5              ;;FORM THIS DIGIT
 (1)  014404  006105                        ROL     R5
 (1)  014406  006105                        ROL     R5
 (1)  014410  010503                        MOV     R5,R3
 (1)  014412  006103                3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
 (1)  014414  105337  014516                DECB    $OMODE          ;;TYPE THIS DIGIT?
```

```
(1)  014420  100016                          BPL     7$                      ;;BR IF NO
(1)  014422  042703  177770                  BIC     #177770,R3              ;;GET RID OF JUNK
(1)  014426  001002                          BNE     4$                      ;;TEST FOR 0
(1)  014430  005704                          TST     R4                      ;;SUPPRESS THIS 0?
(1)  014432  001403                          BEQ     5$                      ;;BR IF YES
(1)  014434  005204                  4$:     INC     R4                      ;;DON'T SUPPRESS ANYMORE 0'S
(1)  014436  052703  000060                  BIS     #'0,R3                  ;;MAKE THIS DIGIT ASCII
(1)  014442  052703  000040          5$:     BIS     #' ,R3                  ;;MAKE ASCII IF NOT ALREADY
(1)  014446  110337  014512                  MOVB    R3,8$                   ;;SAVE FOR TYPING
(1)  014452  104401  014512                  TYPE    ,8$                     ;;GO TYPE THIS DIGIT
(1)  014456  105337  014514          7$:     DECB    $OCNT                   ;;COUNT BY 1
(1)  014462  003347                          BGT     2$                      ;;BR IF MORE TO DO
(1)  014464  002402                          BLT     6$                      ;;BR IF DONE
(1)  014466  005204                          INC     R4                      ;;INSURE LAST DIGIT ISN'T A BLANK
(1)  014470  000744                          BR      2$                      ;;GO DO THE LAST DIGIT
(1)  014472  012605                  6$:     MOV     (SP)+,R5                ;;RESTORE R5
(1)  014474  012604                          MOV     (SP)+,R4                ;;RESTORE R4
(1)  014476  012603                          MOV     (SP)+,R3                ;;RESTORE R3
(1)  014500  016666  000002  000004          MOV     2(SP),4(SP)             ;;SET THE STACK FOR RETURNING
(1)  014506  012616                          MOV     (SP)+,(SP)
(1)  014510  000002                          RTI                             ;;RETURN
(1)  014512     000                  8$:     .BYTE   0                       ;;STORAGE FOR ASCII DIGIT
(1)  014513     000                          .BYTE   0                       ;;TERMINATOR FOR TYPE ROUTINE
(1)  014514     000                  $OCNT:  .BYTE   0                       ;;OCTAL DIGIT COUNTER
(1)  014515     000                  $OFILL: .BYTE   0                       ;;ZERO FILL SWITCH
(1)  014516  000000                  $OMODE: .WORD   0                       ;;NUMBER OF DIGITS TO TYPE
1547                                 .SBTTL  BINARY TO ASCII AND TYPE ROUTINE
(1)
(2)                                  ;;**********************************************************************
(1)                                  ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 16-BIT
(1)                                  ;*BINARY-ASCII NUMBER AND TYPE IT.
(1)                                  ;*CALL:
(1)                                  ;*      MOV     NUMBER,-(SP)            ;;NUMBER TO BE TYPED
(1)                                  ;*      TYPBN                           ;;TYPE IT
(1)
(1)  014520  010146                  $TYPBN: MOV     R1,-(SP)                ;;SAVE R1 ON THE STACK
(1)  014522  016601  000006                  MOV     6(SP),R1                ;;GET THE INPUT NUMBER
(1)  014526  000261                          SEC                             ;;SET "C" SO CAN KEEP TRACK OF THE NUMBER OF BITS
(1)  014530  112737  000060  014572  1$:     MOVB    #'0,$BIN                ;;SET CHARACTER TO AN ASCII "0".
(1)  014536  006101                          ROL     R1                      ;;GET THIS BIT
(1)  014540  001406                          BEQ     2$                      ;;DONE?
(1)  014542  105537  014572                  ADCB    $BIN                    ;;NO--SET THE CHARACTER EQUAL TO THIS BIT
(1)  014546  104401  014572                  TYPE    ,$BIN                   ;;GO TYPE THIS BIT
(1)  014552  000241                          CLC                             ;;CLEAR "C" SO CAN KEEP TRACK OF BITS
(1)  014554  000765                          BR      1$                      ;;GO DO THE NEXT BIT
(1)  014556  012601                  2$:     MOV     (SP)+,R1                ;;POP THE STACK INTO R1
(1)  014560  016666  000002  000004          MOV     2(SP),4(SP)             ;;ADJUST THE STACK
(1)  014566  012616                          MOV     (SP)+,(SP)
(1)  014570  000002                          RTI                             ;;RETURN TO USER
(1)  014572     000     000          $BIN:   .BYTE   0,0                     ;;STORAGE FOR ASCII CHAR. AND TERMINATOR
1548
1559
1560                                 .SBTTL  ERROR HANDLER ROUTINE
(1)
(2)                                  ;;**********************************************************************
(1)                                  ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
```

```
(1)                                     ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
(1)                                     ;*AND GO TO $ERRTYP ON ERROR
(1)                                     ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
(1)                                     ;*SW15=1        HALT ON ERROR
(1)                                     ;*SW13=1        INHIBIT ERROR TYPEOUTS
(1)                                     ;*SW10=1        BELL ON ERROR
(1)                                     ;*SW09=1        LOOP ON ERROR
(1)                                     ;*CALL
(1)                                     ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER
(1)
(1)  014574                             $ERROR:
(1)  014574  104407                             CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
(1)  014576  105237  001103             7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
(1)  014602  001775                             BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
(1)  014604  013777  001102  164330             MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
(1)  014612  032777  002000  164320             BIT     #BIT10,@SWR     ;;BELL ON ERROR?
(1)  014620  001402                             BEQ     1$              ;;NO - SKIP
(1)  014622  104401  001164                     TYPE    ,$BELL          ;;RING BELL
(1)  014626  005237  001112             1$:     INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
(1)  014632  011637  001116                     MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
(1)  014636  162737  000002  001116             SUB     #2,$ERRPC
(1)  014644  117737  164246  001114             MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
(1)  014652  032777  020000  164260             BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
(1)  014660  001004                             BNE     20$             ;;SKIP TYPEOUTS
(1)  014662  004737  015002                     JSR     PC,$ERRTYP      ;;GO TO USER ERROR ROUTINE
(1)  014666  104401  001171                     TYPE    ,$CRLF
(1)  014672                             20$:
(1)  014672  122737  000001  001214             CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
(1)  014700  001007                             BNE     2$              ;;NO,SKIP APT ERROR REPORT
(1)  014702  113737  001114  014714             MOVB    $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
(1)  014710  004737  016474                     JSR     PC,$ATY4        ;;REPORT FATAL ERROR TO APT
(1)  014714     000                     21$:    .BYTE   0
(1)  014715     000                             .BYTE   0
(1)  014716  000777                     22$:    BR      22$             ;;APT ERROR LOOP
(1)  014720  005777  164214             2$:     TST     @SWR            ;;HALT ON ERROR
(1)  014724  100002                             BPL     3$              ;;SKIP IF CONTINUE
(1)  014726  000000                             HALT                    ;;HALT ON ERROR!
(1)  014730  104407                             CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
(1)  014732  032777  001000  164200     3$:     BIT     #BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
(1)  014740  001402                             BEQ     4$              ;;BR IF NO
(1)  014742  013716  001110                     MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
(1)  014746  005737  001162             4$:     TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
(1)  014752  001402                             BEQ     5$              ;;BR IF NONE
(1)  014754  013716  001162                     MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
(1)  014760                             5$:
(3)
(3)  014760  005237  001432                     INC     ERCNT           ;/UPDATE ERROR COUNT.
(3)  014764  001002                             BNE     10$             ;/BUT DON'T LET IT OVERFLOW.
(3)  014766  005337  001432                     DEC     ERCNT           ;/KEEP AT 177777 IF OVERFLOW.
(3)  014772                             10$:
(3)  014772  043737  001426  001430             BIC     ROTATE,UTEST    ;/REMOVE UNIT FROM LIST OF GOOD ONES.
(3)  015000  000002                             RTI                     ;/EXIT.
(3)
1561                                    .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE
(1)
(2)                                     ;;**************************************************************
```

```
(1)                                     ;*THIS ROUTINE USES THE ''ITEM CONTROL BYTE'' ($ITEMB) TO DETERMINE WHICH
(1)                                     ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE ''ERROR TABLE'' ($ERRTB),
(1)                                     ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
(1)
(1)  015002                             $ERRTYP:
(1)  015002  104401  001171                     TYPE    ,$CRLF          ;;''CARRIAGE RETURN'' & ''LINE FEED''
(1)  015006  010046                             MOV     R0,-(SP)        ;;SAVE R0
(1)  015010  005000                             CLR     R0              ;;PICKUP THE ITEM INDEX
(1)  015012  153700  001114                     BISB    @#$ITEMB,R0
(1)  015016  001004                             BNE     1$              ;;IF ITEM NUMBER IS ZERO, JUST
(1)                                                                     ;;TYPE THE PC OF THE ERROR
(2)  015020  013746  001116                     MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
(2)                                                                     ;;ERROR ADDRESS
(2)  015024  104402                             TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
(1)  015026  000426                             BR      6$              ;;GET OUT
(1)  015030  005300                     1$:     DEC     R0              ;;ADJUST THE INDEX SO THAT IT WILL
(1)  015032  006300                             ASL     R0              ;;      WORK FOR THE ERROR TABLE
(1)  015034  006300                             ASL     R0
(1)  015036  006300                             ASL     R0
(1)  015040  062700  001256                     ADD     #$ERRTB,R0      ;;FORM TABLE POINTER
(1)  015044  012037  015054                     MOV     (R0)+,2$        ;;PICKUP ''ERROR MESSAGE'' POINTER
(1)  015050  001404                             BEQ     3$              ;;SKIP TYPEOUT IF NO POINTER
(1)  015052  104401                             TYPE                    ;;TYPE THE ''ERROR MESSAGE''
(1)  015054  000000                     2$:     .WORD   0               ;;''ERROR MESSAGE'' POINTER GOES HERE
(1)  015056  104401  001171                     TYPE    ,$CRLF          ;;''CARRIAGE RETURN'' & ''LINE FEED''
(1)  015062  012037  015072             3$:     MOV     (R0)+,4$        ;;PICKUP ''DATA HEADER'' POINTER
(1)  015066  001404                             BEQ     5$              ;;SKIP TYPEOUT IF 0
(1)  015070  104401                             TYPE                    ;;TYPE THE ''DATA HEADER''
(1)  015072  000000                     4$:     .WORD   0               ;;''DATA HEADER'' POINTER GOES HERE
(1)  015074  104401  001171                     TYPE    ,$CRLF          ;;''CARRIAGE RETURN'' & ''LINE FEED''
(1)  015100  011000                     5$:     MOV     (R0),R0         ;;PICKUP ''DATA TABLE'' POINTER
(1)  015102  001004                             BNE     7$              ;;GO TYPE THE DATA
(1)  015104  012600                     6$:     MOV     (SP)+,R0        ;;RESTORE R0
(1)  015106  104401  001171                     TYPE    ,$CRLF          ;;''CARRIAGE RETURN'' & ''LINE FEED''
(1)  015112  000207                             RTS     PC              ;;RETURN
(1)  015114                             7$:
(2)  015114  013046                             MOV     @(R0)+,-(SP)    ;;SAVE @(R0)+ FOR TYPEOUT
(2)  015116  104402                             TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
(1)  015120  005710                             TST     (R0)            ;;IS THERE ANOTHER NUMBER?
(1)  015122  001770                             BEQ     6$              ;;BR IF NO
(1)  015124  104401  015132                     TYPE    ,8$             ;;TYPE TWO(2) SPACES
(1)  015130  000771                             BR      7$              ;;LOOP
(1)  015132  020040      000           8$:     .ASCIZ  /  /            ;;TWO(2) SPACES
(1)          015136                             .EVEN
1562                                    .SBTTL  SCOPE HANDLER ROUTINE
(1)
(2)                                     ;;******************************************************************************
(1)                                     ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
(1)                                     ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
(1)                                     ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
(1)                                     ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
(1)                                     ;*SW14=1        LOOP ON TEST
(1)                                     ;*SW11=1        INHIBIT ITERATIONS
(1)                                     ;*SW09=1        LOOP ON ERROR
(1)                                     ;*SW08=1        LOOP ON TEST IN SWR<7:0>
(1)                                     ;*CALL
```

```
                                                    L 7
   (1)                                  :*      SCOPE           ;;SCOPE=IOT
   (1)
   (1)  015136                          $SCOPE:
   (1)  015136  104407                          CKSWR                   ;;TEST  FOR CHANGE IN SOFT-SWR
   (2)  015140  104407                          CKSWR
   (1)  015142  032777  040000  163770  1$:     BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
   (1)  015150  001114                          BNE     $OVER           ;;YES IF SW14=1
   (1)                                  ;#####START OF CODE FOR THE XOR TESTER#####
   (1)  015152  000416                  $XTSTR: BR      6$              ;;IF RUNNING ON THE ''XOR'' TESTER CHANGE
   (1)                                                                  ;;THIS INSTRUCTION TO A ''NOP'' (NOP=240)
   (1)  015154  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
   (1)  015160  012737  015200  000004          MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
   (1)  015166  005737  177060                  TST     @#177060        ;;TIME OUT ON XOR?
   (1)  015172  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
   (1)  015176  000463                          BR      $SVLAD          ;;GO TO THE NEXT TEST
   (1)  015200  022626                  5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
   (1)  015202  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
   (1)  015206  000423                          BR      7$              ;;LOOP ON THE PRESENT TEST
   (1)  015210                          6$:;#####END OF CODE FOR THE XOR TESTER#####
   (1)  015210  032777  000400  163722          BIT     #BIT08,@SWR     ;;LOOP ON SPEC. TEST?
   (1)  015216  001404                          BEQ     2$              ;;BR IF NO
   (1)  015220  127737  163714  001102          CMPB    @SWR,$TSTNM     ;;ON THE RIGHT TEST?    SWR<7:0>
   (1)  015226  001465                          BEQ     $OVER           ;;BR IF YES
   (1)  015230  105737  001103          2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
   (1)  015234  001421                          BEQ     3$              ;;BR IF NO
   (1)  015236  123737  001115  001103          CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
   (1)  015244  101015                          BHI     3$              ;;BR IF NO
   (1)  015246  032777  001000  163664          BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
   (1)  015254  001404                          BEQ     4$              ;;BR IF NO
   (1)  015256  013737  001110  001106  7$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
   (1)  015264  000446                          BR      $OVER
   (1)  015266  105037  001103          4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
   (1)  015272  005037  001160                  CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
   (1)  015276  000415                          BR      1$              ;;ESCAPE TO THE NEXT TEST
   (1)  015300  032777  004000  163632  3$:     BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
   (1)  015306  001011                          BNE     1$              ;;BR IF YES
   (1)  015310  005737  001202                  TST     $PASS           ;;IF FIRST PASS OF PROGRAM
   (1)  015314  001406                          BEQ     1$              ;;      INHIBIT ITERATIONS
   (1)  015316  005237  001104                  INC     $ICNT           ;;INCREMENT ITERATION COUNT
   (1)  015322  023737  001160  001104          CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
   (1)  015330  002024                          BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
   (1)  015332  012737  000001  001104  1$:     MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
   (1)  015340  013737  015416  001160          MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
   (1)  015346  105237  001102          $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
   (1)  015352  113737  001102  001200          MOVB    $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
   (1)  015360  011637  001106                  MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
   (1)  015364  011637  001110                  MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
   (1)  015370  005037  001162                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
   (1)  015374  112737  000001  001115          MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(!) ERROR ON NEXT TEST
   (1)  015402  013777  001102  163532  $OVER:  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
   (1)  015410  013716  001106                  MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
   (1)  015414  000002                          RTI                     ;;FIXES PS
   (1)  015416  003720                  $MXCNT: 2000.                   ;;MAX. NUMBER OF ITERATIONS
1563                                    .SBTTL  TTY INPUT ROUTINE
   (1)
   (2)                                  ;;**********************************************************************
```

```
(1)                                     .ENABL  LSB
(1)
(2)                                     ;;**********************************************************************
(1)                                     ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
(1)                                     ;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
(1)                                     ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
(1)                                     ;*WHEN OPERATING IN TTY FLAG MODE.
(1)  015420  022737  000176  001140  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED?
(1)  015426  001074                         BNE     15$             ;;BRANCH IF NO
(1)  015430  105777  163510                 TSTB    @$TKS           ;;CHAR THERE?
(1)  015434  100071                         BPL     15$             ;;IF NO, DON'T WAIT AROUND
(1)  015436  117746  163504                 MOVB    @$TKB,-(SP)     ;;SAVE THE CHAR
(1)  015442  042716  177600                 BIC     #^C177,(SP)     ;;STRIP-OFF THE ASCII
(1)  015446  022726  000007                 CMP     #7,(SP)+        ;;IS IT A CONTROL G?
(1)  015452  001062                         BNE     15$             ;;NO, RETURN TO USER
(1)  015454  123727  001134  000001         CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
(1)  015462  001456                         BEQ     15$             ;;BRANCH IF YES
(1)
(1)  015464  104401  016145                 TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
(1)  015470  104401  016152          $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
(2)  015474  013746  000176                 MOV     SWREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
(2)  015500  104402                         TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
(1)  015502  104401  016163                 TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
(1)  015506  005046                  19$:   CLR     -(SP)           ;;CLEAR COUNTER
(1)  015510  005046                         CLR     -(SP)           ;;THE NEW SWR
(1)  015512  105777  163426          7$:    TSTB    @$TKS           ;;CHAR THERE?
(1)  015516  100375                         BPL     7$              ;;IF NOT TRY AGAIN
(1)
(1)  015520  117746  163422                 MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
(1)  015524  042716  177600                 BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
(1)
(1)
(1)
(1)  015530  021627  000025          9$:    CMP     (SP),#25        ;;IS IT A CONTROL-U?
(1)  015534  001005                         BNE     10$             ;;BRANCH IF NOT
(1)  015536  104401  016140                 TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (^U)
(1)  015542  062706  000006          20$:   ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
(1)  015546  000757                         BR      19$             ;;LET'S TRY IT AGAIN
(1)
(1)
(1)  015550  021627  000015          10$:   CMP     (SP),#15        ;;IS IT A <CR>?
(1)  015554  001022                         BNE     16$             ;;BRANCH IF NO
(1)  015556  005766  000004                 TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
(1)  015562  001403                         BEQ     11$             ;;BRANCH IF YES
(1)  015564  016677  000002  163346         MOV     2(SP),@SWR      ;;SAVE NEW SWR
(1)  015572  062706  000006          11$:   ADD     #6,SP           ;;CLEAR UP STACK
(1)  015576  104401  001171          14$:   TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
(1)  015602  123727  001135  000001         CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
(1)  015610  001003                         BNE     15$             ;;BRANCH IF NOT
(1)  015612  012777  000100  163324         MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
(1)  015620  000002                  15$:   RTI                     ;;RETURN
(1)  015622  004737  016406          16$:   JSR     PC,$TYPEC       ;;ECHO CHAR
(1)  015626  021627  000060                 CMP     (SP),#60        ;;CHAR < 0?
(1)  015632  002420                         BLT     18$             ;;BRANCH IF YES
(1)  015634  021627  000067                 CMP     (SP),#67        ;;CHAR > 7?
(1)  015640  003015                         BGT     18$             ;;BRANCH IF YES
```

```
(1)  015642  042726  000060            BIC     #60,(SP)+       ;;STRIP-OFF ASCII
(1)  015646  005766  000002            TST     2(SP)           ;;IS THIS THE FIRST CHAR
(1)  015652  001403                    BEQ     17$             ;;BRANCH IF YES
(1)  015654  006316                    ASL     (SP)            ;;NO, SHIFT PRESENT
(1)  015656  006316                    ASL     (SP)            ;;   CHAR OVER TO MAKE
(1)  015660  006316                    ASL     (SP)            ;;   ROOM FOR NEW ONE.
(1)  015662  005266  000002    17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
(1)  015666  056616  177776            BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
(1)  015672  000707                    BR      7$              ;;GET THE NEXT ONE
(1)  015674  104401  001170    18$:    TYPE    ,$QUES          ;;TYPE ?<CR><LF>
(1)  015700  000720                    BR      20$             ;;SIMULATE CONTROL-U
(1)                            .DSABL  LSB
(1)
(1)
(2)                            ;;*****************************************************************
(1)                            ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
(1)                            ;*CALL:
(1)                            ;*      RDCHR                   ;;INPUT A SINGLE CHARACTER FROM THE TTY
(1)                            ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
(1)                            ;*                              ;;WITH PARITY BIT STRIPPED OFF
(1)                            ;
(1)
(1)  015702  011646    $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
(1)  015704  016666  000004  000002    MOV     4(SP),2(SP)     ;;SAVE THE PS
(1)  015712  105777  163226    1$:     TSTB    @$TKS           ;;WAIT FOR
(1)  015716  100375                    BPL     1$              ;;A CHARACTER
(1)  015720  117766  163222  000004    MOVB    @$TKB,4(SP)     ;;READ THE TTY
(1)  015726  042766  177600  000004    BIC     #^C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
(1)  015734  026627  000004  000023    CMP     4(SP),#23       ;;IS IT A CONTROL-S?
(1)  015742  001013                    BNE     3$              ;;BRANCH IF NO
(1)  015744  105777  163174    2$:     TSTB    @$TKS           ;;WAIT FOR A CHARACTER
(1)  015750  100375                    BPL     2$              ;;LOOP UNTIL ITS THERE
(1)  015752  117746  163170            MOVB    @$TKB,-(SP)     ;;GET CHARACTER
(1)  015756  042716  177600            BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
(1)  015762  022627  000021            CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
(1)  015766  001366                    BNE     2$              ;;IF NOT DISCARD IT
(1)  015770  000750                    BR      1$              ;;YES, RESUME
(1)  015772  026627  000004  000140    CMP     4(SP),#140      ;;IS IT UPPER CASE?
(1)  016000  002407                    BLT     4$              ;;BRANCH IF YES
(1)  016002  026627  000004  000175    CMP     4(SP),#175      ;;IS IT A SPECIAL CHAR?
(1)  016010  003003                    BGT     4$              ;;BRANCH IF YES
(1)  016012  042766  000040  000004    BIC     #40,4(SP)       ;;MAKE IT UPPER CASE
(1)  016020  000002            4$:     RTI                     ;;GO BACK TO USER
(2)                            ;;*****************************************************************
(1)                            ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
(1)                            ;*CALL:
(1)                            ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
(1)                            ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
(1)                            ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
(1)
(1)  016022  010346    $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
(1)  016024  012703  016130    1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
(1)  016030  022703  016140    2$:     CMP     #$TTYIN+8.,R3   ;;BUFFER FULL?
(1)  016034  101405                    BLOS    4$              ;;BR IF YES
(1)  016036  104410                    RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
(1)  016040  112613                    MOVB    (SP)+,(R3)      ;;GET CHARACTER
```

```
  (1)  016042  122713  000177          10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
  (1)  016046  001003                          BNE     3$              ;;SKIP IF NOT
  (1)  016050  104401  001170          4$:     TYPE    ,$QUES          ;;TYPE A '?'
  (1)  016054  000763                          BR      1$              ;;CLEAR THE BUFFER AND LOOP
  (1)  016056  111337  016126          3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
  (1)  016062  104401  016126                  TYPE    ,9$
  (1)  016066  122723  000015                  CMPB    #15,(R3)+       ;;CHECK FOR RETURN
  (1)  016072  001356                          BNE     2$              ;;LOOP IF NOT RETURN
  (1)  016074  105063  177777                  CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
  (1)  016100  104401  001172                  TYPE    ,$LF            ;;TYPE A LINE FEED
  (1)  016104  012603                          MOV     (SP)+,R3        ;;RESTORE R3
  (1)  016106  011646                          MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
  (1)  016110  016666  000004  000002          MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
  (1)  016116  012766  016130  000004          MOV     #$TTYIN,4(SP)
  (1)  016124  000002                          RTI                     ;;RETURN
  (1)  016126     000                  9$:     .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
  (1)  016127     000                          .BYTE   0               ;;TERMINATOR
  (1)  016130  000010                  $TTYIN: .BLKB   8.              ;;RESERVE 8 BYTES FOR TTY INPUT
  (1)  016140  052536  005015     000  $CNTLU: .ASCIZ  /^U/<15><12>    ;;CONTROL 'U'
  (1)  016145     136  006507  000012  $CNTLG: .ASCIZ  /^G/<15><12>    ;;CONTROL 'G'
  (1)  016152  005015  053523  020122  $MSWR:  .ASCIZ  <15><12>/SWR = /
  (1)  016160  020075     000
  (1)  016163     040  047040  053505  $MNEW:  .ASCIZ  /  NEW = /
  (1)  016170  036440  000040
 1564                                  .SBTTL  TYPE ROUTINE
  (1)
  (2)                                  ;;**********************************************************************
  (1)                                  ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
  (1)                                  ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
  (1)                                  ;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
  (1)                                  ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
  (1)                                  ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
  (1)                                  ;*
  (1)                                  ;*CALL:
  (1)                                  ;*1) USING A TRAP INSTRUCTION
  (1)                                  ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
  (1)                                  ;*OR
  (1)                                  ;*      TYPE
  (1)                                  ;*      MESADR
  (1)                                  ;*
  (1)
  (1)  016174  105737  001157          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
  (1)  016200  100002                          BPL     1$              ;;BR IF YES
  (1)  016202  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
  (1)  016204  000430                          BR      3$              ;;LEAVE
  (1)  016206  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
  (1)  016210  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
  (1)  016214  122737  000001  001214          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
  (1)  016222  001011                          BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
  (1)  016224  132737  000100  001215          BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
  (1)  016232  001405                          BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
  (1)  016234  010037  016244                  MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
  (1)  016240  004737  016464                  JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
  (1)  016244  000000                  61$:    .WORD   0               ;;MESSAGE ADDRESS
  (1)  016246  132737  000040  001215  62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
  (1)  016254  001003                          BNE     60$             ;;YES,SKIP TYPE OUT
```

```
(1)  016256  112046                2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
(1)  016260  001005                        BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
(1)  016262  005726                        TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
(1)  016264  012600                60$:    MOV     (SP)+,R0        ;;RESTORE R0
(1)  016266  062716  000002        3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
(1)  016272  000002                        RTI                     ;;RETURN
(1)  016274  122716  000011        4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
(1)  016300  001430                        BEQ     8$
(1)  016302  122716  000200                CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
(1)  016306  001006                        BNE     5$
(1)  016310  005726                        TST     (SP)+           ;;POP  <CR><LF> EQUIV
(1)  016312  104401                        TYPE                    ;;TYPE A CR AND LF
(1)  016314  001171                        $CRLF
(1)  016316  105037  016452                CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
(1)  016322  000755                        BR      2$              ;;GET NEXT CHARACTER
(1)  016324  004737  016406        5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
(1)  016330  123726  001156        6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
(1)  016334  001350                        BNE     2$              ;;IF NO GO GET NEXT CHAR.
(1)  016336  013746  001154                MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
(1)                                                                 ;;AND THE NULL CHAR.
(1)  016342  105366  000001        7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
(1)  016346  002770                        BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
(1)  016350  004737  016406                JSR     PC,$TYPEC       ;;GO TYPE A NULL
(1)  016354  105337  016452                DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
(1)  016360  000770                        BR      7$              ;;LOOP
(1)
(1)                                ;HORIZONTAL TAB PROCESSOR
(1)
(1)  016362  112716  000040        8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
(1)  016366  004737  016406        9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
(1)  016372  132737  000007  016452        BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
(1)  016400  001372                        BNE     9$              ;;TAB STOP
(1)  016402  005726                        TST     (SP)+           ;;POP SPACE OFF STACK
(1)  016404  000724                        BR      2$              ;;GET NEXT CHARACTER
(1)  016406  105777  162536        $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
(1)  016412  100375                        BPL     $TYPEC
(1)  016414  116677  000002  162530        MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
(1)  016422  122766  000015  000002        CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
(1)  016430  001003                        BNE     1$              ;;BRANCH IF NO
(1)  016432  105037  016452                CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
(1)  016436  000406                        BR      $TYPEX          ;;EXIT
(1)  016440  122766  000012  000002  1$:    CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
(1)  016446  001402                        BEQ     $TYPEX          ;;BRANCH IF YES
(1)  016450  105227                        INCB    (PC)+           ;;COUNT THE CHARACTER
(1)  016452  000000                $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
(1)  016454  000207                $TYPEX: RTS     PC
(1)
1565                               .SBTTL  APT COMMUNICATIONS ROUTINE
(1)
(2)                                ;;**********************************************************************
(1)  016456  112737  000001  016722 $ATY1:   MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
(1)  016464  112737  000001  016720 $ATY3:   MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
(1)  016472  000403                        BR      $ATYC
(1)  016474  112737  000001  016722 $ATY4:   MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
(1)  016502                        $ATYC:
(3)  016502  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
```

```
(3)  016504  010146                        MOV     R1,-(SP)          ;;PUSH R1 ON STACK
(1)  016506  105737  016720                TSTB    $MFLG             ;;SHOULD TYPE A MESSAGE?
(1)  016512  001450                        BEQ     5$                ;;IF NOT:  BR
(1)  016514  122737  000001  001214        CMPB    #APTENV,$ENV      ;;OPERATING UNDER APT?
(1)  016522  001031                        BNE     3$                ;;IF NOT:  BR
(1)  016524  132737  000100  001215        BITB    #APTSPOOL,$ENVM   ;;SHOULD SPOOL MESSAGES?
(1)  016532  001425                        BEQ     3$                ;;IF NOT:  BR
(1)  016534  017600  000004                MOV     @4(SP),R0         ;;GET MESSAGE ADDR.
(1)  016540  062766  000002  000004        ADD     #2,4(SP)                  ;;BUMP RETURN ADDR.
(1)  016546  005737  001174          1$:   TST     $MSGTYPE          ;;SEE IF DONE W/ LAST XMISSION?
(1)  016552  001375                        BNE     1$                ;;IF NOT:  WAIT
(1)  016554  010037  001210                MOV     R0,$MSGAD         ;;PUT ADDR IN MAILBOX
(1)  016560  105720                  2$:   TSTB    (R0)+             ;;FIND END OF MESSAGE
(1)  016562  001376                        BNE     2$
(1)  016564  163700  001210                SUB     $MSGAD,R0         ;;SUB START OF MESSAGE
(1)  016570  006200                        ASR     R0                ;;GET MESSAGE LNGTH IN WORDS
(1)  016572  010037  001212                MOV     R0,$MSGLGT        ;;PUT LENGTH IN MAILBOX
(1)  016576  012737  000004  001174        MOV     #4,$MSGTYPE       ;;TELL APT TO TAKE MSG.
(1)  016604  000413                        BR      5$
(1)  016606  017637  000004  016632  3$:   MOV     @4(SP),4$         ;;PUT MSG ADDR IN JSR LINKAGE
(1)  016614  062766  000002  000004        ADD     #2,4(SP)                  ;;BUMP RETURN ADDRESS
(3)  016622  013746  177776                MOV     177776,-(SP)      ;;PUSH 177776 ON STACK
(1)  016626  004737  016174                JSR     PC,$TYPE          ;;CALL TYPE MACRO
(1)  016632  000000                  4$:   .WORD   0
(1)  016634                          5$:
(1)  016634  105737  016722          10$:  TSTB    $FFLG             ;;SHOULD REPORT FATAL ERROR?
(1)  016640  001416                        BEQ     12$               ;;IF NOT:  BR
(1)  016642  005737  001214                TST     $ENV              ;;RUNNING UNDER APT?
(1)  016646  001413                        BEQ     12$               ;;IF NOT:  BR
(1)  016650  005737  001174          11$:  TST     $MSGTYPE          ;;FINISHED LAST MESSAGE?
(1)  016654  001375                        BNE     11$               ;;IF NOT:  WAIT
(1)  016656  017637  000004  001176        MOV     @4(SP),$FATAL     ;;GET ERROR #
(1)  016664  062766  000002  000004        ADD     #2,4(SP)                  ;;BUMP RETURN ADDR.
(1)  016672  005237  001174                INC     $MSGTYPE          ;;TELL APT TO TAKE ERROR
(1)  016676  105037  016722          12$:  CLRB    $FFLG             ;;CLEAR FATAL FLAG
(1)  016702  105037  016721                CLRB    $LFLG             ;;CLEAR LOG FLAG
(1)  016706  105037  016720                CLRB    $MFLG             ;;CLEAR MESSAGE FLAG
(3)  016712  012601                        MOV     (SP)+,R1          ;;POP STACK INTO R1
(3)  016714  012600                        MOV     (SP)+,R0          ;;POP STACK INTO R0
(1)  016716  000207                        RTS     PC                ;;RETURN
(1)  016720     000                $MFLG:  .BYTE   0                 ;;MESSG. FLAG
(1)  016721     000                $LFLG:  .BYTE   0                 ;;LOG FLAG
(1)  016722     000                $FFLG:  .BYTE   0                 ;;FATAL FLAG
(1)          016724                         .EVEN
(1)          000200                APTSIZE=200
(1)          000001                APTENV=001
(1)          000100                APTSPOOL=100
(1)          000040                APTCSUP=040
1566                               .SBTTL  POWER DOWN AND UP ROUTINES
(1)
(2)                                ;;******************************************************************
(1)                                ;POWER DOWN ROUTINE
(1)  016724  012737  017064  000024 $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
(1)  016732  012737  000340  000026        MOV     #340,@#PWRVEC+2 ;;PRIO:7
(3)  016740  010046                        MOV     R0,-(SP)          ;;PUSH R0 ON STACK
(3)  016742  010146                        MOV     R1,-(SP)          ;;PUSH R1 ON STACK
```

```
(3)  016744  010246                        MOV     R2,-(SP)          ;;PUSH R2 ON STACK
(3)  016746  010346                        MOV     R3,-(SP)          ;;PUSH R3 ON STACK
(3)  016750  010446                        MOV     R4,-(SP)          ;;PUSH R4 ON STACK
(3)  016752  010546                        MOV     R5,-(SP)          ;;PUSH R5 ON STACK
(3)  016754  017746  162160                MOV     @SWR,-(SP)        ;;PUSH @SWR ON STACK
(1)  016760  010637  017070                MOV     SP,$SAVR6         ;;SAVE SP
(1)  016764  012737  016776  000024        MOV     #$PWRUP,@#PWRVEC  ;;SET UP VECTOR
(1)  016772  000000                        HALT
(1)  016774  000776                        BR      .-2               ;;HANG UP
(1)
(2)                                 ;;*****************************************************************
(1)                                 ;POWER UP ROUTINE
(1)  016776  012737  017064  000024 $PWRUP: MOV     #$ILLUP,@#PWRVEC  ;;SET FOR FAST DOWN
(1)  017004  013706  017070                MOV     $SAVR6,SP         ;;GET SP
(1)  017010  005037  017070                CLR     $SAVR6            ;;WAIT LOOP FOR THE TTY
(1)  017014  005237  017070        1$:     INC     $SAVR6            ;;WAIT FOR THE INC
(1)  017020  001375                        BNE     1$                ;;OF  WORD
(3)  017022  012677  162112                MOV     (SP)+,@SWR        ;;POP STACK INTO @SWR
(3)  017026  012605                        MOV     (SP)+,R5          ;;POP STACK INTO R5
(3)  017030  012604                        MOV     (SP)+,R4          ;;POP STACK INTO R4
(3)  017032  012603                        MOV     (SP)+,R3          ;;POP STACK INTO R3
(3)  017034  012602                        MOV     (SP)+,R2          ;;POP STACK INTO R2
(3)  017036  012601                        MOV     (SP)+,R1          ;;POP STACK INTO R1
(3)  017040  012600                        MOV     (SP)+,R0          ;;POP STACK INTO R0
(1)  017042  012737  016724  000024        MOV     #$PWRDN,@#PWRVEC  ;;SET UP THE POWER DOWN VECTOR
(1)  017050  012737  000340  000026        MOV     #340,@#PWRVEC+2   ;;PRIO:7
(1)  017056  104401                        TYPE                      ;REPORT THE POWER FAILURE
(1)  017060  017072        $PWRMG: .WORD   $POWER                    ;;POWER FAIL MESSAGE POINTER
(1)  017062  000002                        RTI
(1)  017064  000000        $ILLUP: HALT                              ;;THE POWER UP SEQUENCE WAS STARTED
(1)  017066  000776                        BR      .-2               ;; BEFORE THE POWER DOWN WAS COMPLETE
(1)  017070  000000        $SAVR6: 0                                 ;;PUT THE SP HERE
(1)  017072  005015  047520  042527 $POWER: .ASCIZ  <15><12>'POWER'
(1)  017100  000122
(1)                                        .EVEN
1567                                       ;*
1568                                       ;*THIS ROUTINE WILL PROTECT THE PROGRAM
1569                                       ;*FROM INTERUPTS (BAD ONES).
1570                                       ;*
1571                                       ;*THE TRAP CATCHER IS SET UP FOR
1572                                       ;*      .WORD   .+2
1573                                       ;*      JSR PC,R0
1574                                       ;*
1575                                       ;*ILLEGAL INTERRUPTS OR INTERRUPTS TO THE WRONG VECTOR
1576                                       ;*GOTO THE VECTOR AND PCITK UP THE ".+2" AS AN ADDRESS
1577
1578                                       ;*AND "4700" AS NEW STATUS.
1579                                       ;*THE .+2 AS A PC WILL CAUSE EXECUTION OF THE "JSR PC,R0" (AN ILLEGAL INSTR.).
1580                                       ;*AND TRAP TO LOCATION "4". IN LOCATION 4 WE HAVE A
1581                                       ;*POINTER HERE. IF THIS CONDITION CAUSES A TRAP TO LOC. 4.
1582                                       ;*WE WILL REPORT IT IN THE SAME MANNER THAT WER WOULD
1583                                       ;*REPORT ANY OTHER ERROR.
1584                                       ;*IF A BUSS ERROR TRAP DID OCCUT AND CAUSE A TRAP TO 4.
1585                                       ;*WE WILL HALT.
1586
1587  017102  011637  017146        IOTRD:  MOV     (6),TRTO          ;GET WHERE WE CAME TO.
```

```
1588  017106  162737  000004  017146          SUB     #4,TRTO         ;FORM READ ADDR.
1589
1590  017114  023727  017146  001000          CMP     TRTO,#1000      ;DID TRAP FROM LESS THAN ADDR. 1000?
1591  017122  003402                          BLE     2$              ;NO-CONTINUE.
1592
1593  017124  000000                  1$:     HALT                    ;A BUSS ERROR TIME OUT TRAP BROUGHT US HERE.
1594                                          ;ADDRESS CONTAINED IN TRTO.
1595
1596  017126  000776                          BR      1$              ;DON'T ALLOW CONTINUE.
1597
1598  017130  016637  000004  017150  2$:     MOV     4(6),TRFRO      ;GET TRAPPED FROM ADDR.
1599  017136  062706  000004                  ADD     #4,SP                   ;/ADD #4 TO STACK POINTER.
1600
                                      ;;$$$$$$$$$$$$$$$$$$$$$$$$$>>> ERROR <<<$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

1601  017142  104004                          ERROR   4               ;ERROR! ILLEGAL INTERRUPT OR
1602                                                                  ;INTERRUPT TO WRONG VECTOR.
1603                                                                  ;IF TEST NO. IS LESS THAN 10,ITS
1604                                                                  ;LIKELY(BUT NO EXCLUSIVELY)TO BE A
1605                                                                  ;DEVICE OTHER THAN THE DEVICE UNDER TEST.
1606                                                                  ;IF THE INTERRUPT OCCURED
1607                                                                  ;DURING AN INTERRUPT TEST, I'D
1608                                                                  ;SUSPECT A PROBLEM WITH THE DEVICE UNDER TEST.
1609                                                                  ;IF THE ADDRESS THE INTERRUPT
1610                                                                  ;VECTORED TO IS WITHIN THE RANGE OF
1611                                                                  ;VECTORS ASSIGNED TO THE DEVICE,
1612                                                                  ;THEN I'D SUSPECT THE DEVICE
1613                                                                  ;INTERRUPTD ILLEGALLY.
1614                                                                  ;IF THE ADDRESS THE INTERRUPT
1615                                                                  ;VECTORED TO IS OUTSIDE OF THE
1616                                                                  ;RANGE ASSIGNED TO THE DEVICE
1617                                                                  ;I'D SUSPECT THAT THE
1618                                                                  ;DEVICE PUT THE WRONG INTERRUPT
1619                                                                  ;VICTOR ON THE BUS DURING THE INTERRUPT
1620                                                                  ;PROCESS.
1621                                                                  ;       NOTE:
1622                                                                  ;FOR THIS ERROR - DON'T USE
1623                                                                  ; 'LOOP ON ERROR" OPTION.
1624                                                                  ;ALSO EXPECT THAT THE INTERRUPT TEST TO
1625                                                                  ;WILL REPOT THAT THE DEVICE DIDN'T
1626                                                                  ;INTERRUPT.
1627                                                                  ;FOLLOW THE RECOMMENDED PROCEEDURE
1628                                                                  ; IN THE DOCUMENT (ON THIS DIAGNOSTIC)
1629                                                                  ;FOR LOOPING ON TEST.
1630
1631

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

1632  017144  000002                          RTI
1633  017146  000000          TRTO:   .WORD   0               ;CONTAINS ADDR. WE TRAPPED OR INTERRUPTED TO.
1634  017150  000000          TRFRO:  .WORD   0               ;CONTAINS ADDR. WE TRAPPED OR INTR. FROM.
1635                          .SBTTL  TRAP DECODER
 (1)
 (2)                      ;;**************************************************************************
 (1)                      ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
 (1)                      ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
```

```
(1)                                     ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
(1)                                     ;*GO TO THAT ROUTINE.
(1)
(1)  017152  010046                     $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
(1)  017154  016600  000002                     MOV     2(SP),R0        ;;GET TRAP ADDRESS
(1)  017160  005740                             TST     -(R0)           ;;BACKUP BY 2
(1)  017162  111000                             MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
(1)  017164  006300                             ASL     R0              ;;POSITION FOR INDEXING
(1)  017166  016000  017206                     MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
(1)  017172  000200                             RTS     R0              ;;GO TO ROUTINE
(1)
(1)
(1)                                     ;;THIS IS USE TO HANDLE THE ''GETPRI'' MACRO
(1)
(1)  017174  011646                     $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
(1)  017176  016666  000004  000002             MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
(1)  017204  000002                             RTI                     ;;RESTORE THE PSW
(1)
(3)                                     .SBTTL  TRAP TABLE
(3)
(3)                                     ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
(3)                                     ;*BY THE ''TRAP'' INSTRUCTION.
(3)
(3)                                     ;       ROUTINE
(3)                                     ;       -------
(3)  017206  017174                     $TRPAD: .WORD   $TRAP2
(3)  017210  016174                             $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
(3)  017212  014316                             $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
(3)  017214  014272                             $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
(3)  017216  014332                             $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
(3)  017220  014520                             $TYPBN  ;;CALL=TYPBN    TRAP+5(104405)  TYPE BINARY (ASCII) NUMBER
(1)
(3)  017222  015470                             $GTSWR  ;;CALL=GTSWR    TRAP+6(104406)  GET SOFT-SWR SETTING
(1)
(3)  017224  015420                             $CKSWR  ;;CALL=CKSWR    TRAP+7(104407)  TEST FOR CHANGE IN SOFT-SWR
(3)  017226  015702                             $RDCHR  ;;CALL=RDCHR    TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
(3)  017230  016022                             $RDLIN  ;;CALL=RDLIN    TRAP+11(104411) TTY TYPEIN STRING ROUTINE
1636 017232  005015  046103  041517  EM1:      .ASCIZ  <15><12>/CLOCK SR FUNCTION ERROR/
     017240  020113  051123  043040
     017246  047125  052103  047511
     017254  020116  051105  047522
     017262  000122
1637 017264  005015  046103  041517  EM2:      .ASCIZ  <15><12>/CLOCK SR DATA ERROR/
     017272  020113  051123  042040
     017300  052101  020101  051105
     017306  047522  000122
1638 017312  005015  046103  041517  EM3:      .ASCIZ  <15><12>/CLOCK BR DATA ERROR/
     017320  020113  051102  042040
     017326  052101  020101  051105
     017334  047522  000122
1639 017340  044600  052116  051105  EM4:      .ASCIZ  <200>/INTERRUPT ERROR/
     017346  052522  052120  042440
     017354  051122  051117     000
1640 017361     015  041412  052517  EM5:      .ASCIZ  <15><12>/COUNT REG. ERROR/
     017366  052116  051040  043505
     017374  020056  051105  047522
```

```
                017402  000122
1641            017404  005015  047503  047125  EM11:   .ASCIZ  <15><12>#COUNT ERROR #
                017412  020124  051105  047522
                017420  020122  000
1642            017423  015     041412  052517  EM12:   .ASCIZ  <15><12>#COUNT FUNCTION ERROR#
                017430  052116  043040  047125
                017436  052103  047511  020116
                017444  051105  047522  000122
1643            017452  005015  046103  041517  EM16:   .ASCIZ  <15><12>#CLOCK INTERRUPT ERROR #
                017460  020113  047111  042524
                017466  051122  050125  020124
                017474  051105  047522  020122
                017502  000
1644            017503  015     051012  050105  EM20:   .ASCIZ  <15><12>#REPEATABILITY ERROR #
                017510  040505  040524  044502
                017516  044514  054524  042440
                017524  051122  051117  000040
1645            017532  005015  042101  051104  EM26:   .ASCIZ  <15><12>#ADDRESSING ERROR#
                017540  051505  044523  043516
                017546  042440  051122  051117
                017554  000
1646
1647            017555  015     042412  051122  DH1:    .ASCIZ  <15><12>#ERRPC  ASR     WAS     S/B#
                017562  041520  040411  051123
                017570  053411  051501  051411
                017576  041057  000
1648            017601  015     042412  051122  DH3:    .ASCIZ  <15><12>#ERRPC  ABR     WAS     S/B#
                017606  041520  040411  051102
                017614  053411  051501  051411
                017622  041057  000
1649            017625  200     051105  050122  DH4A:   .ASCIZ  <200>#ERRPC   TO      FROM ADDR.#
                017632  020103  020040  047524
                017640  020040  020040  020040
                017646  051106  046517  040440
                017654  042104  027122  000
1650            017661  015     042412  051122  DH12:   .ASCIZ  <15><12>#ERRPC  ASR     #
                017666  041520  040411  051123
                017674  000011
1651            017676  005015  051105  050122  DH20:   .ASCIZ  <15><12>#ERRPC  ASR     2NDCNT  1STNCT  3RDCNT#
                017704  004503  051501  004522
                017712  047062  041504  052116
                017720  030411  052123  041516
                017726  004524  051063  041504
                017734  052116  000
1652            017737  015     042412  051122  DH26:   .ASCIZ  <15><12>#ERRPC  CLOCK ADDR.#
                017744  041520  041411  047514
                017752  045503  040440  042104
                017760  027122  000
1653
1654                    017764                  .EVEN
1655
1656            017764  001116  001376  001126  DT1:    .WORD   $ERRPC,ASR,$BDDAT,$GDDAT,0
                017772  001124  000000
1657            017776  001116  001400  001126  DT3:    .WORD   $ERRPC,ABR,$BDDAT,$GDDAT,0
                020004  001124  000000
1658            020010  001116  017146  017150  DT4:    .WORD   $ERRPC,TRTO,TRFRO,0
```

```
                020016  000000
1659            020020  001116  001376  000000  DT12:   .WORD   $ERRPC,ASR,0
1660            020026  001116  001376  001126  DT20:   .WORD   $ERRPC,ASR,$BDDAT,$GDDAT,$TMP0,0
                020034  001124  001420  000000
1661            020042  001116  001376  001126  DT22:   .WORD   $ERRPC,ASR,$BDDAT,$TMP0,0
                020050  001420  000000
1662            020054  001116  001420  000000  DT26:   .WORD   $ERRPC,$TMP0,0
1663
1664            020062  000000  000000          DF0:    .WORD   0,0
1665
1666                    000001                          .END
```

```
ABASE = 170420          102#  112   189   190
ABR     001400          190#  256*  257*  295   385*  387*  463*  471   489*  497   559*  563   626*
                        630   650*  668   678*  694*  724*  744*  806*  808*  810*  812*  814*  816*
                        822*  832   848*  881*  1021* 1063* 1071  1082* 1093  1172* 1173* 1174* 1176*
                        1177* 1201* 1211  1220  1241* 1251  1261  1287* 1292  1304  1318* 1323  1462*
                        1467  1490* 1657
ACDW1 = 000000          112
ACDW2 = 000000          112
ACPUOP= 000000          112
ADDW0 = 000000          112
ADDW1 = 000000          112
ADDW10= 000000          112
ADDW11= 000000          112
ADDW12= 000000          112
ADDW13= 000000          112
ADDW14= 000000          112
ADDW15= 000000          112
ADDW2 = 000000          112
ADDW3 = 000000          112
ADDW4 = 000000          112
ADDW5 = 000000          112
ADDW6 = 000000          112
ADDW7 = 000000          112
ADDW8 = 000000          112
ADDW9 = 000000          112
ADEVCT= 000000          112
ADEVM = 000000          112
AENV  = 000000          112
AENVM = 000000          112
AFATAL= 000000          112
AMADR1= 000000          112
AMADR2= 000000          112
AMADR3= 000000          112
AMADR4= 000000          112
AMAMS1= 000000          112
AMAMS2= 000000          112
AMAMS3= 000000          112
AMAMS4= 000000          112
AMSGAD= 000000          112
AMSGLG= 000000          112
AMSGTY= 000000          112
AMTYP1= 000000          112
AMTYP2= 000000          112
AMTYP3= 000000          112
AMTYP4= 000000          112
ANYKEY  013740          1290  1303  1321  1417#
APASS = 000000          112
APRIOR= 000200          104#  112   195
APTCSU= 000040          1564  1565#
APTENV= 000001          1560  1564  1565#
APTSIZ= 000200          225   1565#
APTSPO= 000100          1564  1565#
ASR     001376          189#  232*  256   294   342*  343*  344*  345*  346*  347*  348*  349*  350*
                        351*  406*  407*  412   432*  434*  438*  444*  446   462*  467*  471*  488*
                        493*  497*  522*  526   580*  581*  583   605*  606*  607   616*  625*  627*
                        628*  630*  649*  654*  656*  668*  677*  693*  696*  698*  700   722*  726*
```

```
                    728*  732   743*  745*  747*  749   756*  806*  808*  810*  812*  814*  816*
                    821*  824*  826*  832*  834   842*  847*  849*  854   880*  883*  884*  895*
                    901*  903*  904*  914*  1002* 1003* 1004* 1005* 1007  1010  1020* 1022* 1028
                   1031   1040* 1041* 1042* 1044* 1046* 1049  1058* 1062* 1066* 1067* 1071* 1077*
                   1081*  1083* 1084* 1087* 1093* 1100* 1172* 1173* 1174* 1176* 1177* 1200* 1202*
                   1205*  1209* 1220* 1240* 1242* 1245* 1249* 1261* 1270* 1271  1286* 1288* 1292*
                   1304*  1317* 1319* 1323* 1333  1358* 1362  1389* 1461* 1463* 1464* 1465* 1466*
                   1489*  1491* 1492* 1495  1530* 1531* 1532  1536  1656  1659  1660  1661
ASWREG= 000000      112
ATESTN= 000000      112
AUNIT = 000000      112
AUSWR = 000000      112
AVECT1= 000440      103#  112   191   192   193   194
AVECT2= 000000      112
BIT0  = 000001      100#  351   467   471   493   497   627   630   654   668   696   726   732
                    749   806   808   810   812   814   816   824   832   903   1011  1040  1044
                   1046   1067  1071  1084  1093  1172  1173  1174  1176  1177  1220  1261  1292
                   1304   1323
BIT00 = 000001      100#  607
BIT01 = 000002      100#
BIT02 = 000004      100#
BIT03 = 000010      100#
BIT04 = 000020      100#
BIT05 = 000040      100#
BIT06 = 000100      100#
BIT07 = 000200      100#
BIT08 = 000400      100#  1562
BIT09 = 001000      100#  1560  1562
BIT1  = 000002      100#  350   660
BIT10 = 002000      100#  540   575   1087  1172  1173  1174  1176  1177  1205  1245  1560
BIT11 = 004000      100#  344   471   497   540   575   630   668   806   808   810   812   814
                    816   832   1066  1071  1083  1093  1172  1173  1174  1176  1177  1220  1261
                   1292   1304  1323  1562
BIT12 = 010000      100#  540   575   816   1007  1011  1028  1052  1536
BIT13 = 020000      100#  343   605   1560
BIT14 = 040000      100#  342   903   1562
BIT15 = 100000      100#  1011
BIT2  = 000004      100#  349   471   497   594   630   668   806   808   810   812   814   816
                    832   1071  1093  1172  1173  1174  1176  1177  1220  1261  1292  1304  1323
BIT3  = 000010      100#  348   1067  1084
BIT4  = 000020      100#  347   627   654   696   726
BIT5  = 000040      100#  346   627   654   690   696   726
BIT6  = 000100      100#  345
BIT7  = 000200      100#
BIT8  = 000400      100#  628   656   698   728   747   826   884   1492  1531
BIT9  = 001000      100#  471   497   581   606   630   668   806   808   810   812   814   816
                    832   904   1004  1005  1041  1042  1071  1093  1172  1173  1174  1176  1177
                   1209   1220  1249  1261  1292  1304  1323  1464  1466
BPTVEC= 000014      100#
CKSWR = 104407     1460   1488  1529  1560  1562  1635#
CR    = 000015      100#  1564
CRLF  = 000200      100#  234   1564
DDISP = 177570      100#  112   225
DF0     020062      124   131   138   145   152   159   166   173   180   187   1664#
DH1     017555      122   129   150   178   1647#
DH12    017661      157   164   1650#
```

```
DH20     017676            171    1651#
DH26     017737            185    1652#
DH3      017601            136    1648#
DH4A     017625            143    1649#
DISPLA   001142            112#    225*  1560*  1562*
DISPRE   000174             78#    225
DR       001414            196#    541    594    660
DR2      001416            197#    540*   575*
DSWR   = 177570            100#    112    225
DT1      017764            123     130    151   1656#
DT12     020020            158     165   1659#
DT20     020026            172    1660#
DT22     020042            179    1661#
DT26     020054            186    1662#
DT3      017776            137    1657#
DT4      020010            144    1658#
EMTVEC=  000030            100#    225*
EM1      017232            121    1636#
EM11     017404            177    1641#
EM12     017423            156    1642#
EM16     017452            163    1643#
EM2      017264            128    1637#
EM20     017503            170    1644#
EM26     017532            184    1645#
EM3      017312            135    1638#
EM4      017340            142    1639#
EM5      017361            149    1640#
ENDP     013056           1283    1315   1330   1334   1346#
ERCNT    001432            203#    241*  1410   1560*
ERRVEC=  000004            100#    225*   230*   294*   295*  1359  1360*  1375*  1380*  1562*
EXS      001440            206#    214*   219*   223*   236   538   573   592   658   685   707  1281  1314
GNS    = ****** U          234     238    239   1289   1302  1320  1369  1371  1373  1385  1387  1410  1418
                          1635
GTSWR  = 104406            234    1635#
HT     = 000011            100#   1564
INIT     001526            212     225#  1459   1487   1528
IOTRD    017102             76     230   1587#
IOTST1   014060           1458    1460#  1468   1470
IOTST2   014150           1486    1488#  1496   1498
IOTST3   014230           1527    1529#  1537   1539
IOTVEC=  000020            100#    225*
LCNT     001442            207#   1391*  1392*  1410
LF     = 000012            100#   1564
LOOP     002334            245#   1377   1410
MDEVCT   001434            204#    240*  1353   1356*  1395*
OITST1   014046             88    1458#
OITST2   014136             89    1486#
OITST3   014216             90    1527#
PIRQ   = 177772            100#
PIRQVE=  000240            100#
PRIOR    001412            195#
PR0    = 000000            100#
PR1    = 000040            100#
PR2    = 000100            100#
PR3    = 000140            100#
PR4    = 000200            100#
```

```
PR5    = 000240            100#
PR6    = 000300            100#
PR7    = 000340            100#
PS     = 177776            100#
PSW    = 177776            100#
PWRVEC= 000024             100#    225*   1566*
QSTART  001444              87     211#
RDCHR  = 104410            1563    1635#
RDLIN  = 104411            1635#
RESVEC= 000010             100#
ROTATE  001426             201#    243*    244    1355*   1367    1396*   1560
RSTART  002144             211     235#
STACK  = 001100            100#    225
START  = 001506             86     221#
STKLMT= 177774             100#
SWR     001140             112#    225*    234     816    1560    1562    1563*   1566*
SWREG   000176              79#    225     234    1563
SW0    = 000001            100#
SW00   = 000001            100#
SW01   = 000002            100#
SW02   = 000004            100#
SW03   = 000010            100#
SW04   = 000020            100#
SW05   = 000040            100#
SW06   = 000100            100#
SW07   = 000200            100#
SW08   = 000400            100#
SW09   = 001000            100#
SW1    = 000002            100#
SW10   = 002000            100#
SW11   = 004000            100#
SW12   = 010000            100#
SW13   = 020000            100#
SW14   = 040000            100#
SW15   = 100000            100#
SW2    = 000004            100#
SW3    = 000010            100#
SW4    = 000020            100#
SW5    = 000040            100#
SW6    = 000100            100#
SW7    = 000200            100#
SW8    = 000400            100#
SW9    = 001000            100#
TBITVE= 000014             100#
TKVEC  = 000060            100#
TPVEC  = 000064            100#
TRAPVE= 000034             100#    225*
TRFRO   017150            1598*   1634#   1658
TRTO    017146            1587*   1588*   1590    1633#   1658
TRTVEC= 000014             100#
TSTCNT  001436             205#    215*    218*    222*   1353
TSTSTR  001456              95     214#
TST1    002422             294#
TST10   003244             347#
TST11   003342             348#
TST12   003440             349#
```

```
TST13   003536              350#
TST14   003634              351#
TST15   003732              385#
TST16   004022              387#
TST17   004112              404#
TST2    002514              295#
TST20   004164              430#
TST21   004246              460#
TST22   004374              486#
TST23   004522              519#
TST24   004610              536     539     542     556#
TST25   004662              574     578#
TST26   004732              591     593     595     603#
TST27   004766              623#
TST3    002556              342#
TST30   005076              647#
TST31   005306              666     683#
TST32   005422              706     708     711     720#
TST33   005464              741#
TST34   005532              806#
TST35   005662              808#
TST36   006012              810#
TST37   006142              812#
TST4    002654              343#
TST40   006272              814#
TST41   006422              816#
TST42   006562              816     819#
TST43   006724              845#
TST44   006764              876#
TST45   007104              897#
TST46   007210             1000#
TST47   007264             1018#
TST5    002752              344#
TST50   007350             1038#
TST51   007442             1060#
TST52   007576             1079#
TST53   007730             1172#
TST54   010222             1173#
TST55   010514             1174#
TST56   011006             1176#
TST57   011300             1177#
TST6    003050              345#
TST60   011612             1198#
TST61   012004             1238#
TST62   012220             1258    1268    1272    1278#
TST63   012620             1299    1305    1312#
TST7    003146              346#
TYPBN = 104405             1410    1635#
TYPE  = 104401              234     238     239     1289    1302    1320    1369    1371    1373    1385    1387    1410    1418
                           1546    1547    1560    1561    1563    1564    1566    1635#
TYPOC = 104402             1370    1374    1386    1410    1561    1563    1635#
TYPON = 104404             1635#
TYPOS = 104403             1635#
UTEST   001430              202#    244*    1367*   1410    1560*
VECTP   001404              192#    248*    249*
VECT1   001402              191#    231*    247*    248     250     885*    1376*   1390*
```

```
VECT2   001406          193#    250*    251*    252     902*
VECT2P  001410          194#    252*    253*
WSTART= 001472           93     217#
ZSTART  001774          224     228#
$APTHD  001000          111#
$ASTAT= ****** U        1565
$ATYC   016502          1565#
$ATY1   016456          1565#
$ATY3   016464          1564    1565#
$ATY4   016474          1560    1565#
$AUTOB  001134          112#    234*    1563
$BASE   001250          112#    232     1389
$BDADR  001122          112#
$BDDAT  001126          112#    342*    343*    344*    345*    346*    347*    348*    349*    350*    351*    385*    387*
                        412*    414     417     446*    447     449     471*    473     497*    499     526*    563*    630*
                        668*    670     806*    808*    810*    812*    814*    816*    832*    1010*   1031*   1049*   1052
                        1071*   1093*   1172*   1173*   1174*   1176*   1177*   1211*   1212    1220*   1221    1251*   1252
                        1261*   1271*   1292*   1304*   1323*   1656    1657    1660    1661
$BELL   001164          112#    1560
$BIN    014572          1547#*
$CDW1   001254          112#
$CHARC  016452          1564#*
$CKSWR  015420          1563#   1635
$CMTAG  001100          112#    225
$CM3  = 000000          112#
$CNTLG  016145          1563#
$CNTLU  016140          1563#
$CPUOP  001222          112#
$CRLF   001171          112#    1560    1561    1563    1564
$DEVCT  001204          112#    229*    1370    1372*   1374    1381    1386    1391    1393*
$DEVM   001252          112#
$DOAGN  013730          1410#
$ENDAD  013720          109     1410#
$ENDCT  013466          1410#
$ENULL  013734          1410#
$ENV    001214          112#    234     1560    1564    1565
$ENVM   001215          112#    225     1349    1564    1565
$EOP    013432          1410#
$EOPCT  013460          1410#
$ERFLG  001103          112#    1560*   1562*
$ERMAX  001115          112#    225*    1562*
$ERROR  014574          225     1560#
$ERRPC  001116          112#    1560*   1561    1656    1657    1658    1659    1660    1661    1662
$ERRTB  001256          112#    1561
$ERRTY  015002          1560    1561#
$ERTTL  001112          112#    1560*
$ESCAP  001162          112#    225*    1560    1562*
$ETABL  001214          112#
$ETEND  001256          111     112#
$FATAL  001176          112#    1565*
$FFLG   016722          1565#*
$FILLC  001156          112#    1564
$FILLS  001155          112#    1564
$GDADR  001120          112#
$GDDAT  001124          112#    342*    343*    344*    345*    346*    347*    348*    349*    350*    351*    385*    387*
                        414*    415*    447*    448*    469*    473     495*    499     521*    558*    651*    669*    670
```

```
                        678     679     837*    1011*   1032*   1051*   1064*   1172*   1173*   1174*   1176*   1177*   1210*
                       1212    1221    1250*    1252    1260*   1273*   1291*   1322*   1656    1657    1660
$GET42  013710         1410#
$GTSWR  015470         1563#   1635
$HD   = 000001           54
$HIBTS  001000          111#
$ICNT   001104          112#   1562*
$ILLUP  017064         1566#
$INTAG  001135          112#   1563
$ITEMB  001114          112#   1560*   1561
$LF     001172          112#   1560    1563    1564
$LFLG   016721         1565#*
$LPADR  001106          112#    225*    294*   1562*
$LPERR  001110          112#    225*    294*    652*   1560    1562*
$MADR1  001226          112#
$MADR2  001232          112#
$MADR3  001236          112#
$MADR4  001242          112#
$MAIL   001174          111     112#    225     234    1560    1562    1564
$MAMS1  001224          112#
$MAMS2  001230          112#
$MAMS3  001234          112#
$MAMS4  001240          112#
$MBADR  001002          111#
$MFLG   016720         1565#*
$MNEW   016163         1563#
$MSGAD  001210          112#   1565*
$MSGLG  001212          112#   1565*
$MSGTY  001174          112#   1565*
$MSWR   016152         1563#
$MTYP1  001225          112#
$MTYP2  001231          112#
$MTYP3  001235          112#
$MTYP4  001241          112#
$MXCNT  015416         1562#
$NULL   001154          112#   1564
$NWTST= 000001          294#    295#    342#    343#    344#    345#    346#    347#    348#    349#    350#    351#    385#
                        387#    404#    430#    460#    486#    519#    556#    578#    603#    623#    647#    683#    720#
                        741#    806#    808#    810#    812#    814#    816#    819#    845#    876#    897#   1000#   1018#
                       1038#   1060#   1079#   1172#   1173#   1174#   1176#   1177#   1198#   1238#   1278#   1312#
$OCNT   014514         1546#*
$OMODE  014516         1546#*
$OVER   015402         1562#
$PASS   001202          112#    225*    242*   1365    1410*   1562
$PASTM  001006          111#
$POWER  017072         1566#
$PWRDN  016724          225    1566#
$PWRMG  017060         1566#
$PWRUP  016776         1566#
$QUES   001170          112#   1560    1563    1564
$RDCHR  015702         1563#   1635
$RDDEC= ****** U       1635
$RDLIN  016022         1563#   1635
$RDOCT= ****** U       1635
$RDSZ = 000010         1563#
$RTNAD  013732         1410#
```

```
$R2A   = ****** U          1635
$SAVRE= ****** U           1635
$SAVR6  017070             1566#*
$SCOPE  015136              225    1562#
$SETUP= 000117              210#    225     234     1410    1560    1562    1563
$STUP  = 177777             210#
$SVLAD  015346             1562#
$SVPC  = 000244             109#
$SWR   = 167400               9#     54      56     106#    112     225     294     295     342     343     344     345     346
                            347     348     349     350     351     385     387     404     430     460     486     519     556
                            578     603     623     647     683     720     741     806     808     810     812     814     816
                            819     845     876     897    1000    1018    1038    1060    1079    1172    1173    1174    1176
                           1177    1198    1238    1278    1312    1410    1560    1562    1566
$SWREG  001216              112#    225
$SWRMK= 000000               56    1562
$TESTN  001200              112#   1562*
$TIMES  001160              112#    225*    294*    342*    343*    344*    345*    346*    347*    348*    349*    350*    351*
                            404*    430*    460*    486*    519*    556*    647*    806*    808*    810*    812*    814*    816*
                            819*    876*   1018*   1060*   1172*   1173*   1174*   1176*   1177*   1198*   1238*   1280*   1312*
                           1410*   1562*
$TKB    001146              112#   1417    1422    1563
$TKS    001144              112#   1420    1563*
$TMP0   001420              198#    211*    224*    226     294*    295*   1172*   1173*   1174*   1176*   1177*   1458*   1486*
                           1527*   1660    1661    1662
$TMP1   001422              199#
$TMP3   001424              200#    471*    497*    630*    668*    806*    808*    810*    812*    814*    816*    832*   1071*
                           1093*   1172*   1173*   1174*   1176*   1177*   1220*   1261*   1292*   1304*   1323*
$TN    = 000064              54#    107#    294#    295#    342#    343#    344#    345#    346#    347#    348#    349#    350#
                            351#    385#    387#    404#    430#    460#    486#    519#    536     539     542     556#    574
                            578#    591     593     595     603#    623#    647#    666     683#    706     708     711     720#
                            741#    806#    808#    810#    812#    814#    816#    819#    845#    876#    897#   1000#   1018#
                           1038#   1060#   1079#   1172#   1173#   1174#   1176#   1177#   1198#   1238#   1258    1268    1272
                           1278#   1299    1305    1312#
$TPB    001152              112#   1564*
$TPFLG  001157              112#   1564
$TPS    001150              112#   1564
$TRAP   017152              225    1635#
$TRAP2  017174             1635#
$TRP   = 000012            1635#
$TRPAD  017206             1635#
$TSTM   001004              111#
$TSTNM  001102              112#    294*   1410*   1560    1562*
$TTYIN  016130             1563#
$TYPBN  014520             1547#   1635
$TYPDS= ****** U           1635
$TYPE   016174             1564#   1565    1635
$TYPEC  016406             1563    1564#
$TYPEX  016454             1564#
$TYPOC  014316             1546#   1635
$TYPON  014332             1546#   1635
$TYPOS  014272             1546#   1635
$UNIT   001206              112#
$UNITM  001010              111#
$USWR   001220              112#
$VECT1  001244              112#    231     1390
$VECT2  001246              112#
```

```
$XTSTR  015152            1562#
$$GET4= 000000            1410#
$OFILL  014515            1546#*
$40CAT= ****** U          1560    1562
.       = 020066           60#     71      72#     75#     77#     80#     85#     92#     94#     109#    110#    111#    112#
                          217     221     225     238#    1289#   1302#   1320#   1369#   1385#   1410#   1418#   1560    1561#
                          1562    1563#   1564    1565#   1566    1654#
.$ASTA= ****** U          1565
.$X     = 001000           111#
```

```
ADTST    258#  294   295
BUFLO    353#  385   387
COMMEN   100#
COUNTM   758#  806   808   810   812   814   816
CSRDTA   304#  342   343   344   345   346   347   348   349   350   351
DFC      114#  124   131   138   145   152   159   166   173   180   187
DIVCH   1102#  1172  1173  1174  1176  1177
ECB      12#   294   295   342   343   344   345   346   347   348   349   350   351   385   387
         421   426   452   456   477   482   503   508   530   535   543   546   567   571   586
         590   596   599   610   615   632   635   662   665   672   674   703   714   717   735
         751   754   806   808   810   812   814   816   838   841   857   860   890   892   909
         911   1012  1015  1033  1036  1054  1057  1073  1076  1096  1099  1172  1173  1174  1176
        1177   1214  1217  1223  1227  1254  1257  1263  1267  1274  1276  1294  1298  1306  1310
        1326   1329  1335  1338  1600  1631
ENDCOM   100#
ENDPAS  1397#  1410
ERROR    100#  294   295   342   343   344   345   346   347   348   349   350   351   385   387
         422   453   478   504   532   544   568   587   597   611   633   663   673   704   715
         736   752   806   808   810   812   814   816   839   858   891   910   1013  1034  1055
        1074   1097  1172  1173  1174  1176  1177  1215  1224  1255  1264  1275  1295  1307  1327
        1336   1469  1497  1534  1538  1601
ESCAPE   100#
GETPRI   100#
GETSWR   100#  234#
INSTR2  1549#  1560
LOCKM    868#  878   886   889   899   905   908
MULT     100#
NEWTST   100#  294   295   342   343   344   345   346   347   348   349   350   351   385   387
         404   430   460   486   519   556   578   603   623   647   683   720   741   806   808
         810   812   814   816   819   845   876   897   1000  1018  1038  1060  1079  1172  1173
        1174   1176  1177  1198  1238  1278  1312
POP      100#  1565  1566
PGPSP2    24#  294   295   894   913   1379  1599
PR        28#  228
PUSH     100#  1565  1566
RDCLK     37#  471   497   630   668   806   808   810   812   814   816   832   1071  1093  1172
        1173   1174  1176  1177  1220  1261  1292  1304  1323
REPORT   100#
SCOPE    100#  295   342   343   344   345   346   347   348   349   350   351   385   387   404
         430   460   486   519   556   578   603   623   647   683   720   741   806   808   810
         812   814   816   819   845   876   897   1000  1018  1038  1060  1079  1172  1173  1174
        1176   1177  1198  1238  1278  1312  1346
SETPRI   100#
SETTRA  1635#
SETUP    100#  225
SKIP     100#  536   539   542   574   591   593   595   666   706   708   711   816   1258  1268
        1272   1299  1305
SLASH    100#
SPACE    100#
STARS    100#  109   111   112   294   295   342   343   344   345   346   347   348   349   350
         351   385   387   404   430   460   486   519   556   578   603   623   647   683   720
         741   806   808   810   812   814   816   819   845   876   897   1000  1018  1038  1060
        1079   1172  1173  1174  1176  1177  1198  1238  1278  1312  1410  1546  1547  1560  1561
        1562   1563  1564  1565  1566  1635
SWRSU    100#  225#
TRMTRP  1635#
```

```
TYPBIN    100#  1410
TYPDEC    100#
TYPNAM    100#   234
TYPNUM    100#
TYPOCS      6#   100#
TYPOCT    100#  1370  1374  1386  1410  1561  1563
TYPTXT    100#   238   239  1289  1302  1320  1369  1371  1373  1385  1387  1410  1418
ZIOTM    1424#  1448  1476
ZP21      394#   404   430
ZP25      511#   519
ZP25A     547#   556
ZZ1       296#   342   343   344   345   346   347   348   349   350   351   638#  647   806#  808#
          810#   812#  814#  816#  1179#  1198  1230#  1238
$$CMRE    112#
$$CMTM    112#
$$ESCA    100#
$$NEWT    100#   294   295   342   343   344   345   346   347   348   349   350   351   385   387
          404    430   460   486   519   556   578   603   623   647   683   720   741   806   808
          810    812   814   816   819   845   876   897  1000  1018  1038  1060  1079  1172  1173
         1174   1176  1177  1198  1238  1278  1312
$$SET    1635#
$$SETM    225#
$$SKIP    100#   536   539   542   574   591   593   595   666   706   708   711   816  1258  1268
         1272   1299  1305
.EQUAT      6#   100
.HEADE      5#    54
.SETTR      5#
.SETUP      5#   210
.SWRHI      6#    56
.SWRLO     56#
.TRMTR      5#
.$ACT1      8#   109
.$APTB    112#
.$APTH      8#   111
.$APTY      8#  1565
.$CATC      6#
.$CMTA      6#   112
.$EOP       7#  1410
.$ERRO      7#  1560
.$ERRT      7#  1561
.$POWE      6#  1566
.$RDOC      5#
.$READ      7#  1563
.$SCOP      7#  1562
.$TRAP      5#  1635
.$TYPB      5#  1547
.$TYPD      7#
.$TYPE      7#  1564
.$TYPO      6#  1546

. ABS.  020066     000     OVR   RO    REL   LCL   I

 ERRORS DETECTED:  0
```

```
CVKWAC,CVKWAC/CRF=CVKWAC
RUN-TIME: 28 17 1 SECONDS
RUN-TIME RATIO: 78/47=1.6
CORE USED:  27K  (53 PAGES)
```