KW11-L

LINE TIME CLOCK

MD-11-DZKWA-D

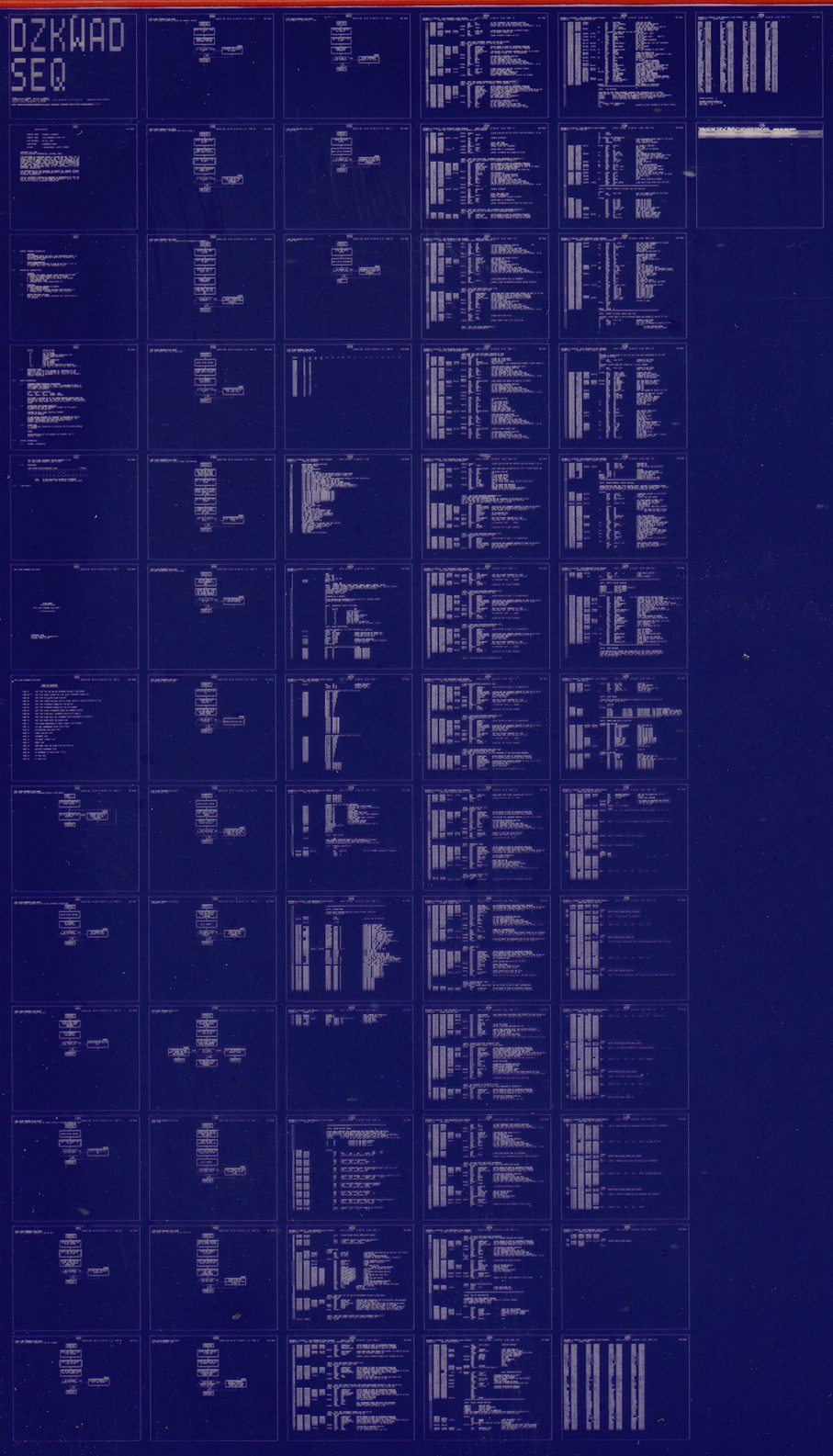
EP-DZKWA-D-DL-B

SEP 1975

digital

Made In U.S.A.

FICHE 1 OF 1



B01

DZKWAD

SEQ

LPTSPL Version 6(434) Running on MTB401
START User BABE [1,2] Job DZKWAD Seq. 1 Date 26-Sep-75 14:19:14 Monitor RS212b SYS #40/2 *START*
Request created: 26-Sep-75 14:19:12

[illegible] 14:19:15

IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DZKWA-D

PRODUCT NAME: LINE FREQUENCY CLOCK TEST

DATE REVISED: MAY 20, 1975

MAINTAINER: DIAGNOSTIC GROUP

AUTHOR: J RODENHISER/J LACEY/J COMEAU

1.0 GENERAL PROGRAM INFORMATION

1.1 ABSTRACT
THIS PROGRAM TESTS THE KW11L LINE FREQUENCY CLOCK. IT VALIDATES PROPER OPERATION UNDER BOTH INTERRUPT AND NON-INTERRUPT MODES.

1.2 SYSTEM REQUIREMENTS
THIS PROGRAM IS DESIGNED TO RUN ON ANY PDP-11 WITH 4K OF MEMORY AND A KW11 LINE FREQUENCY CLOCK.

2.0 OPPERATING INSTRUCTIONS

2.1 LOADING
PROCEDURE FOR NORMAL BINARY TAPES SHOULD BE FOLLOWED
A ABSOLUTE LOADER PROGRAM MUST BE IN MEMORY
B PLACE THE BINARY TAPE IN THE PAPER TAPE READER
C LOAD ADDRESS 17500
D DEPRESS START (TAPE SHOULD READ IN)

2.2 STARTING
PROGRAM STARTING ADDRESS IS 000200
A LOAD ADDRESS 000200
B SELECT SWITCH REGISTER OPTIONS (SEE SECTION 2.3)
C DEPRESS START (PROGRAM SHOULD START RUNNING)

2.3 SWITCH REGISTER OPTIONS
HERE IS A LIST OF CONSOLE SWITCHES AND THEIR EFFECT ON THE PROGRAM...

SWITCH	ACTION IF SET
15	HALT ON ERROR
14	LOOP ON CURRENTLY EXECUTING TEST
13	INHIBIT ERROR PRINTOUTS
12	(UNUSED)
11	INHIBIT ITERATIONS
10	BELL ON ERROR
9	LOOP ON ERROR
8	LOOP ON TEST SPECIFIED IN SWR<7:0>
7-0	# OF TEST TO LOOP ON (ONLY WHEN SWR8 = 1)

2.4 EXECUTION TIMES

EXECUTION TIME FOR THIS PROGRAM IS DEPENDENT ON THE MODEL OF PDP-11 IT IS BEING RUN ON. FOR A PDP-11/40 ABOUT 5 SECONDS IS NECESSARY TO DO 1 PASS OF THE PROGRAM WITHOUT ITERATIONS.

3.0 ERROR INFORMATION

3.1 STANDARD ERROR REPORTING PROCEDURES

ERROR PRINTOUTS CONSIST OF FROM 4 TO 8 COLUMNS OF DATA, A DATA HEADER, AND POSSIBLY A SHORT ERROR MESSAGE DESCRIBING THE ERROR. FOR EXAMPLE...

```
CLOCK FAILED TO INTERRUPT
PC      PS      SP      TEST#   LKS
002262  000344  000764  000007  000300
```

THE FIRST 4 COLUMNS OF OF THE ERROR MESSAGE ALWAYS SHOW THE CONTENTS OF THE PC, PS, SP, AND THE TEST NUMBER. MORE COLUMNS OF DATA ARE ADDED WHERE THEY MIGHT BE RELAVENT TO A PARTICULAR ERROR.

3.2 UNEXPECTED TRAP ERROR REPORTING

AN UNEXPECTED TRAP TO ADDRESS 4 CAUSES THE FOLLOWING MESSAGE TO BE PRINTED OUT...

```
TRAPPED TO LOC 4 FROM LOCATION "XXXXXX"
RESTARTING PROGRAM
```

IN THE ACTUAL MESSAGE THE "XXXXXX" IS REPLACED BY THE PC ADDRESS PUSHED ONTO THE STACK WHEN THE UNEXPECTED TRAP OCCURS. THE PROGRAM THEN TRYS TO RESTART ITSELF DESPITE SWITCH REGISTER SETTINGS.

3.3 POWER FAIL

IF A POWER FAIL CONDITION IS DETECTED THE FOLLOWING MESSAGE IS PRINTED...

```
POWER
```

AFTER PRINTING OUT THE MESSAGE THE PROGRAM TRYS TO RESTART ITSELF.

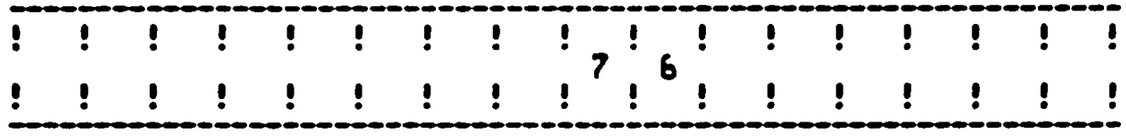
5.0 DEVICE INFORMATION

5.1 GENERAL INFORMATION

THE LINE CLOCK INTERRUPT VECTOR ADDRESS IS 100
THE LINE CLOCK PRIORITY LEVEL IS BR6

5.2 REGISTERS

LINE CLOCK STATUS REGISTER (LKS) 777546



BIT6 IF SET MONITOR=1 CAUSES AN INTERRUPT
BIT7 MONITOR BIT. SET BY CLOCK, CLEARED BY USER

7.0 FLOW CHARTS

FLOW CHART
****

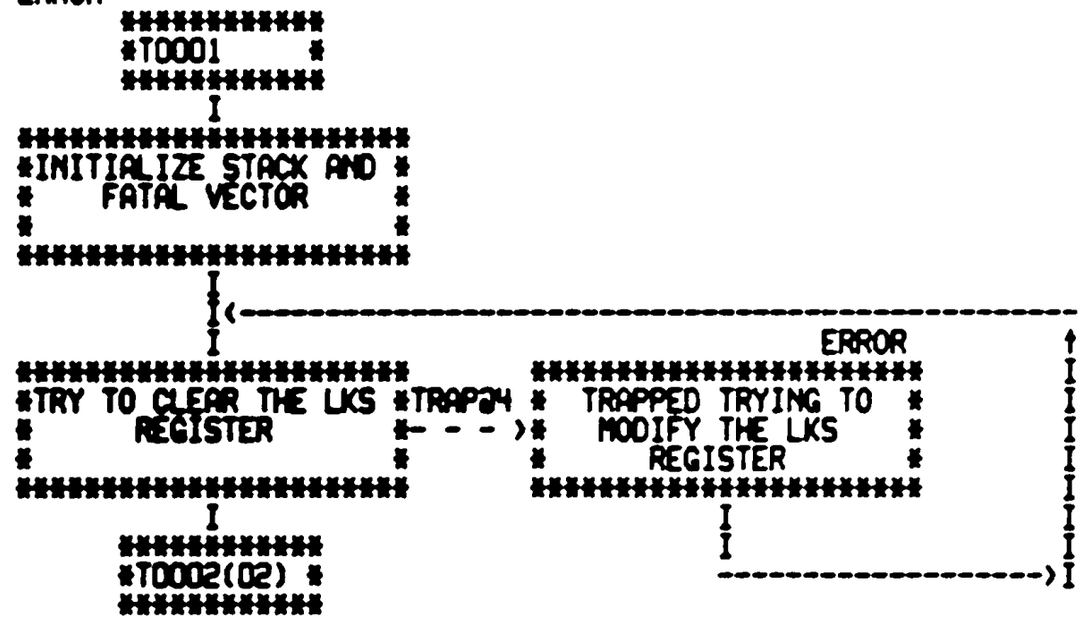
LINE CLOCK PROGRAM FLOW CHART

****

TABLE OF CONTENTS
****

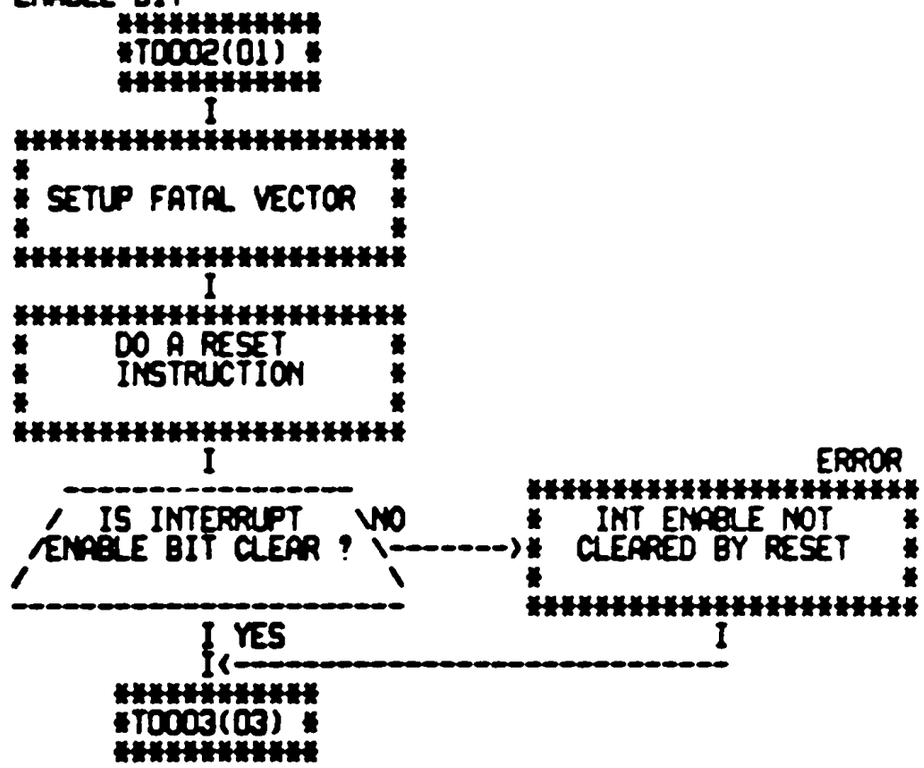
I01

LINE CLOCK PROGRAM FLOW CHART
TEST THAT THE LKS CAN BE ACCESSED WITHOUT A BUS ERROR

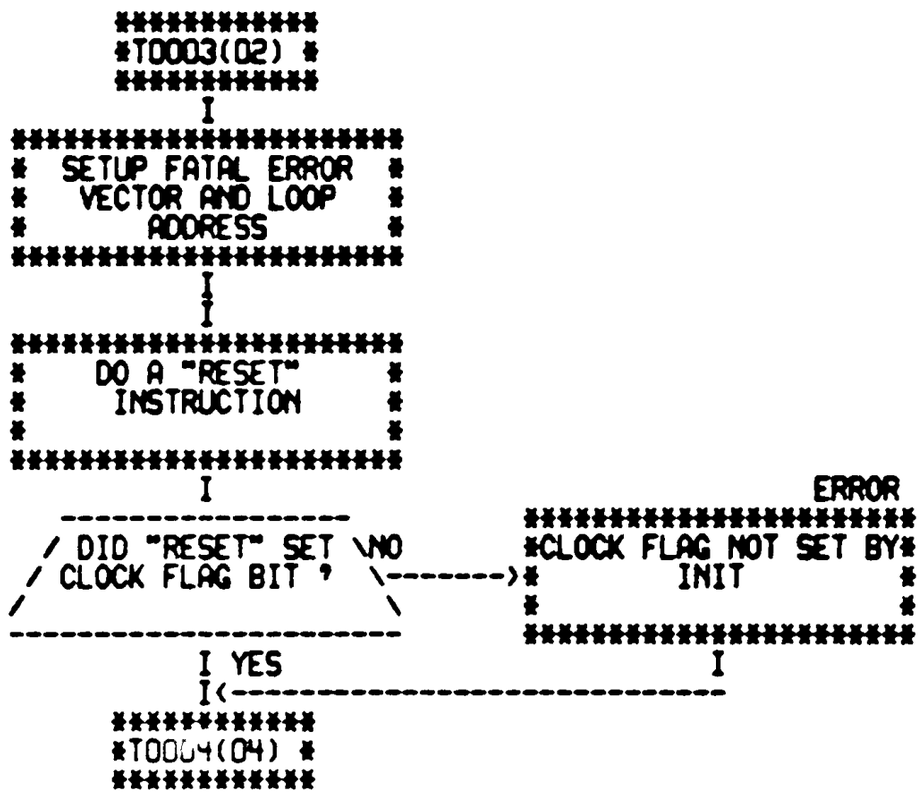


J01

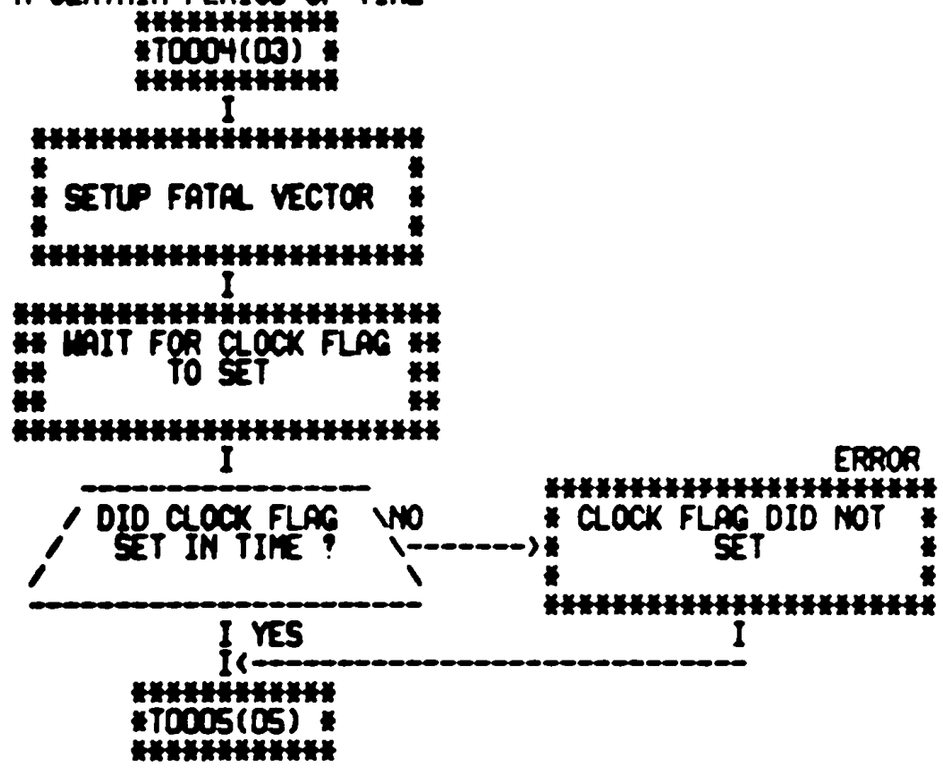
LINE CLOCK PROGRAM FLOW CHART
TEST THAT RESET CLEARS THE LINE CLOCK INTERRUPT ENABLE BIT

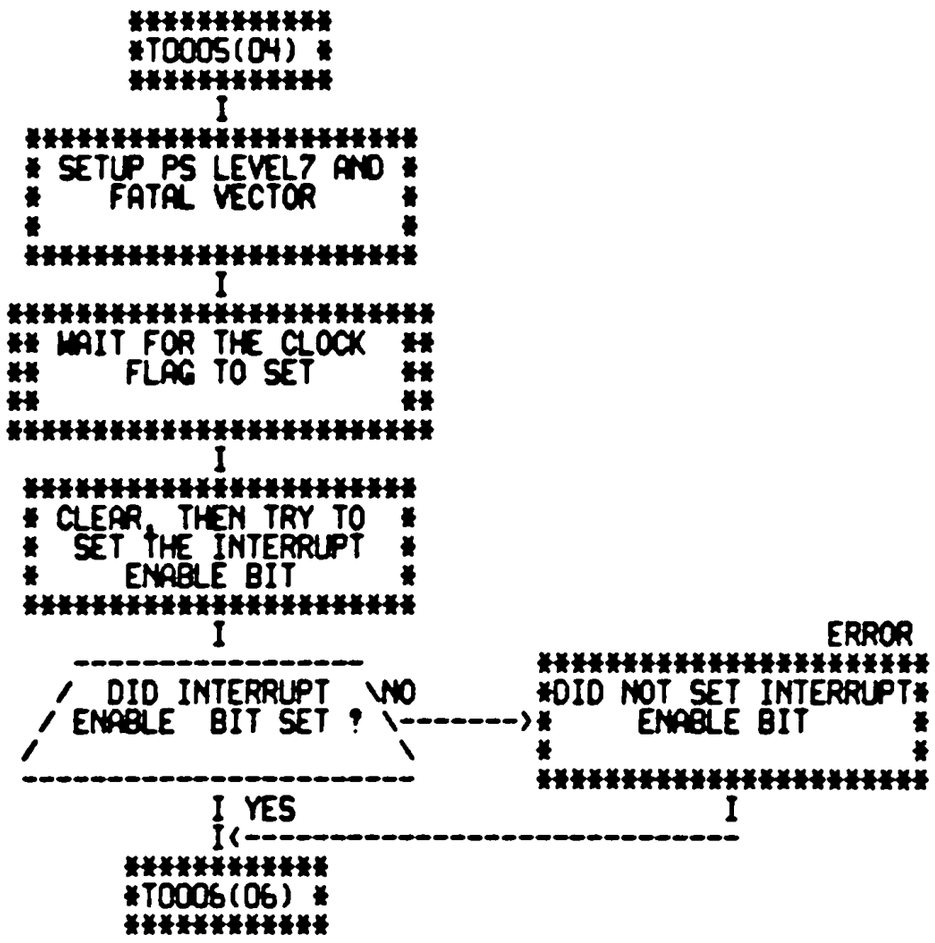


LINE CLOCK PROGRAM FLOW CHART
TEST THAT INIT SETS CLOCK FLAG BIT

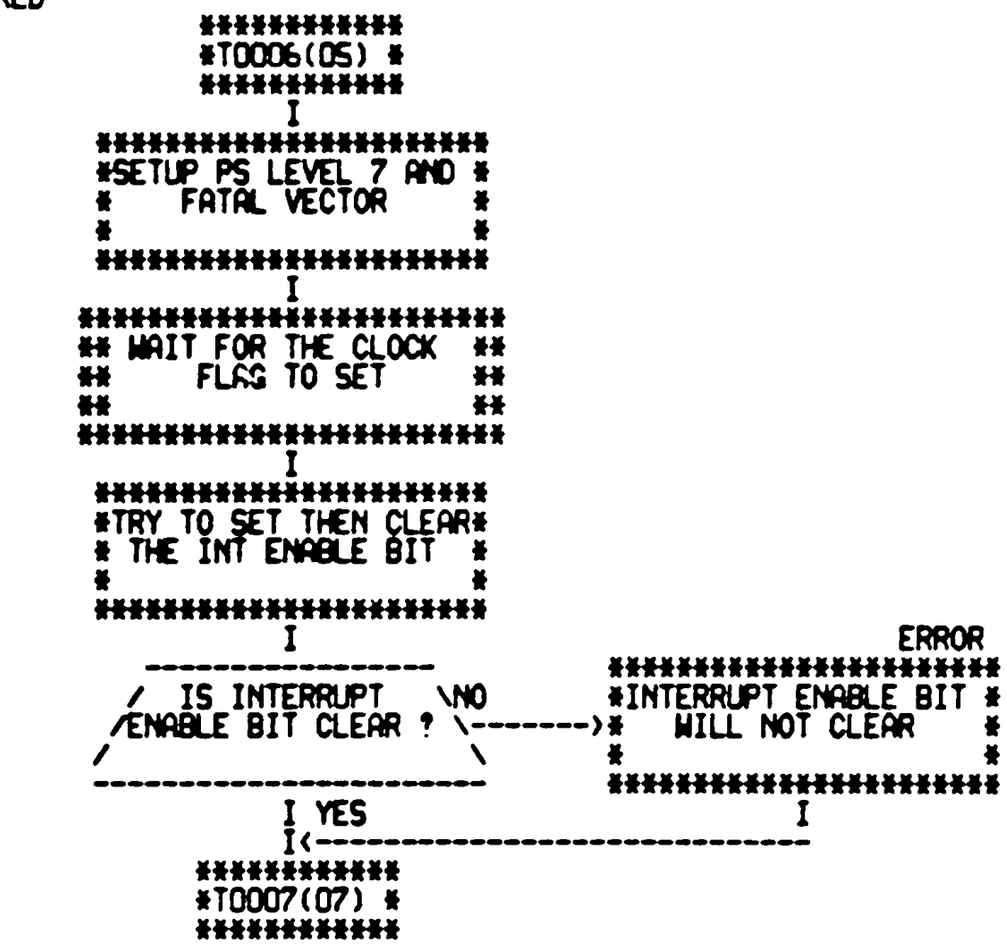


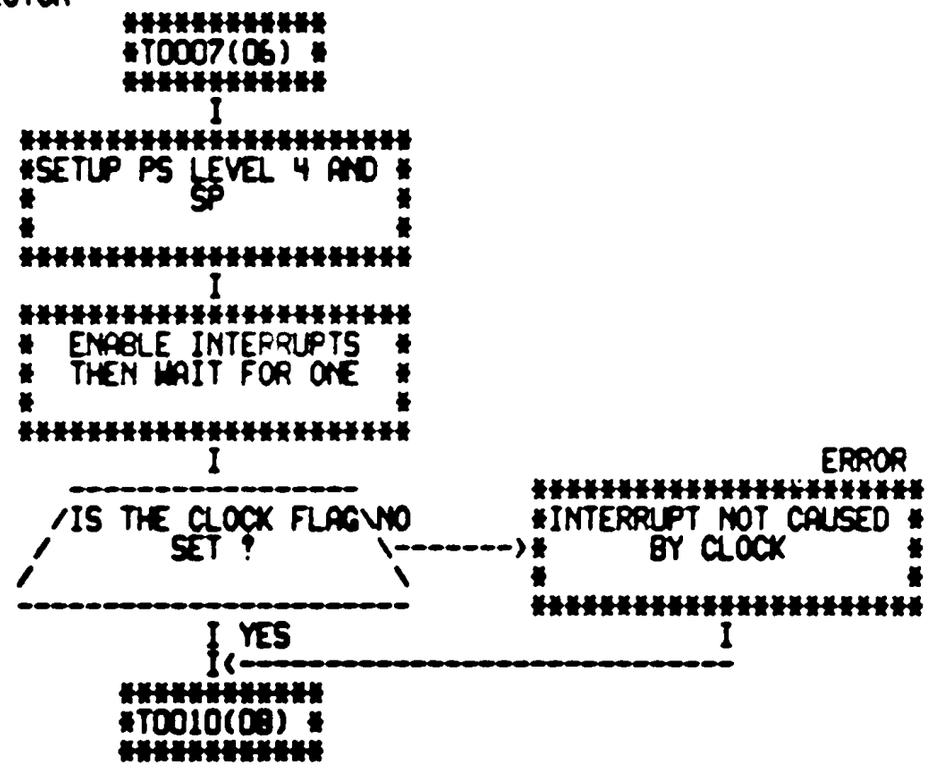
TEST THAT CLOCK FLAG WILL SET BY ITSELF WITHIN A CERTAIN PERIOD OF TIME



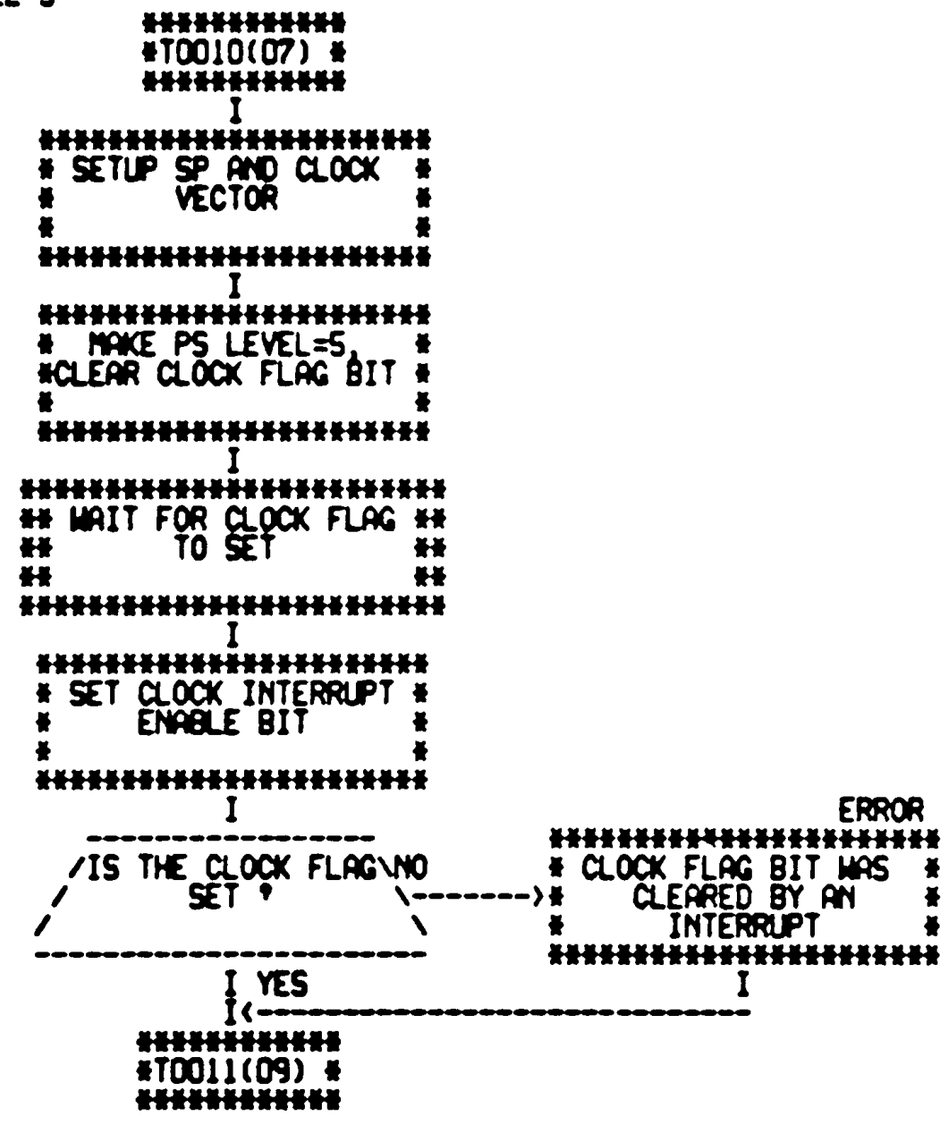

*TOOO5(04) *
* SETUP PS LEVEL7 AND *
* FATAL VECTOR *
** WAIT FOR THE CLOCK **
** FLAG TO SET **
* CLEAR, THEN TRY TO *
* SET THE INTERRUPT *
* ENABLE BIT *
DID INTERRUPT ENABLE BIT SET ?
NO
ERROR
DID NOT SET INTERRUPT
* ENABLE BIT *
YES
*TOOO6(06) *

NO1

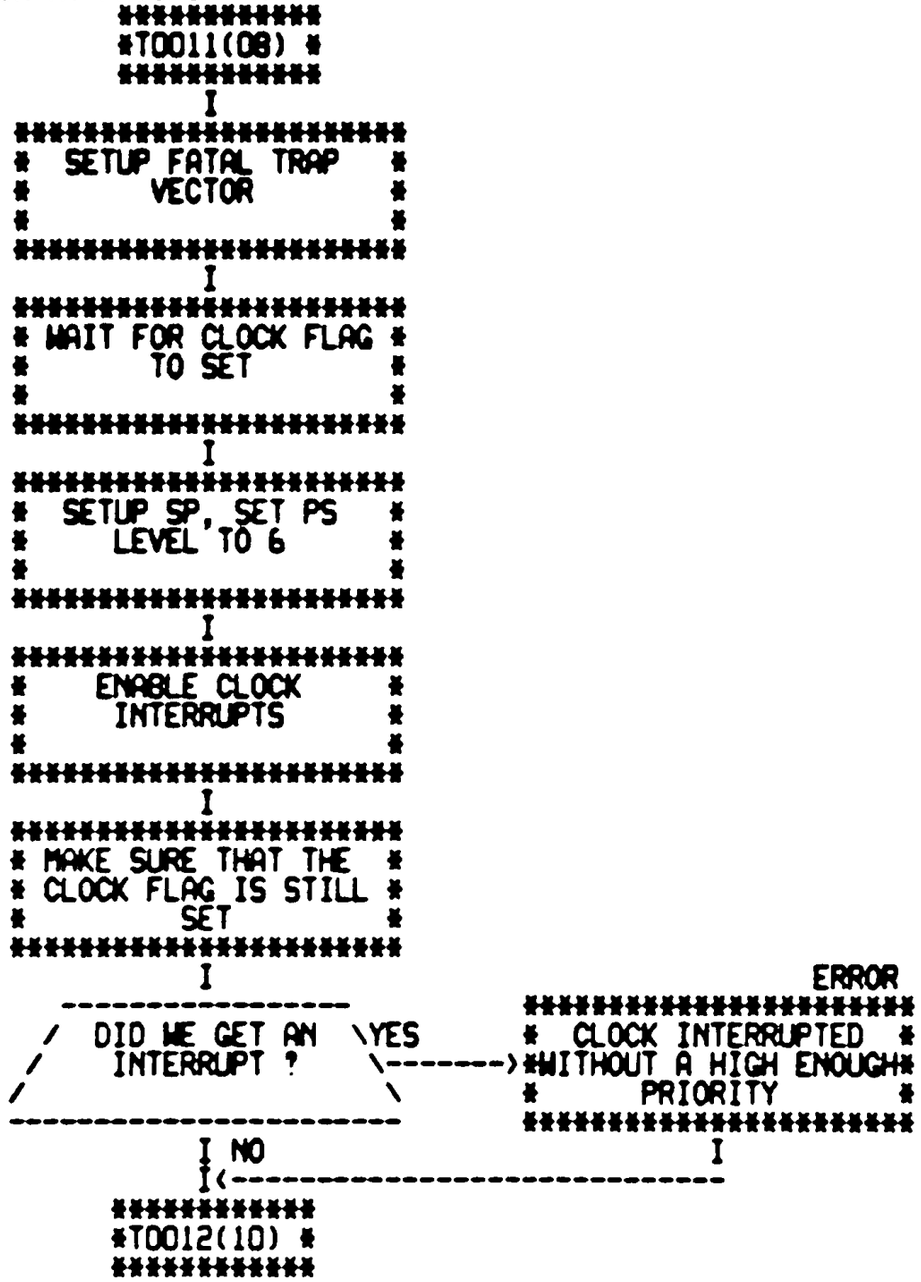

T0006(05)
SETUP PS LEVEL 7 AND FATAL VECTOR
WAIT FOR THE CLOCK FLAG TO SET
TRY TO SET THEN CLEAR THE INT ENABLE BIT
IS INTERRUPT ENABLE BIT CLEAR ?
NO
ERROR
INTERRUPT ENABLE BIT WILL NOT CLEAR
YES
T0007(07)

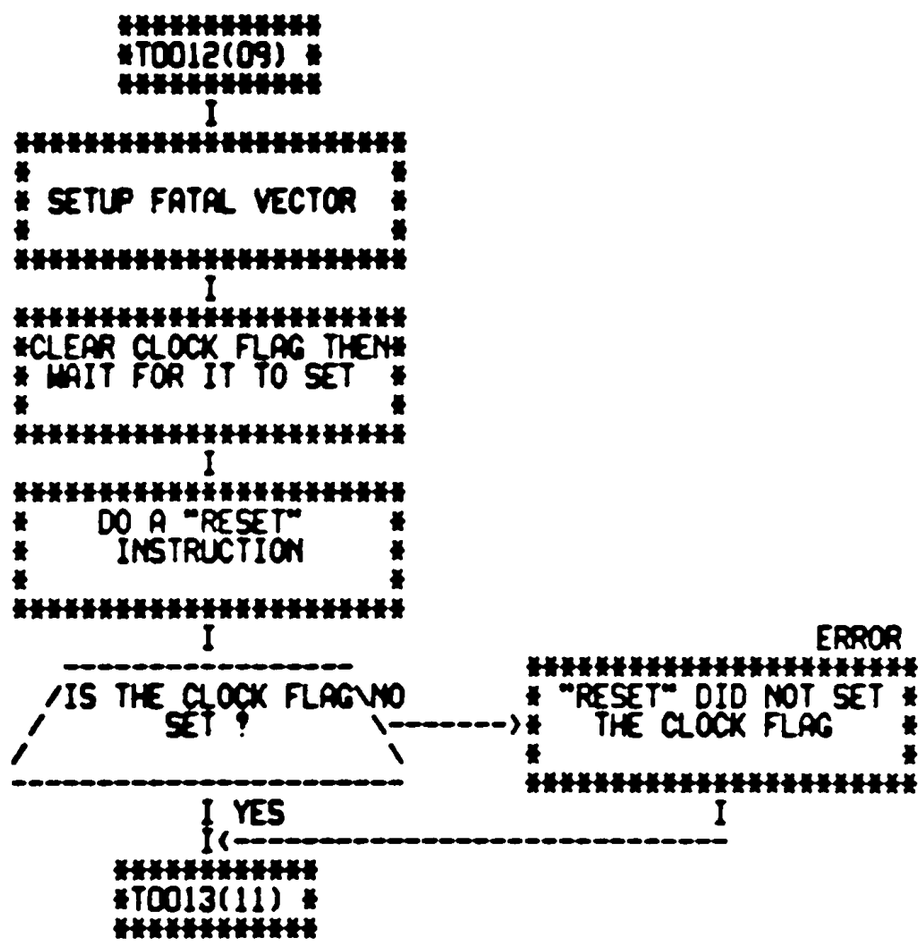

T0007(06)
SETUP PS LEVEL 4 AND SP
ENABLE INTERRUPTS THEN WAIT FOR ONE
IS THE CLOCK FLAG SET ?
NO
ERROR
INTERRUPT NOT CAUSED BY CLOCK
YES
T0010(08)

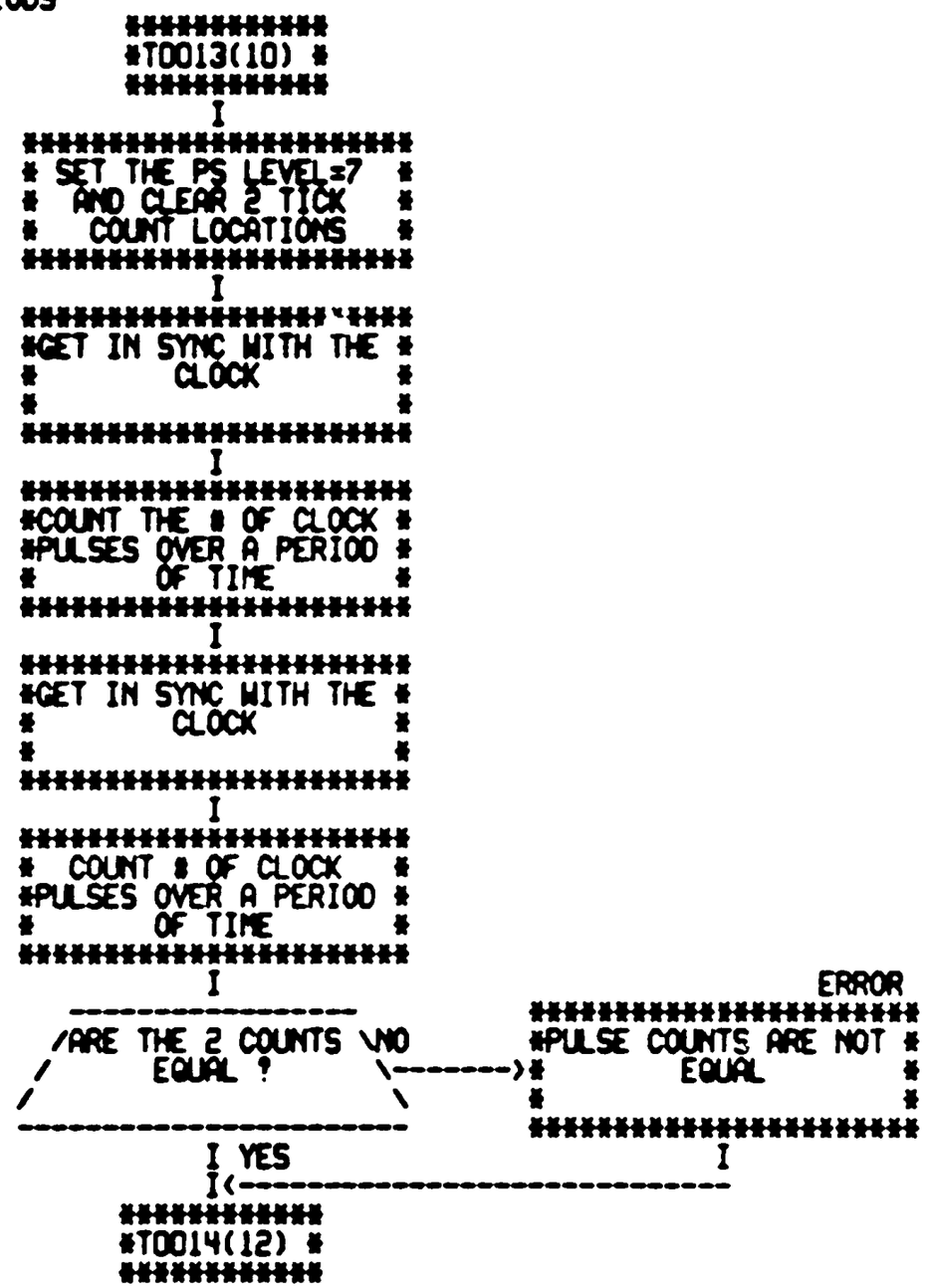
TEST THAT CLOCK WILL INTERRUPT WITH PS AT LEVEL 5

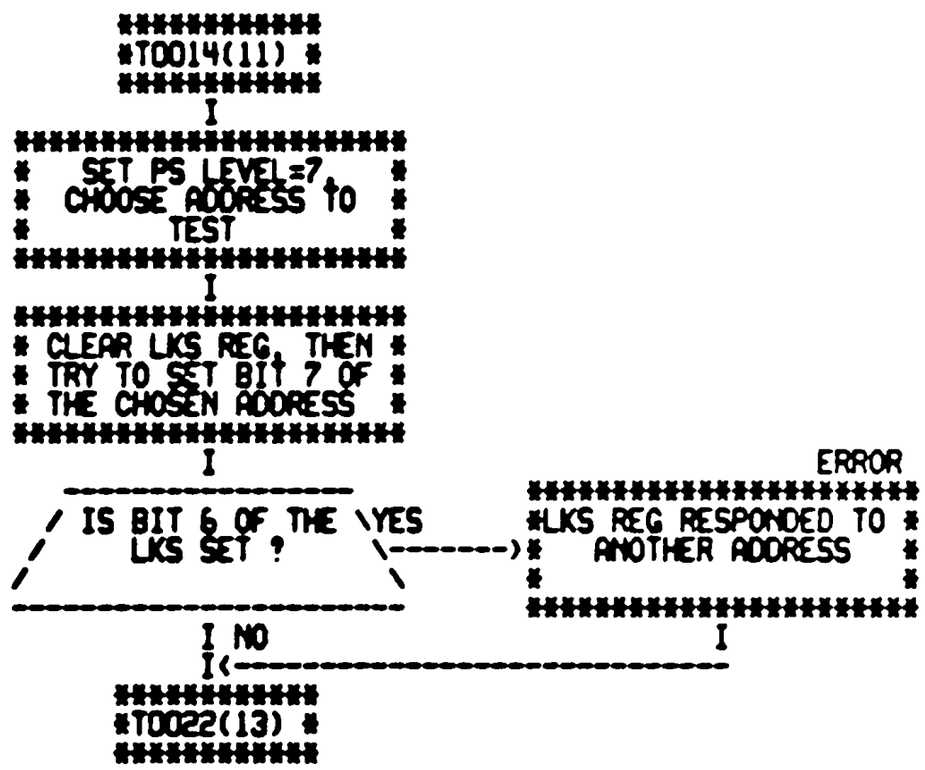


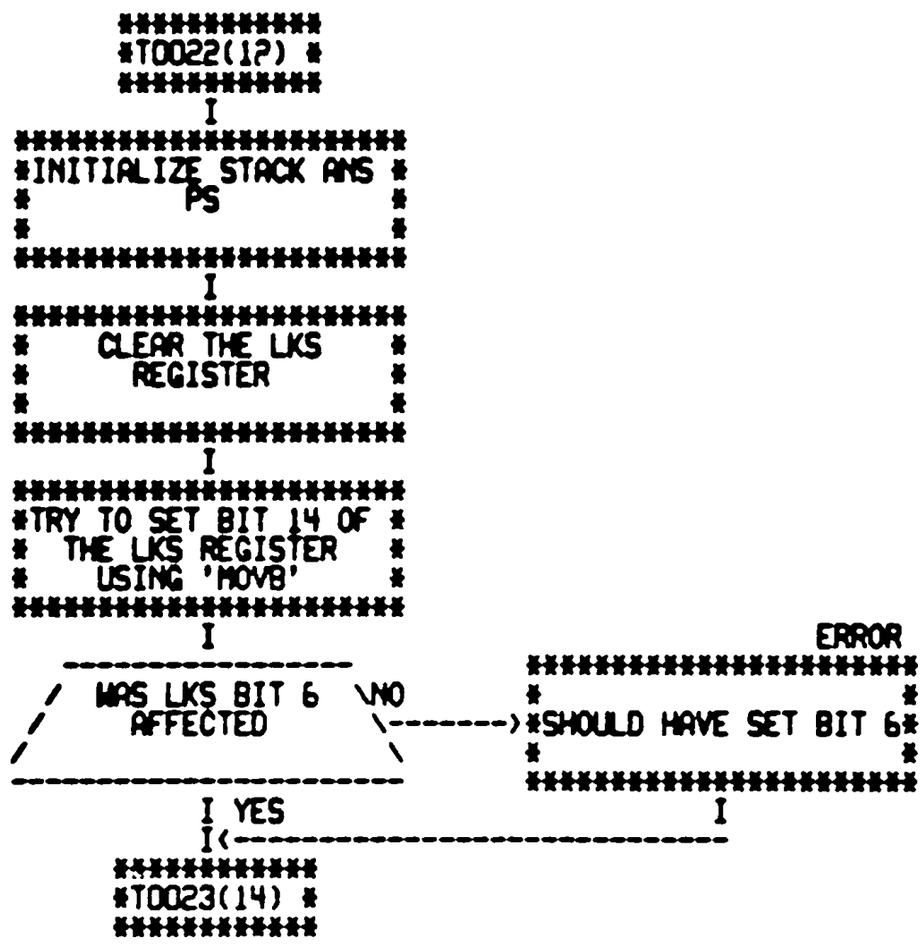
TEST THAT CLOCK WILL NOT INTERRUPT WITH PROCESSOR AT PRIORITY 4

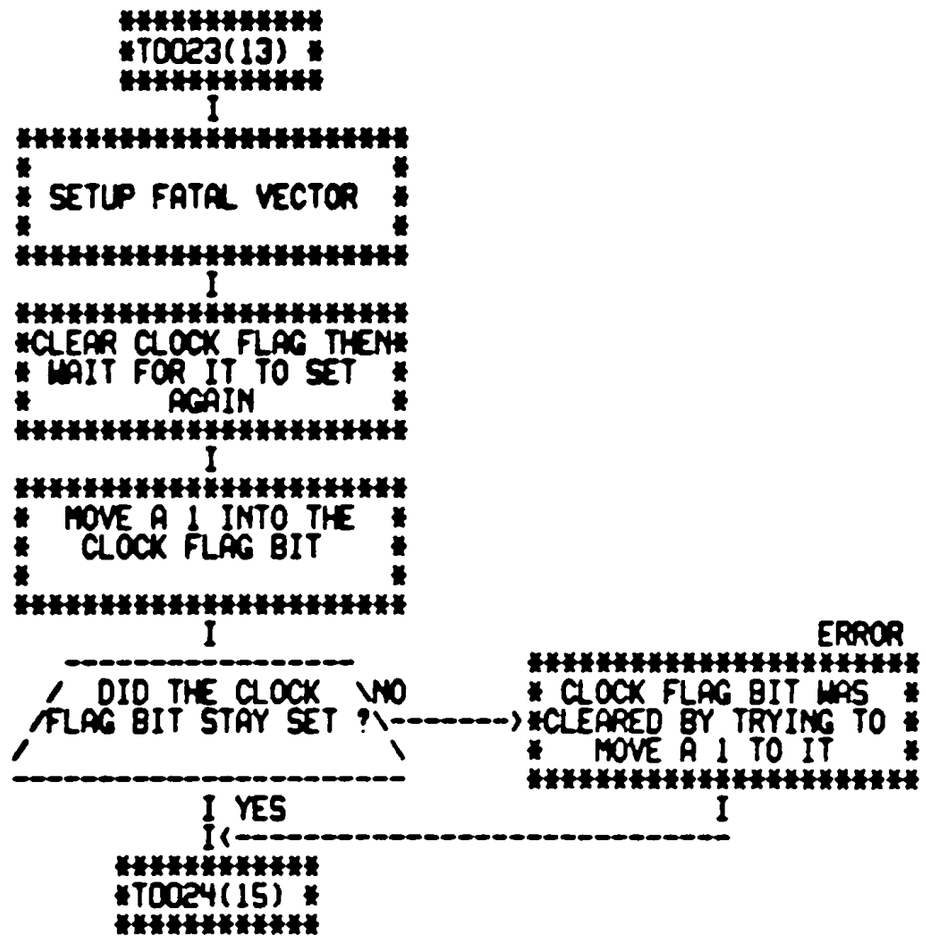


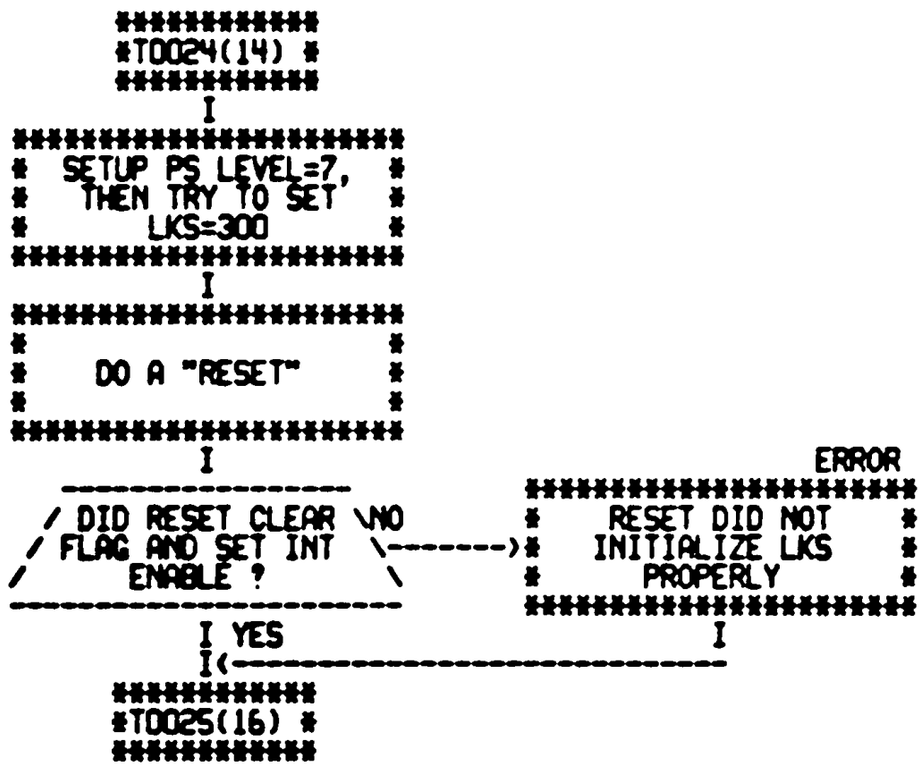

*TOO12(09) *
SETUP FATAL VECTOR
CLEAR CLOCK FLAG THEN
* WAIT FOR IT TO SET *
DO A "RESET"
INSTRUCTION
/IS THE CLOCK FLAG\NO
SET ?
YES
ERROR
* "RESET" DID NOT SET *
THE CLOCK FLAG
*TOO13(11) *

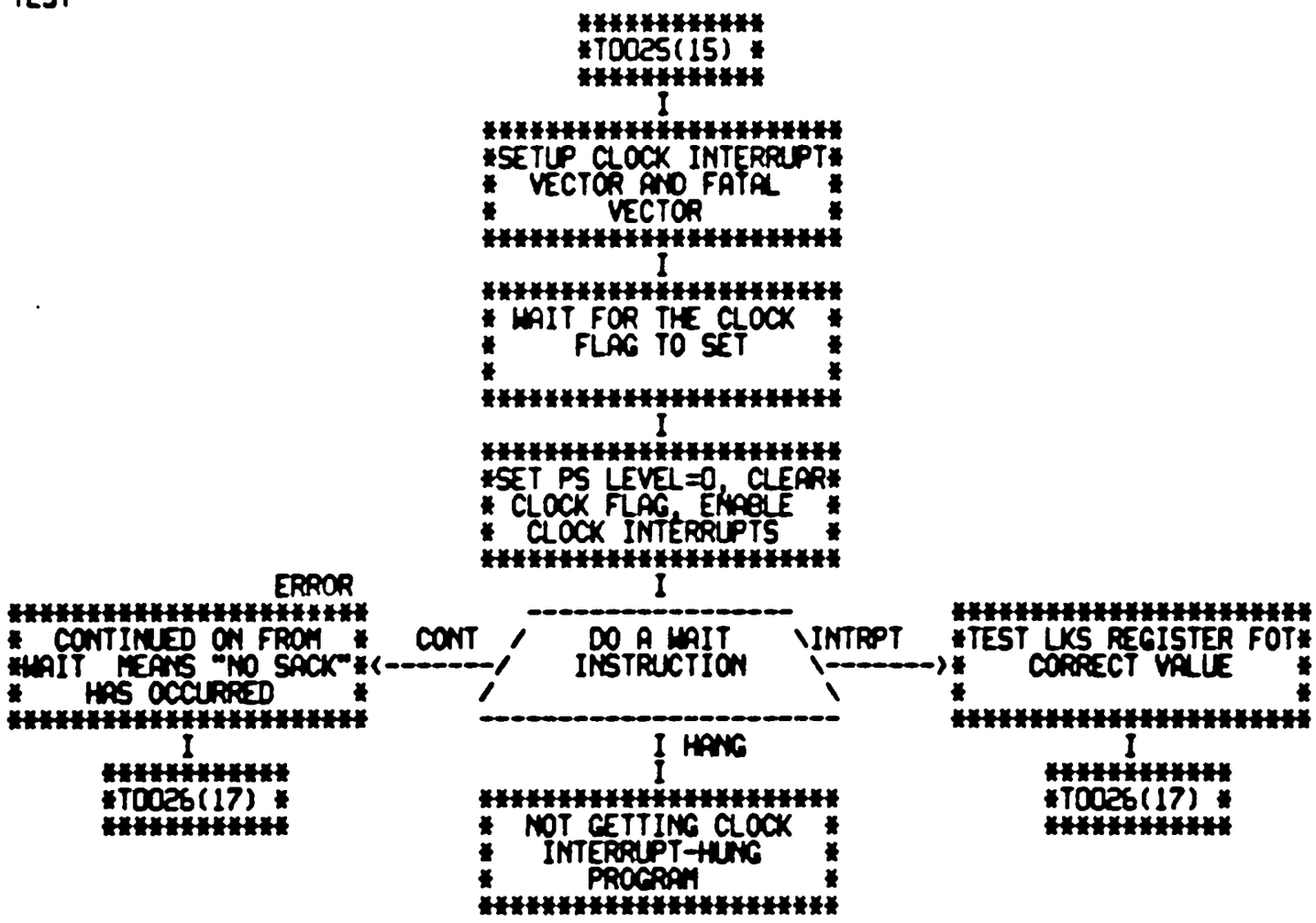

T0013(10)
SET THE PS LEVEL=7 AND CLEAR 2 TICK COUNT LOCATIONS
GET IN SYNC WITH THE CLOCK
COUNT THE # OF CLOCK PULSES OVER A PERIOD OF TIME
GET IN SYNC WITH THE CLOCK
COUNT # OF CLOCK PULSES OVER A PERIOD OF TIME
ARE THE 2 COUNTS EQUAL ?
NO
ERROR
PULSE COUNTS ARE NOT EQUAL
YES
T0014(12)

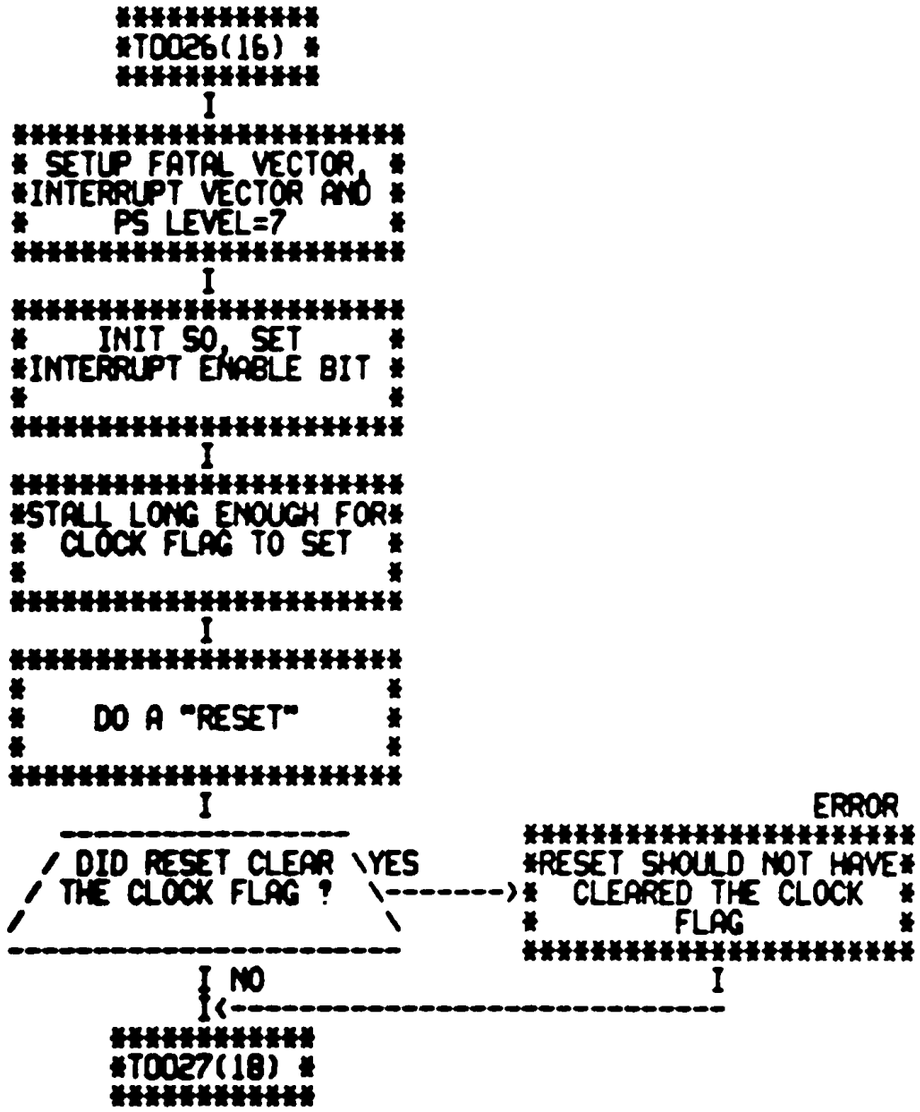

T0014(11)
SET PS LEVEL=7, CHOOSE ADDRESS TO TEST
CLEAR LKS REG, THEN TRY TO SET BIT 7 OF THE CHOSEN ADDRESS
IS BIT 7 OF THE LKS SET ?
YES
ERROR
LKS REG RESPONDED TO ANOTHER ADDRESS
NO
T0022(13)

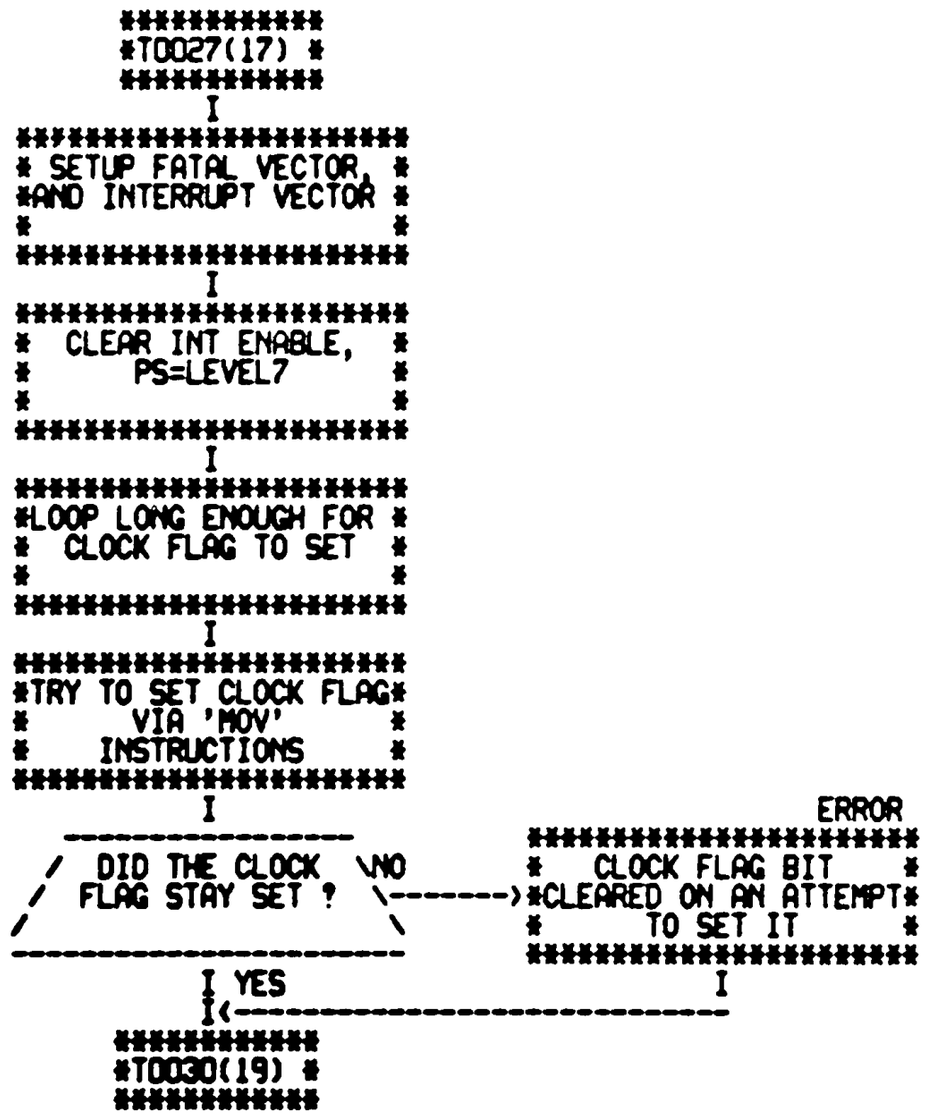

T0022(12)
INITIALIZE STACK AND PS
CLEAR THE LKS REGISTER
TRY TO SET BIT 14 OF THE LKS REGISTER USING 'MOVB'
WAS LKS BIT 6 AFFECTED
NO
ERROR
SHOULD HAVE SET BIT 6
YES
T0023(14)

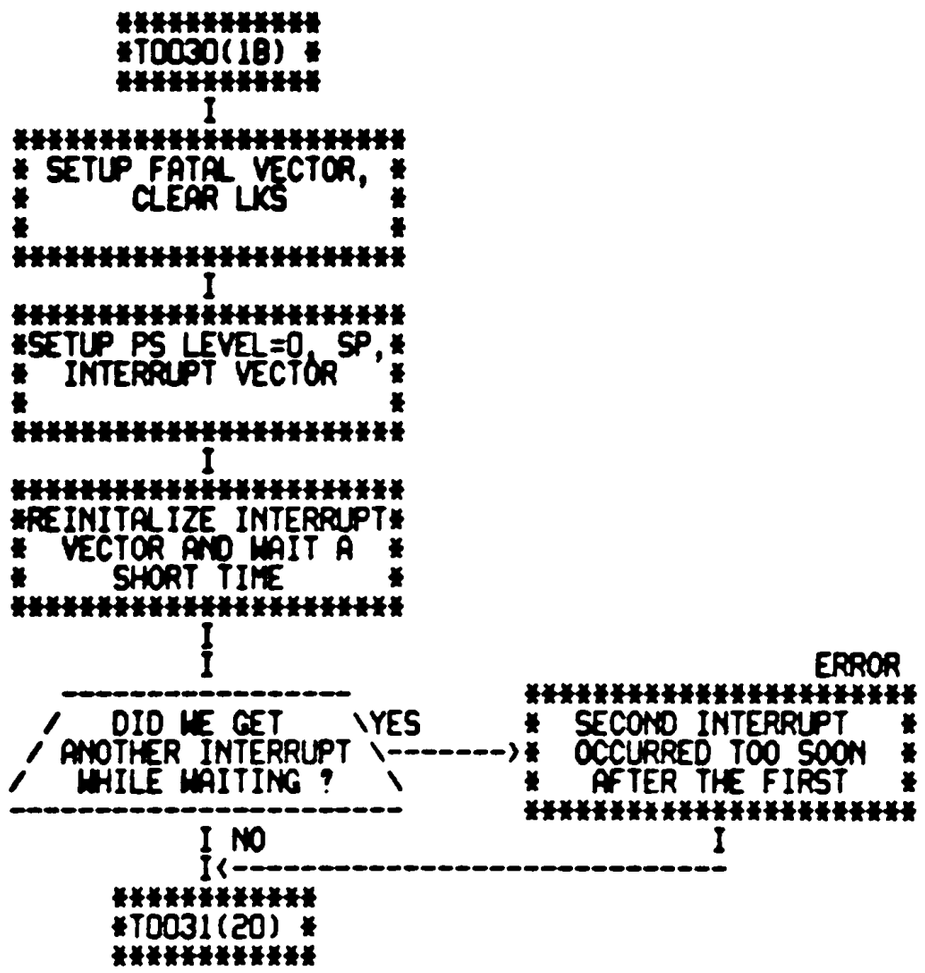

*T0023(13) *
SETUP FATAL VECTOR
CLEAR CLOCK FLAG THEN
WAIT FOR IT TO SET
AGAIN
MOVE A 1 INTO THE
CLOCK FLAG BIT
DID THE CLOCK
FLAG BIT STAY SET ?
NO
YES
ERROR
CLOCK FLAG BIT WAS
CLEARED BY TRYING TO
MOVE A 1 TO IT
*T0024(15) *

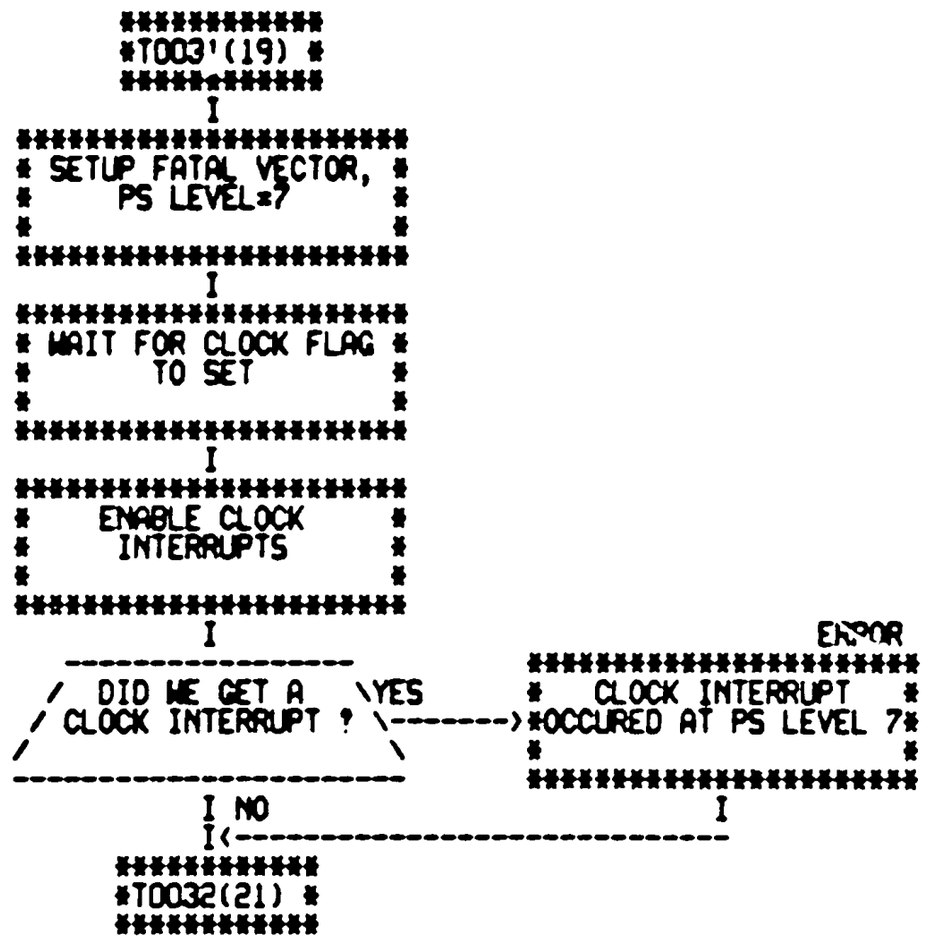

#T0024(14) #
SETUP PS LEVEL=7, THEN TRY TO SET LKS=300
DO A "RESET"
DID RESET CLEAR FLAG AND SET INT ENABLE ?
NO
ERROR
RESET DID NOT INITIALIZE LKS PROPERLY
YES
#T0025(16) #

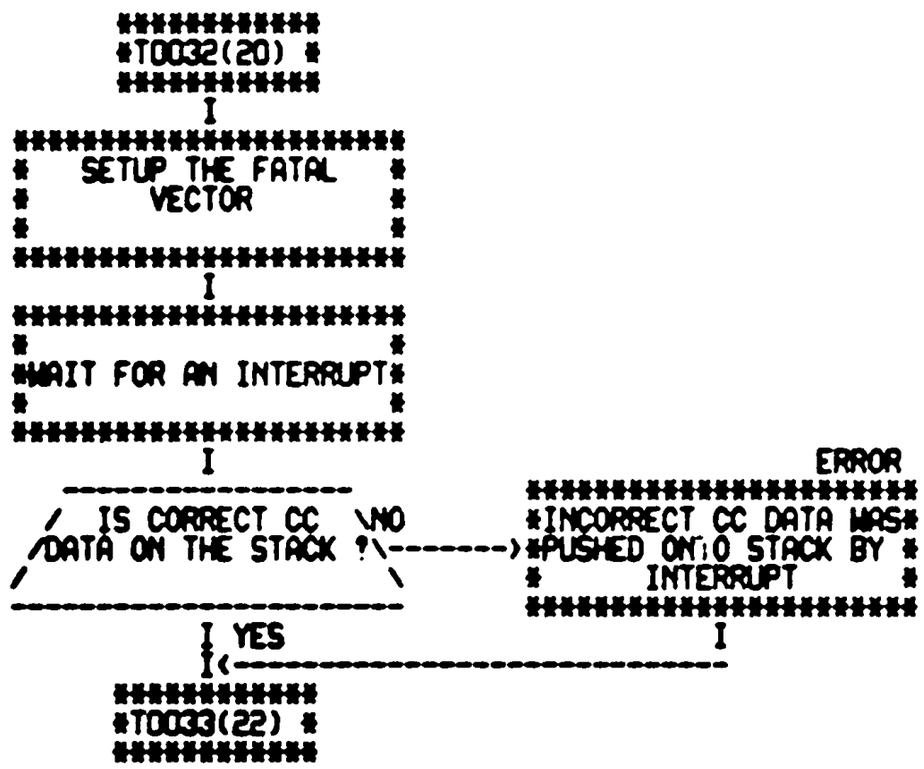

T0025(15)
SETUP CLOCK INTERRUPT VECTOR AND FATAL VECTOR
WAIT FOR THE CLOCK FLAG TO SET
SET PS LEVEL=0, CLEAR CLOCK FLAG, ENABLE CLOCK INTERRUPTS
DO A WAIT INSTRUCTION
CONT
INTRPT
ERROR
CONTINUED ON FROM WAIT MEANS "NO SACK" HAS OCCURRED
TEST LKS REGISTER FOT CORRECT VALUE
HANG
NOT GETTING CLOCK INTERRUPT-HUNG PROGRAM
T0026(17)
T0026(17)

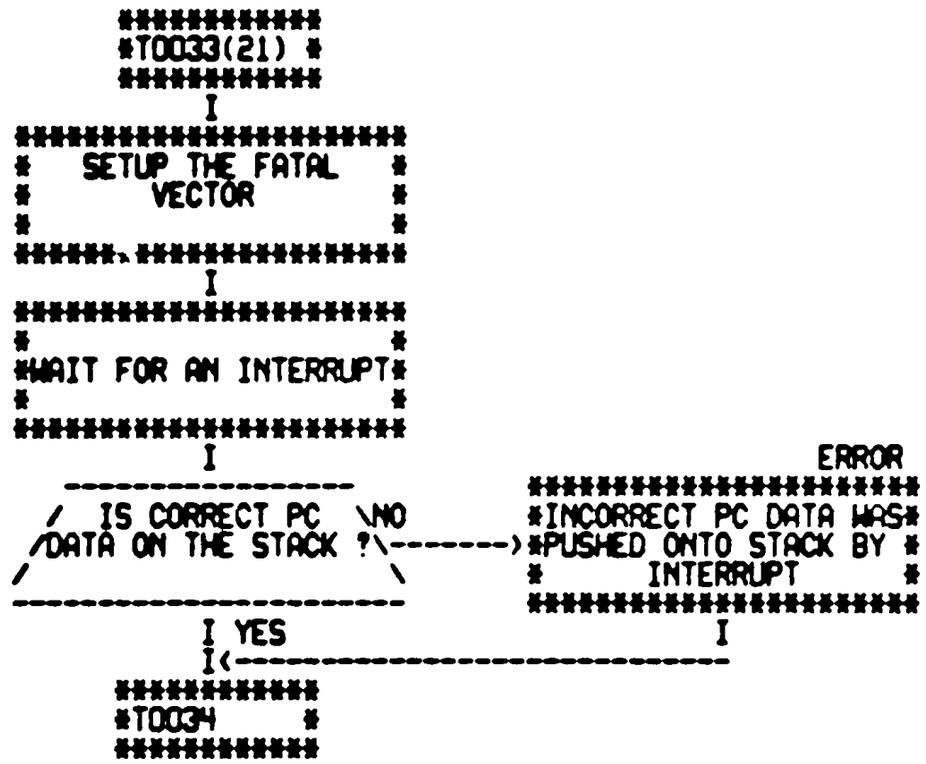

T0026(16)
SETUP FATAL VECTOR, INTERRUPT VECTOR AND PS LEVEL=7
INIT SO, SET INTERRUPT ENABLE BIT
STALL LONG ENOUGH FOR CLOCK FLAG TO SET
DO A "RESET"
DID RESET CLEAR THE CLOCK FLAG ?
YES
ERROR
RESET SHOULD NOT HAVE CLEARED THE CLOCK FLAG
NO
T0027(18)


*T0027(17) *
SETUP FATAL VECTOR,
AND INTERRUPT VECTOR
CLEAR INT ENABLE,
PS=LEVEL7
LOOP LONG ENOUGH FOR
CLOCK FLAG TO SET
TRY TO SET CLOCK FLAG
VIA 'MOV'
INSTRUCTIONS
DID THE CLOCK
FLAG STAY SET ?
NO
YES
ERROR
CLOCK FLAG BIT
CLEARED ON AN ATTEMPT
TO SET IT
*T0030(19) *


*TOO30(18) *
SETUP FATAL VECTOR, CLEAR LKS
SETUP PS LEVEL=0, SP, INTERRUPT VECTOR
REINITALIZE INTERRUPT VECTOR AND WAIT A SHORT TIME
DID WE GET ANOTHER INTERRUPT WHILE WAITING ?
YES
ERROR
SECOND INTERRUPT OCCURRED TOO SOON AFTER THE FIRST
NO
*TOO31(20) *


*T003'(19) *
SETUP FATAL VECTOR,
PS LEVEL=7
WAIT FOR CLOCK FLAG
TO SET
ENABLE CLOCK
INTERRUPTS
DID WE GET A
CLOCK INTERRUPT ?
YES
ERROR
CLOCK INTERRUPT
OCCURED AT PS LEVEL 7
NO
*T0032(21) *


T0032(20)
SETUP THE FATAL VECTOR
WAIT FOR AN INTERRUPT
IS CORRECT CC DATA ON THE STACK ?
NO
ERROR
INCORRECT CC DATA WAS PUSHED ONTO STACK BY INTERRUPT
YES
T0033(22)


*T0033(21) *
SETUP THE FATAL VECTOR
WAIT FOR AN INTERRUPT
IS CORRECT PC DATA ON THE STACK ?
NO
ERROR
INCORRECT PC DATA WAS
*PUSHED ONTO STACK BY *
INTERRUPT
YES
*T0034 *

ERROR	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18
	19	20	21	22														
T0001	01																	
T0002	01	02																
T0003	02	03																
T0004	03	04																
T0005	04	05																
T0006	05	06																
T0007	06	07																
T0010	07	08																
T0011	08	09																
T0012	09	10																
T0013	10	11																
T0014	11	12																
T0022	12	13																
T0023	13	14																
T0024	14	15																
T0025	15	16																
T0026	16	16	17															
T0027	17	18																
T0030	18	19																
T0031	19	20																
T0032	20	21																
T0033	21	22																
T0034	22																	

DZKWAD.P11 TABLE OF CONTENTS

```
                                .ABS
                                .ENABL AMA
                                .LIST   ME
                                .NLIST  MC, MD, CND
167400                          $SWR=167400
000000                          $TN=0
                                        .ENABL  ABS
                                .MCALL  .HEADER,.SCATCH,.SEOP,.EQUAT,.SWRHI,.SWRLO,.$SCOPE,.SETUP
                                .MCALL  .$TYPOCT,.$TYPDEC,.STRAP,.SPOWER,.SERROR,.$TYPE,STARS,.SERRTYP,.$CMTAG
                                .TITLE  MAINDEC-11-DZKWA-D  LINE FREQUENCY CLOCK PROGRAM
(1)                             ;*COPYRIGHT (C) 1970,1972,1975
(1)                             ;*DIGITAL EQUIPMENT CORP.
(1)                             ;*MAYNARD, MASS. 01754
(1)                             ;*
(1)                             ;*PROGRAM BY J. COMEAU
(1)                             ;*
(1)                             ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
(1)                             ;*PACKAGE (MAINDEC-11-DZQAC-A1).
(1)                             ;*

                                .SBTTL  OPERATIONAL SWITCH SETTINGS
(1)                             ;*
(1)                             ;*      SWITCH                  USE
(1)                             ;*      ------          ------------------
(1)                             ;*        15            HALT ON ERROR
(1)                             ;*        14            LOOP ON TEST
(1)                             ;*        13            INHIBIT ERROR TYPEOUTS
(1)                             ;*        11            INHIBIT ITERATIONS
(1)                             ;*        10            BELL ON ERROR
(1)                             ;*         9            LOOP ON ERROR
(1)                             ;*         8            LOOP ON TEST IN SWR<7:0>
(1)                             ;*       7-0            #OF TEST TO LOOP ON IF SWR<8> IS SET

                                .SBTTL  BASIC DEFINITIONS
(1)
(1)                             ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
001100                          STACK=  1100
(1)                             .EQUIV  EMT,ERROR               ;BASIC DEFINITION OF ERROR CALL
(1)                             .EQUIV  IOT,SCOPE               ;BASIC DEFINITION OF SCOPE CALL
177776                          PS=     177776                  ;PROCESSOR STATUS WORD
(1)                             .EQUIV  PS,PSW
177774                          STKLMT= 177774                  ;STACK LIMIT REGISTER
177772                          PIRQ=   177772                  ;PROGRAM INTERRUPT REQUEST REGISTER
177570                          SWR=    177570                  ;SWITCH REGISTER
177570                          DISPLAY=SWR
(1)
(1)                             ;*GENERAL PURPOSE REGISTER DEFINITIONS
000000                          R0=     %0                      ;GENERAL REGISTER
000001                          R1=     %1                      ;GENERAL REGISTER
000002                          R2=     %2                      ;GENERAL REGISTER
000003                          R3=     %3                      ;GENERAL REGISTER
000004                          R4=     %4                      ;GENERAL REGISTER
000005                          R5=     %5                      ;GENERAL REGISTER
000006                          R6=     %6                      ;GENERAL REGISTER
```

```
(1)             000007                  R7=     %7                      ;GENERAL REGISTER
(1)                                     .EQUIV  R6,SP                   ;STACK POINTER
(1)                                     .EQUIV  R7,PC                   ;PROGRAM COUNTER
(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
 29                                 ;MISCELANHUS EQUATES
 30         177546                  LKS=177546
 31         000240                  NOP=240
 32         000774                  BUF2=774
 33         000776                  BUF1=776
 34
(1)                                 .SBTTL  TRAP CATCHER
(1)
(1)         000000                          .=0
(1)                                 ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
(1)                                 ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
(1)                                 ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
(1)
(1)                                 .SBTTL  STARTING ADDRESS(ES)
(1)         000200                          .=200
(1)
(1)  000200 000137  001400                  JMP     @#KSTART                ;JUMP TO STARTING ADDRESS OF PROGRAM
 38         000050                          .=50
 39  000050 000000                          .WORD   0
 40  000052 000000                          .WORD   0
```

```
 41                                 ;;*****************************************************************
(1)                                 
(1)                                 .SBTTL  COMMON TAGS
(1)                                 
(1)                                 ;*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
(1)                                 ;*USED IN THE PROGRAM.
(1)                                 
(1)          000046                         .=46
(1)  000046  005640                         $ENDAD                          ;LOGICAL END OF PROGRAM
(1)                                 
(1)          001100                         .=1100
(1)                                 
(1)  001100                         $CMTAG:                                 ;START OF COMMON TAGS
(1)  001100  000000                 $PASS:  .WORD   0                       ;CONTAINS PASS COUNT
(1)  001102     000                 $TSTNM: .BYTE   0                       ;CONTAINS THE TEST NUMBER
(1)  001103     000                 $ERFLG: .BYTE   0                       ;CONTAINS ERROR FLAG
(1)  001104  000000                 $ICNT:  .WORD   0                       ;CONTAINS SUBTEST ITERATION COUNT
(1)  001106  000000                 $LPADR: .WORD   0                       ;CONTAINS SCOPE LOOP 1100
(1)  001110  000000                 $LPERR: .WORD   0                       ;CONTAINS SCOPE RETURN FOR ERRORS
(1)  001112  000000                 $ERTTL: .WORD   0                       ;CONTAINS TOTAL ERRORS DETECTED
(1)  001114     000                 $ITEMB: .BYTE   0                       ;CONTAINS ITEM CONTROL BYTE
(1)  001115     001                 $ERMAX: .BYTE   1                       ;CONTAINS MAX. ERRORS PER TEST
(1)  001116  000000                 $ERRPC: .WORD   0                       ;CONTAINS PC OF LAST ERROR INSTRUCTION
(1)  001120  000000                 $GDADR: .WORD   0                       ;CONTAINS 1100 OF 'GOOD' DATA
(1)  001122  000000                 $BDADR: .WORD   0                       ;CONTAINS 1100 OF 'BAD' DATA
(1)  001124  000000                 $GDDAT: .WORD   0                       ;CONTAINS 'GOOD' DATA
(1)  001126  000000                 $BDDAT: .WORD   0                       ;CONTAINS 'BAD' DATA
(1)  001130  000000  000000  000000         .WORD   0,0,0                   ;RESERVED--NOT TO BE USED
(1)  001136  177560                 $TKS:   177560                          ;TTY KBD STATUS
(1)  001140  177562                 $TKB:   177562                          ;TTY KBD BUFFER
(1)  001142  177564                 $TPS:   177564                          ;TTY PRINTER STATUS REG. 1100
(1)  001144  177566                 $TPB:   177566                          ;TTY PRINTER BUFFER REG. 1100
(1)  001146     000                 $NULL:  .BYTE   0                       ;CONTAINS NULL CHARACTER FOR FILLS
(1)  001147     002                 $FILLS: .BYTE   2                       ;CONTAINS # OF FILLER CHARACTERS REQUIRED
(1)  001150     012                 $FILLC: .BYTE   12                      ;INSERT FILL CHARS. AFTER A "LINE FEED"
(1)  001151     000                 $TPFLG: .BYTE   0                       ;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
(1)  001152  000000                 $REGAD: .WORD   0                       ;CONTAINS THE 1100 FROM
(1)                                                                         ;WHICH  ($REG0) WAS OBTAINED
(3)  001154  000000                 $REG0:  .WORD   0                       ;CONTAINS (($REGAD)+0)
(3)  001156  000000                 $REG1:  .WORD   0                       ;CONTAINS (($REGAD)+2)
(3)  001160  000000                 $REG2:  .WORD   0                       ;CONTAINS (($REGAD)+4)
(3)  001162  000000                 $REG3:  .WORD   0                       ;CONTAINS (($REGAD)+6)
(3)  001164  000000                 $REG4:  .WORD   0                       ;CONTAINS (($REGAD)+10)
(3)  001166  000000                 $REG5:  .WORD   0                       ;CONTAINS (($REGAD)+12)
(3)  001170  000000                 $REG6:  .WORD   0                       ;CONTAINS (($REGAD)+14)
(3)  001172  000000                 $REG7:  .WORD   0                       ;CONTAINS (($REGAD)+16)
(3)  001174  000000                 $TMP0:  .WORD   0                       ;USER DEFINED
(3)  001176  000000                 $TMP1:  .WORD   0                       ;USER DEFINED
(3)  001200  000000                 $TMP2:  .WORD   0                       ;USER DEFINED
(3)  001202  000000                 $TMP3:  .WORD   0                       ;USER DEFINED
(3)  001204  000000                 $TMP4:  .WORD   0                       ;USER DEFINED
(3)  001206  000000                 $TMP5:  .WORD   0                       ;USER DEFINED
(3)  001210  000000                 $TMP6:  .WORD   0                       ;USER DEFINED
(3)  001212  000000                 $TMP7:  .WORD   0                       ;USER DEFINED
```

```
(1)  001214  000000                 $TIMES: 0                          ;MAX. NUMBER OF ITERATIONS
(1)  001216  000000                 $ESCAPE:0                          ;ESCAPE ON ERROR 1100
(1)  001220  177607  000377         $BELL:  .ASCIZ  <207><377><377>    ;CODE FOR BELL
(1)  001224     077                 $QUES:  .ASCII  /?/                ;QUESTION MARK
(1)  001225     015                 $CRLF:  .ASCII  <15>               ;CARRIAGE RETURN
(1)  001226  000012                 $LF:    .ASCIZ  <12>               ;LINE FEED
(3)  001230  000000                 WORD:   000000
```

```
(2)                                 ;;*****************************************************************
(1)
(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)  001232                         $ERRTB:
 42  001232  007752                         EM1     ;"                                      LKS     LKS     "
 43  001234  010033                         DH1     ;"PC      PS      SP      TEST#   WAS     S/B     "
 44  001236  010114                         DT1     ;$ERRPC,$REG7,$REG6,$REG5,LKS,$GDDAT
 45  001240  000000                         0
 46
 47  001242  010132                         EM2     ;"CLOCK FAILED TO INTERRUPT"
 48  001244  010164                         DH2     ;"PC      PS      SP      TEST#   LKS     "
 49  001246  010236                         DT2     ;$ERRPC,$REG7,$REG6,$REG5,LKS
 50  001250  000000                         0
 51
 52  001252  010252                         EM3     ;"CLOCK INTERRUPTED WHEN THE PROCESSOR PRIORITY WAS TOO HIGH"
 53  001254  010345                         DH3     ;"PC      PS      SP      TEST#   LKS     "
 54  001256  010416                         DT3     ;$ERRPC,$REG7,$REG6,$REG5,LKS
 55  001260  000000                         0
 56
 57  001262  010432                         EM4     ;"CLOCK GIVES UNEQUAL # OF PULSES OVER TWO EQUAL PERIODS OF TIME"
 58  001264  010531                         DH4     ;"PC      PS      SP      TEST#   1ST     2ND"
 59  001266  010666                         DT4     ;$ERRPC,$REG7,$REG6,$REG5,$REG1,$REG0
 60  001270  000000                         0
 61
 62  001272  010704                         EM5     ;"LKS REGISTER RESPONDS TO ANOTHER ADDRESS"
 63  001274  010755                         DH5     ;"PC      PS      SP      TEST#   ADDRESS"
 64  001276  011026                         DT5     ;$ERRPC,$REG7,$REG6,$REG5,$GDADR
 65  001300  000000                         0
 66
 67  001302  011042                         EM6     ;"A NO SACK TIMEOUT HAS OCCURED"
 68  001304  011100                         DH6     ;"PC      PS      SP      TEST#   LKS     "
 69  001306  011152                         DT6     ;$ERRPC,$REG7,$REG6,$REG5,LKS
 70  001310  000000                         0
 71
 72  001312  011166                         EM7     ;"WRONG CONDITION CODES WERE PUT ONTO STACK BY INTERRUPT"
 73  001314  011255                         DH7     ;"PC      PS      SP      TEST#   CC      CC"
 74  001316  011334                         DT7     ;$ERRPC,$REG7,$REG6,$REG5,BUF1,$GDDAT
 75  001320  000000                         0
 76
 77  001322  011352                         EM10    ;"WRONG PC PUT ONTO THE STACK BY AN INTERRUPT"
 78  001324  011426                         DH10    ;"PC      PS      SP      TEST#   "
```

```
                                                DT10    ;$ERRPC,$REG7,$REG6,$REG5,BUF2,$GDDAT
                                                0
                                                EM11    ;"TRAPPED TRYING TO ACCESS LKS REGISTER"
                                                DH11    ;"(PC)    (PS)    (SP)    TEST#"
                                                DT11    ;$ERRPC,$REG7,$REG6,$REG5
                                        0
                                        ;STARTUP CODE
                                        .=1400
                                        KSTART: MOV     #1000,SP                ;INITIALIZE THE STACK SO WE CAN CALL THE TYPEOUT
                                                MOV     PS,-(SP)                ;SAVE STATUS
                                                JSR     PC,$TYPE                ;PRINTOUT STARTUP MESSAGE
                                                STMES                           ;ADDRESS OF MESSAGE "MAINDEC-11-DZKWA-D"
                                        START:
                                                MOV     #$CMTAG,R6              ;FIRST LOCATION TO BE CLEARED
                                                CLR     (R6)+                   ;CLEAR MEMORY LOCATION
                                                CMP     #$TKS,R6                ;DONE?
                                                BNE     .-6                     ;LOOP BACK IF NO
                                                MOV     #1000,SP                        ;SETUP THE STACK POINTER
                                                MOV     #$SCOPE,@#IOTVEC        ;IOT VECTOR FOR SCOPE ROUTINE
                                                MOV     #340,@#IOTVEC+2         ;LEVEL 7
                                                MOV     #$ERROR,@#EMTVEC        ;EMT VECTOR FOR ERROR ROUTINE
                                                MOV     #340,@#EMTVEC+2         ;LEVEL 7
                                                MOV     #$TRAP,@#TRAPVEC        ;TRAP VECTOR FOR TRAP CALLS
                                                MOV     #340,@#TRAPVEC+2        ;LEVEL 7
                                                MOV     #$PWRDN,@#PWRVEC        ;POWER FAILURE VECTOR
                                                MOV     #340,@#PWRVEC+2         ;LEVEL 7
                                                MOV     $ENDCT,$EOPCT           ;SETUP END-OF-PROGRAM COUNTER
                                                MOVB    #1,$ERMAX               ;ALLOW ONE ERROR PER TEST
                                                MOV     #.,$LPADR               ;INITIALIZE THE LOOP ADDRESS FOR SCOPE
                                                TST     42              ;LOADED BY A MONITOR
                                                BEQ     T0001           ;BR IF NO
                                                RESET                   ;YES--GENERATE AN INIT


                                        .SBTTL  TEST THAT THE LKS CAN BE REFERENCED WITHOUT A BUS ERROR
                                        ;LKS ACCESS TEST
                                        T0001:  SCOPE
                                                MOV     #E0001,@#4      ;PREPARE FOR ADDRESSING THE LKS REGISTER. BAD HARDWARE
                                                MOV     #340,@#6        ;COULD CAUSE A TRAP TO 4
                                                MOV     #R0001,$LPADR   ;TIGHTEN UP THE SCOPE LOOP A BIT IN CASE OF AN ERROR
                                        R0001:  MOV     #1000,SP        ;SETUP THE STACK POINTER IN CASE OF AN ERROR
                                        I0001:  CLR     @#LKS           ;JUST REFERENCE LKS. DONT WORRY IF IT DIDNT CLEAR YET
                                                BR      T0002           ;WE DIDNT TRAP IF WE REACH HERE. GO ON TO NEXT TEST
                                        E0001:  ERROR 11                ;ERROR:::TRAPED TRYING TO ACCESS THE LKS REGISTER


                                        .SBTTL  TEST THAT START CLEARS LINE CLOCK INTERRUPT ENABLE BIT
                                        ;TEST THAT START CLEARS LINE CLOCK INTERRUPT ENABLE BIT
                                        T0002:  SCOPE
```

```
   117  001612  012737  005674  000004          MOV     #TRAP0,@#4      ;SETUP VECOR IN CASE OF UNFORSEEN PROBLEMS
   (1)  001620  012737  000340  000006          MOV     #340,@#6        ;NO INTERRUPTS WHILE PRINTING FATAL MESSAGE
   118  001626  012737  001634  001106          MOV     #R0002,$LPADR   ;SETUP LOOPBACK ADDRESS IN CASE OF AN ERROR
   119  001634  000005                  R0002:  RESET
   120  001636  012737  000200  001124          MOV     #200,$GDDAT     ;HAVE GOOD DATA INFO READY FOR TYPEOUT IN CASE OF AN ERR
   121  001644  032737  000100  177546          BIT     #100,LKS        ;TEST THE INTERRUPT ENABLE BIT
   122  001652  001401                          BEQ     T0003
   123  001654  104001                  E0002:  ERROR 1                 ;ERROR, CLOCK INTERRUPT ENABLE NOT CLEARED BY INIT
   124
   125
   126
   127                                  .SBTTL  TEST THAT START SETS CLOCK FLAG
   128                                  ;TEST THAT START SETS CLOCK FLAG
   129  001656  000004                  T0003:  SCOPE
   130  001660  012737  005674  000004          MOV     #TRAP0,@#4      ;SETUP VECOR IN CASE OF UNFORSEEN PROBLEMS
   (1)  001666  012737  000340  000006          MOV     #340,@#6        ;NO INTERRUPTS WHILE PRINTING FATAL MESSAGE
   131  001674  012737  000200  001124          MOV     #200,$GDDAT     ;HAVE GOOD DATA INFO READY FOR TYPEOUT IN CASE OF AN ERR
   132  001702  012737  001710  001106          MOV     #R0003,$LPADR   ;SETUP LOOPBACK ADDRESS IN CASE OF AN ERROR
   133  001710  000005                  R0003:  RESET                   ;SHOULD SET THE  CLOCK FLAG
   134  001712  105737  177546                  TSTB    LKS             ;FIND OUT IF IT DID
   135  001716  100401                          BMI     T0004           ;GO ON TO THE NEXT TEST IF IT SET THE CLOCK FLAG
   136  001720  104001                  E0003:  ERROR 1                 ;ERROR, CLOCK FLAG NOT SET BY INIT
   137
   138
   139
   140                                  .SBTTL  TEST THAT CLOCK FLAG WILL SET AFTER SUFFICIENT PERIOD OF TIME (20 MS MIN)
   141                                  ;TEST THAT CLOCK FLAG WILL SET AFTER SUFFICIENT PERIOD OF TIME (20 MS MIN)
   142  001722  000004                  T0004:  SCOPE
   143  001724  012737  005674  000004          MOV     #TRAP0,@#4      ;SETUP VECOR IN CASE OF UNFORSEEN PROBLEMS
   (1)  001732  012737  000340  000006          MOV     #340,@#6        ;NO INTERRUPTS WHILE PRINTING FATAL MESSAGE
   144  001740  012737  001746  001106          MOV     #R0004,$LPADR   ;SETUP LOOPBACK ADDRESS IN CASE OF AN ERROR
   145  001746  012737  000200  001124  R0004:  MOV     #200,$GDDAT     ;HAVE GOOD DATA INFO READY FOR TYPEOUT IN CASE OF AN ERR
   146  001754  005037  177546                  CLR     LKS             ;CLEAR THE CLOCK FLAG
   147  001760  005000                          CLR     R0              ;AND A TIMER LOCATION
   148  001762  105737  177546          A0004:  TSTB    LKS             ;IS CLOCK FLAG SET
   149  001766  100403                          BMI     T0005
   150  001770  005200                          INC     R0              ;NO, INCREMENT COUNT DO3 WAIT FOR SOMEMORE
   151  001772  001373                          BNE     A0004           ;WAIT SUFFICIENT AMOUNT OF TIME FOR CLOCK
   152  001774  104001                  E0004:  ERROR 1                 ;ERROR, CLOCK FLAG FAILED TO SET
   153
   154
   155
   156                                  .SBTTL  TEST THAT INTERRUPT ENABLE BIT MAY BE SET
   157                                  ;TEST THAT INTERRUPT ENABLE BIT MAY BE SET
   158  001776  000004                  T0005:  SCOPE
   159  002000  012737  005674  000004          MOV     #TRAP0,@#4      ;SETUP VECOR IN CASE OF UNFORSEEN PROBLEMS
   (1)  002006  012737  000340  000006          MOV     #340,@#6        ;NO INTERRUPTS WHILE PRINTING FATAL MESSAGE
   160  002014  012737  000100  001124          MOV     #100,$GDDAT     ;HAVE GOOD DATA INFO READY FOR TYPEOUT IN CASE OF AN ERR
   161  002022  012737  002036  001106          MOV     #R0005,$LPADR   ;SETUP LOOP BACK ADDRESS IN CASE OF ERROR
   162  002030  012737  000340  177776          MOV     #340,PS         ;SET PRIORITY TO LEVEL 7, NO INTERRUPTS
   163  002036  005037  177546          R0005:  CLR     LKS
   164  002042  005003                          CLR     R3              ;INITIALIZE A COUNTER LOCATION
   (1)  002044  105737  177546          A0005:  TSTB    LKS             ;IS THE CLOCK FLAG SET?
   (1)  002050  100404                          BMI     B0005           ;IF SO, CONTINUE ON WITH THE TEST
```

```
(1)  002052  005203                        INC     R3              ;IF NOT INCREMENT THE COUNTER LOCATION
(1)  002054  001373                        BNE     A0005           ;AND GO TEST THE CLOCK FLAG AGAIN. UNLESS...
(1)  002056  104001                E0005:   ERROR 1                 ;CLOCK FLAG DID NOT SET AFTER A WAITING PERIOD > 20 MS
(1)  002060  000410                        BR      T0006
(1)  002062                        B0005:
165  002062  012737  000100  177546        MOV     #100,LKS        ;CLEAR CLOCK FLAG AND SET INTERRUPT ENABLE
166  002070  032737  000100  177546        BIT     #100,LKS        ;IS INTERRUPT ENABLE SET?
167  002076  001001                        BNE     T0006
168  002100  104001                E1005:   ERROR 1                 ;ERROR INTERRUPT ENABLE NOT SET
169
170
171
172                                .SBTTL  TEST THAT INTERRUPT ENABLE BIT MAY BE CLEARED
173                                ;TEST THAT INTERRUPT ENABLE BIT MAY BE CLEARED
174  002102  000004                T0006:   SCOPE
175  002104  012737  005674  000004        MOV     #TRAP0,@#4      ;SETUP VECOR IN CASE OF UNFORSEEN PROBLEMS
(1)  002112  012737  000340  000006        MOV     #340,@#6        ;NO INTERRUPTS WHILE PRINTING FATAL MESSAGE
176  002120  012737  000000  001124        MOV     #0,$GOOAT
177  002126  012737  002142  001106        MOV     #R0006,$LPADR   ;INITIALIZE THE LOOPBACK ADDRESS IN CASE OF AN ERROR
178  002134  012737  000340  177776        MOV     #340,PS         ;SET PRIORITY LEVEL TO 7. NO INTERRUPTS
179  002142  005037  177546        R0006:  CLR     LKS
180  002146  005003                        CLR     R3              ;INITIALIZE A COUNTER LOCATION
(1)  002150  105737  177546        A0006:  TSTB    LKS             ;IS THE CLOCK FLAG SET?
(1)  002154  100404                        BMI     B0006           ;IF SO, CONTINUE ON WITH THE TEST
(1)  002156  005203                        INC     R3              ;IF NOT INCREMENT THE COUNTER LOCATION
(1)  002160  001373                        BNE     A0006           ;AND GO TEST THE CLOCK FLAG AGAIN. UNLESS...
(1)  002162  104001                E0006:   ERROR 1                 ;CLOCK FLAG DID NOT SET AFTER A WAITING PERIOD > 20 MS
(1)  002164  000412                        BR      T0007
(1)  002166                        B0006:
181  002166  012737  000100  177546        MOV     #100,LKS        ;CLEAR CLOCK FLAG AND SET INTERRUPT ENABLE
182  002174  005037  177546                CLR     LKS             ;CLEAR INTERRUPT ENABLE
183  002200  032737  000100  177546        BIT     #100,LKS        ;TEST THE INTERRUPT ENABLE BIT
184  002206  001401                        BEQ     T0007           ;IS INTERRUPT ENABLE CLEARED
185  002210  104001                E10006:  ERROR 1                 ;ERROR, ERROR INTERRUPT BIT CAN NOT BE CLEARED
186
187
188
189                                .SBTTL  TEST THAT CLOCK INTERRUPTS TO CORRECT VECTOR ADDRESS
190                                ;TEST THAT CLOCK INTERRUPTS TO CORRECT VECTOR ADDRESS
191  002212  000004                T0007:   SCOPE
192  002214  012737  005674  000004        MOV     #TRAP0,@#4      ;SETUP VECOR IN CASE OF UNFORSEEN PROBLEMS
(1)  002222  012737  000340  000006        MOV     #340,@#6        ;NO INTERRUPTS WHILE PRINTING FATAL MESSAGE
193  002230  012737  000300  001124        MOV     #300,$GOOAT     ;HAVE GOOD DATA INFO READY FOR TYPEOUT IN CASE OF AN ERR
194  002236  012737  002260  001106        MOV     #R0007,$LPADR   ;INITIALIZE THE LOOPBACK ADDRESS IN CASE OF AN ERROR
195  002244  012737  002340  000100        MOV     #B0007,100      ;SET UP VECTOR RETURN POINTER
196  002252  012737  000340  000102        MOV     #340,@#102      ;1 INTERRUPT IS ENOUGH
197  002260  012706  001000        R0007:  MOV     #1000,SP        ;GET STACK READY FOR INTERRUPTS
198  002264  012737  000200  177776        MOV     #200,PS         ;SET PROCESSOR PRIORITY 4
199  002272  005037  177546                CLR     LKS
200  002276  005003                        CLR     R3              ;INITIALIZE A COUNTER LOCATION
(1)  002300  105737  177546        A0007:  TSTB    LKS             ;IS THE CLOCK FLAG SET?
(1)  002304  100404                        BMI     B0007           ;IF SO, CONTINUE ON WITH THE TEST
(1)  002306  005203                        INC     R3              ;IF NOT INCREMENT THE COUNTER LOCATION
(1)  002310  001373                        BNE     A0007           ;AND GO TEST THE CLOCK FLAG AGAIN. UNLESS...
```

```
(1)  002312  104001                    E0007:  ERROR 1                 ;CLOCK FLAG DID NOT SET AFTER A WAITING PERIOD > 20 MS
(1)  002314  000415                            BR      T0010
(1)  002316                             B0007:
201  002316  012737  000100  177546            MOV     #100,LKS        ;ENABLE INTERRUPT
202  002324  005000                            CLR     R0
203  002326  005200                    C0007:  INC     R0
204  002330  000240                            NOP                     ;STALL FOR TIME
205  002332  001375                            BNE     C0007           ;WAIT FOR INTERRUPT
206  002334  104002                    E10007: ERROR 2                 ;ERROR, DIDNT GET INTERRUPT
207  002336  000404                            BR      T0010
208  002340  105737  177546            D0007:  TSTB    LKS             ;ENTER HERE IF INTERRUPTED
209  002344  100401                            BMI     T0010
210  002346  104001                    E20007: ERROR 1                 ;ERROR, INTERRUPT NOT CAUSED BY CLOCK
211
212
213
214                                     .SBTTL  TEST THAT CLOCK WILL INTERRUPT WITH PROCESSOR AT PRIORITY 5
215                                     ;TEST THAT CLOCK WILL INTERRUPT WITH PROCESSOR AT PRIORITY 5
216  002350  000004                    T0010:  SCOPE
217  002352  012737  005674  000004            MOV     #TRAP0,@#4      ;SETUP VECOR IN CASE OF UNFORSEEN PROBLEMS
(1)  002360  012737  000340  000006            MOV     #340,@#6        ;NO INTERRUPTS WHILE PRINTING FATAL MESSAGE
218  002366  012737  002410  001106            MOV     #R0010,$LPADR   ;INITIALIZE THE LOOPBACK ADDRESS IN CASE OF AN ERROR
219  002374  012737  002470  000100            MOV     #D0010,100      ;SET UP VECTOR RETURN POINTER
220  002402  012737  000340  000102            MOV     #340,@#102      ;NO INTERRUPTS ALLOWED AFTER THE FIRST ONE
221  002410  005037  177546            R0010:  CLR     LKS
222  002414  012737  000240  177776            MOV     #240,PS         ;SET PRIORITY 5
223  002422  012706  001000                    MOV     #1000,SP        ;INITIALIZE THE STACK POINTER
224  002426  005003                            CLR     R3              ;INITIALIZE A COUNTER LOCATION
(1)  002430  105737  177546            A0010:  TSTB    LKS             ;IS THE CLOCK FLAG SET?
(1)  002434  100404                            BMI     B0010           ;IF SO, CONTINUE ON WITH THE TEST
(1)  002436  005203                            INC     R3              ;IF NOT INCREMENT THE COUNTER LOCATION
(1)  002440  001373                            BNE     A0010           ;AND GO TEST THE CLOCK FLAG AGAIN. UNLESS...
(1)  002442  104001                    E0010:  ERROR 1                 ;CLOCK FLAG DID NOT SET AFTER A WAITING PERIOD > 20 MS
(1)  002444  000415                            BR      T0011
(1)  002446                             B0010:
225  002446  012737  000100  177546            MOV     #100,LKS        ;ENABLE INTERRUPT
226  002454  005000                            CLR     R0
227  002456  005200                    C0010:  INC     R0
228  002460  000240                            NOP                     ;STALL FOR SOME TIME
229  002462  001375                            BNE     C0010           ;WAIT FOR INTERRUPT
230  002464  104002                    E10010: ERROR 2                 ;ERROR, INTERRUPT FAILED TO OCCUR
231  002466  000404                            BR      T0011
232  002470  105737  177546            D0010:  TSTB    LKS             ;ENTER HERE IF INTERRUPTED
233  002474  100401                            BMI     T0011
234  002476  104001                    E20010: ERROR 1                 ;ERROR, INTERRUPTDID NOT CLEAR THE CLOCK FLAG
235
236
237
238                                     .SBTTL  TEST THAT CLOCK WILL NOT INTERRUPT WITH PROCESSOR PRIORITY 6
239                                     ;TEST THAT CLOCK WILL NOT INTERRUPT WITH PROCESSOR PRIORITY 6
240  002500  000004                    T0011:  SCOPE
241  002502  012737  005674  000004            MOV     #TRAP0,@#4      ;SETUP VECOR IN CASE OF UNFORSEEN PROBLEMS
(1)  002510  012737  000340  000006            MOV     #340,@#6        ;NO INTERRUPTS WHILE PRINTING FATAL MESSAGE
242  002516  012737  002524  001106            MOV     #R0011,$LPADR   ;SETUP LOOPBACK ADDRESS IN CASE OF AN ERROR
```

```
                                R0011:  CLR     LKS
                                        CLR     R3              ;INITIALIZE A COUNTER LOCATION
(1)  002536  100404             A0011:  TSTB    LKS             ;IS THE CLOCK FLAG SET?
(1)  002536  100404                     BMI     B0011           ;IF SO, CONTINUE ON WITH THE TEST
(1)  002540  005203                     INC     R3              ;IF NOT INCREMENT THE COUNTER LOCATION
(1)  002542  001373                     BNE     A0011           ;AND GO TEST THE CLOCK FLAG AGAIN. UNLESS...
(1)  002544  104001             E0011:  ERROR 1                 ;CLOCK FLAG DID NOT SET AFTER A WAITING PERIOD > 20 MS
(1)  002546  000427                     BR      T0012
(1)  002550                     B0011:
002550  012706  001000                    MOV     #1000,SP        ;INITIALIZE THE STACK POINTER
002554  012737  000300  177776            MOV     #300,PS         ;SET PRIORITY 6
002562  012737  002612  000100            MOV     #E1011,100      ;SET UP VECTOR RETURN
002570  012737  000100  177546            MOV     #100,LKS        ;ENABLE INTERRUPT
002576  005003                            CLR     R3              ;INITIALIZE A COUNTER LOCATION
(1)  002600  105737  177546     C0011:  TSTB    LKS             ;IS THE CLOCK FLAG SET?
(1)  002604  100404                     BMI     D0011           ;IF SO, CONTINUE ON WITH THE TEST
(1)  002606  005203                     INC     R3              ;IF NOT INCREMENT THE COUNTER LOCATION
(1)  002610  001373                     BNE     C0011           ;AND GO TEST THE CLOCK FLAG AGAIN. UNLESS...
(1)  002612  104001             E1011:  ERROR 1                 ;CLOCK FLAG DID NOT SET AFTER A WAITING PERIOD > 20 MS
(1)  002614  000731                     BR      T0011
(1)  002616                     D0011:
002616  000240                            NOP
002620  000240                            NOP                     ;GIVE CLOCK EXTRA TIME TO INTERRUPT
002622  000401                            BR      T0012
002624  104003             E20011: ERROR 3                 ;ERROR, CLOCK INTERRUPTED WITHOUT HAVING PRIORITY
```

```
                                .SBTTL  TEST THAT RESET SETS CLOCK FLAG
                                ;TEST THAT RESET SETS CLOCK FLAG
002626  000004                 T0012:  SCOPE
002630  012737  000200  001124            MOV     #200,$GDDAT     ;HAVE GOOD DATA INFO READY FOR TYPEOUT IN CASE OF AN ERR
002636  012737  005674  000004            MOV     #TRAP0,@#4      ;SETUP VECOR IN CASE OF UNFORSEEN PROBLEMS
(1)  002644  012737  000340  000006       MOV     #340,@#6        ;NO INTERRUPTS WHILE PRINTING FATAL MESSAGE
002652  012737  002660  001106            MOV     #R0012,$LPADR   ;SETUP LOOPBACK ADDRESS IN CASE OF AN ERROR
002660  005037  177546     R0012:  CLR     LKS             ;CLEAR CLOCK FLAG
002664  005003                            CLR     R3              ;INITIALIZE A COUNTER LOCATION
(1)  002666  105737  177546     A0012:  TSTB    LKS             ;IS THE CLOCK FLAG SET?
(1)  002672  100404                     BMI     B0012           ;IF SO, CONTINUE ON WITH THE TEST
(1)  002674  005203                     INC     R3              ;IF NOT INCREMENT THE COUNTER LOCATION
(1)  002676  001373                     BNE     A0012           ;AND GO TEST THE CLOCK FLAG AGAIN. UNLESS...
(1)  002700  104001             E0012:  ERROR 1                 ;CLOCK FLAG DID NOT SET AFTER A WAITING PERIOD > 20 MS
(1)  002702  000407                     BR      T0013
(1)  002704                     B0012:
002704  005037  177546                    CLR     LKS
002710  000005                            RESET                   ;SHOULD SET CLOCK FLAG
002712  105737  177546                    TSTB    LKS
002716  100401                            BMI     T0013
002720  104001             E10012: ERROR 1                 ;ERROR, RESET DIDN'T SET CLOCK FLAG
```

```
                                .SBTTL  TEST LINE CLOCK REPEATABILITY
                                ;TEST LINE CLOCK REPEATABILITY
```

```
                                        ;MAKE SURE THAT OVER TWO EQUAL PERIODS OF TIME
                                        ;THE CLOCK PUTS OUT THE SAME NUMBER OF PULSES
002722  000004                  T0013:  SCOPE
002724  005000                  R0013:  CLR     R0              ;CLEAR 1ST TIME COUNT
002726  005001                          CLR     R1              ;CLEAR 1ST CLOCK COUNT
002730  012737  000340  177776          MOV     #340,PS         ;SET PRIORITY 7
002736  012737  002736  001106  R1013:  MOV     #R1013,$LPADR   ;ERROR IN NEXT  FEW INSTRUCTIONS CAUSES A SHORT SCOPE LO
002744  005037  177546                  CLR     LKS
                                                                ;SYNC ON CLOCK FLAG A COUPLE OF TIMES
002750  005003                          CLR     R3              ;INITIALIZE A COUNTER LOCATION
(1) 002752  105737  177546      A0013:  TSTB    LKS             ;IS THE CLOCK FLAG SET?
(1) 002756  100404                      BMI     B0013           ;IF SO, CONTINUE ON WITH THE TEST
(1) 002760  005203                      INC     R3              ;IF NOT INCREMENT THE COUNTER LOCATION
(1) 002762  001373                      BNE     A0013           ;AND GO TEST THE CLOCK FLAG AGAIN. UNLESS...
(1) 002764  104001              E0013:  ERROR 1                 ;CLOCK FLAG DID NOT SET AFTER A WAITING PERIOD > 20 MS
(1) 002766  000510                      BR      T0014
(1) 002770                      B0013:
002770  012737  002770  001106  R2013:  MOV     #R2013,$LPADR   ;MAKE SCOPE LOOP SHORT IN CASE OF AN ERROR
002776  005037  177546                  CLR     LKS
003002  005003                          CLR     R3              ;INITIALIZE A COUNTER LOCATION
(1) 003004  105737  177546      C0013:  TSTB    LKS             ;IS THE CLOCK FLAG SET?
(1) 003010  100404                      BMI     D0013           ;IF SO, CONTINUE ON WITH THE TEST
(1) 003012  005203                      INC     R3              ;IF NOT INCREMENT THE COUNTER LOCATION
(1) 003014  001373                      BNE     C0013           ;AND GO TEST THE CLOCK FLAG AGAIN. UNLESS...
(1) 003016  104001              E1013:  ERROR 1                 ;CLOCK FLAG DID NOT SET AFTER A WAITING PERIOD > 20 MS
(1) 003020  000473                      BR      T0014
(1) 003022                      D0013:
003022  005037  177546                  CLR     LKS
003026  105737  177546      F0013:  TSTB    LKS             ;IS CLOCK FLAG SET
003032  100003                          BPL     G0013           ;NO
003034  005201                          INC     R1              ;+1 TO CLOCK COUNT
003036  005037  177546                  CLR     LKS             ;CLEAR CLOCK IF SET
003042  005200                  G0013:  INC     R0              ;+1 TO TIME COUNT
003044  001370                          BNE     F0013           ;REPEAT UNTIL R0=0
003046  005000                          CLR     R0              ;CLEAR 2ND TIME COUNT
003050  005002                          CLR     R2              ;CLEAR 2ND CLOCK COUNT
003052  012737  003052  001106  R3013:  MOV     #R3013,$LPADR   ;INSURE A SHORT SCOPE LOOP
003060  005037  177546                  CLR     LKS
                                                                ;SYNC ON CLOCK FLAG TWICE
003064  005003                          CLR     R3              ;INITIALIZE A COUNTER LOCATION
(1) 003066  105737  177546      H0013:  TSTB    LKS             ;IS THE CLOCK FLAG SET?
(1) 003072  100404                      BMI     J0013           ;IF SO, CONTINUE ON WITH THE TEST
(1) 003074  005203                      INC     R3              ;IF NOT INCREMENT THE COUNTER LOCATION
(1) 003076  001373                      BNE     H0013           ;AND GO TEST THE CLOCK FLAG AGAIN. UNLESS...
(1) 003100  104001              E2013:  ERROR 1                 ;CLOCK FLAG DID NOT SET AFTER A WAITING PERIOD > 20 MS
(1) 003102  000442                      BR      T0014
(1) 003104                      J0013:
003104  012737  003104  001106  R4013:  MOV     #R4013,$LPADR   ;INSURE A SHORT SCOPE LOOP
003112  005037  177546                  CLR     LKS
003116  005003                          CLR     R3              ;INITIALIZE A COUNTER LOCATION
(1) 003120  105737  177546      K0013:  TSTB    LKS             ;IS THE CLOCK FLAG SET?
(1) 003124  100404                      BMI     L0013           ;IF SO, CONTINUE ON WITH THE TEST
(1) 003126  005203                      INC     R3              ;IF NOT INCREMENT THE COUNTER LOCATION
(1) 003130  001373                      BNE     K0013           ;AND GO TEST THE CLOCK FLAG AGAIN. UNLESS...
```

```
(1)  003132  104001                  E30013: ERROR 1                 ;CLOCK FLAG DID NOT SET AFTER A WAITING PERIOD > 20 MS
(1)  003134  000425                          BR      TO014
(1)  003136                          LO013:
304  003136  012737  002724  001106          MOV     #RO013,$LPADR   ;MUST LOOP BACK TO BEGINING OF TEST IF EEROR COMES NOW
305  003144  005037  177546                  CLR     LKS
306  003150  105737  177546          MO013:  TSTB    LKS             ;IS CLOCK FLAG SET
307  003154  100003                          BPL     NO013           ;NO
308  003156  005202                          INC     R2              ;+1 TO CLOCK COUNT
309  003160  005037  177546                  CLR     LKS             ;CLEAR CLOCK IF SET
310  003164  005200                  NO013:  INC     R0              ;+1 TO TIME COUNT
311  003166  001370                          BNE     MO013           ;REPEAT UNTIL R0=0
312  003170  020102                          CMP     R1,R2           ;IS 1ST CLOCK COUNT EQUAL TO 2ND CLOCK COUNT?
313  003172  001406                          BEQ     TO014           ;YES
314  003174  010137  001156          E40013: MOV     R1,$REG1        ;GET R1 READY FOR PRINTOUT
315  003200  010237  001160                  MOV     R2,$REG2        ;GET R2 READY FOR PRINTOUT
316  003204  104004                          ERROR 4                 ;ERROR, CLOCK FLAG OCCURRED DIFFERENT
317  003206  000240                          NOP                     ;NUMBER OF TIMES IN EQUAL PERIODS
318
319
320
321                                  .SBTTL  LINE CLOCK REGISTER ADDRESSING TEST
322                                  ;LINE CLOCK REGISTER ADDRESSING TEST
323                                  ;TEST THAT THE "LKS" REGISTER CAN NOT BE ADDRESSED AS ANYTHING BUT 177546
324                                  ;SET A LOCATION THAT IS CLOSE(DIFFERS BY 1 BIT) TO THE LKS REGISTER
325                                  ;TO 100. IF THE LKS ALSO CHANGES, THEN SIGNAL AN ERROR
326  003210  000004                  TO014:  SCOPE
327  003212  005037  001124                  CLR     $GDDAT
328  003216  012737  003254  001106          MOV     #RO014,$LPADR   ;INITIALIZE THE LOOPBACK ADDRESS IN CASE OF AN ERROR
329  003224  012737  157546  001120          MOV     #157546,$GDADR  ;SAME AS "LKS" ADDRESS EXCEPT WITH BIT 13 CLEAR
330  003232  012737  003272  000004          MOV     #A0014,4        ;SETUP VECTOR IN CASE IT IS NONEXISTANT
331  003240  012737  000340  000006          MOV     #340,6
332  003246  012737  000340  177776          MOV     #340,@#PS       ;NO INTERRUPTS NOW
333  003254  012706  001000          RO014:  MOV     #1000,SP        ;SETUP THE STACK
334  003260  005037  177546                  CLR     LKS
335  003264  012777  000100  175626  IO014:  MOV     #100,@$GDADR    ;SET THE "CLOSE" ADDRESS TO = 100
336  003272  032737  000100  177546  A0014:  BIT     #100,LKS        ;MAKE SURE THAT "LKS" WAS NOT AFFECTED
337  003300  001401                          BEQ     B0014
338  003302  104005                  E0014:  ERROR 5                 ;IT AFFECTED "LKS"  --  ERROR
339  003304  005037  177546          B0014:  CLR     LKS
340  003310  012737  003322  000004          MOV     #T0015  ,4
341  003316  005077  175576                  CLR     @$GDADR         ;CLEAR OUT THE "CLOSE" ADDRESS
342
343
344
345                                  .SBTTL  LINE CLOCK REGISTER ADDRESSING TEST
346                                  ;LINE CLOCK REGISTER ADDRESSING TEST
347  003322  000004                  T0015:  SCOPE
348  003324  012737  003404  000004          MOV     #A0015,4        ;SETUP VECTOR IN CASE IT IS NONEXISTANT
349  003332  012737  000340  000006          MOV     #340,6
350  003340  005037  001124                  CLR     $GDDAT
351  003344  012737  003366  001106          MOV     #RO015,$LPADR   ;INITIALIZE THE LOOPBACK ADDRESS IN CASE OF AN ERROR
352  003352  012737  177146  001120          MOV     #177146,$GDADR  ;SAME AS "LKS" ADDRESS EXCEPT WITH BIT 8 CLEAR
353  003360  012737  000340  177776          MOV     #340,@#PS       ;NO INTERRUPTS NOW
354  003366  012706  001000          RO015:  MOV     #1000,SP        ;SETUP THE STACK
```

```
                                                CLR     LKS
003376  012777  000100  175514  I0015:  MOV     #100,@SGDADR    ;SET THE "CLOSE" ADDRESS TO = 100
003404  032737  000100  177546  A0015:  BIT     #100,LKS        ;MAKE SURE THAT "LKS" WAS NOT AFFECTED
003412  001401                          BEQ     B0015
003414  104005                  E0015:  ERROR 5                 ;IT AFFECTED "LKS"  --  ERROR
003416  005037  177546          B0015:  CLR     LKS
003422  012737  003434  000004          MOV     #T0016  ,4
003430  005077  175464                  CLR     @SGDADR         ;CLEAR OUT THE "CLOSE" ADDRESS
                                .SBTTL  LINE CLOCK REGISTER ADDRESSING TEST
                                ;LINE CLOCK REGISTER ADDRESSING TEST
003434  000004                  T0016:  SCOPE
003436  012737  003516  000004          MOV     #A0016,4        ;SETUP VECTOR IN CASE IT IS NONEXISTANT
003444  012737  000340  000006          MOV     #340,6
003452  005037  001124                  CLR     $GDDAT
003456  012737  003500  001106          MOV     #R0016,$LPADR   ;INITIALIZE THE LOOPBACK ADDRESS IN CASE OF AN ERROR
003464  012737  177446  001120          MOV     #177446,$GDADR  ;SAME AS "LKS" ADDRESS EXCEPT WITH BIT 6 CLEAR
003472  012737  000340  177776          MOV     #340,@#PS       ;NO INTERRUPTS NOW
003500  012706  001000          R0016:  MOV     #1000,SP        ;SETUP THE STACK
003504  005037  177546                  CLR     LKS
003510  012777  000100  175402  I0016:  MOV     #100,@SGDADR    ;SET THE "CLOSE" ADDRESS TO = 100
003516  032737  000100  177546  A0016:  BIT     #100,LKS        ;MAKE SURE THAT "LKS" WAS NOT AFFECTED
003524  001401                          BEQ     B0016
003526  104005                  E0016:  ERROR 5                 ;IT AFFECTED "LKS"  --  ERROR
003530  005037  177546          B0016:  CLR     LKS
003534  012737  003546  000004          MOV     #T0017  ,4
003542  005077  175352                  CLR     @SGDADR         ;CLEAR OUT THE "CLOSE" ADDRESS
                                .SBTTL  LINE CLOCK REGISTER ADDRESSING TEST
                                ;LINE CLOCK REGISTER ADDRESSING TEST
003546  000004                  T0017:  SCOPE
003550  012737  003630  000004          MOV     #A0017,4        ;SETUP VECTOR IN CASE IT IS NONEXISTANT
003556  012737  000340  000006          MOV     #340,6
003564  005037  001124                  CLR     $GDDAT
003570  012737  003612  001106          MOV     #R0017,$LPADR   ;INITIALIZE THE LOOPBACK ADDRESS IN CASE OF AN ERROR
003576  012737  177556  001120          MOV     #177556,$GDADR  ;SAME AS "LKS" ADDRESS EXCEPT WITH BIT 3 SET
003604  012737  000340  177776          MOV     #340,@#PS       ;NO INTERRUPTS NOW
003612  012706  001000          R0017:  MOV     #1000,SP        ;SETUP THE STACK
003616  005037  177546                  CLR     LKS
003622  012777  000100  175270  I0017:  MOV     #100,@SGDADR    ;SET THE "CLOSE" ADDRESS TO = 100
003630  032737  000100  177546  A0017:  BIT     #100,LKS        ;MAKE SURE THAT "LKS" WAS NOT AFFECTED
003636  001401                          BEQ     B0017
003640  104005                  E0017:  ERROR 5                 ;IT AFFECTED "LKS"  --  ERROR
003642  005037  177546          B0017:  CLR     LKS
003646  012737  003660  000004          MOV     #T0020  ,4
003654  005077  175240                  CLR     @SGDADR         ;CLEAR OUT THE "CLOSE" ADDRESS
                                .SBTTL  LINE CLOCK REGISTER ADDRESSING TEST
```

```
                                                ;LINE CLOCK REGISTER ADDRESSING TEST
003660  000004                          T0020:  SCOPE
003662  012737  003742  000004                  MOV     #A0020,4        ;SETUP VECTOR IN CASE IT IS NONEXISTANT
003670  012737  000340  000006                  MOV     #340,6
003676  005037  001124                          CLR     $GDDAT
003702  012737  003724  001106                  MOV     #R0020,$LPADR   ;INITIALIZE THE LOOPBACK ADDRESS IN CASE OF AN ERROR
003710  012737  177566  001120                  MOV     #177566,$GDADR  ;SAME AS "LKS" ADDRESS EXCEPT WITH BIT 4 SET
003716  012737  000340  177776                  MOV     #340,@#PS       ;NO INTERRUPTS NOW
003724  012706  001000                  R0020:  MOV     #1000,SP        ;SETUP THE STACK
003730  005037  177546                          CLR     LKS
003734  012777  000100  175156          I0020:  MOV     #100,@$GDADR    ;SET THE "CLOSE" ADDRESS TO = 100
003742  032737  000100  177546          A0020:  BIT     #100,LKS        ;MAKE SURE THAT "LKS" WAS NOT AFFECTED
003750  001401                                  BEQ     B0020
003752  104005                          E0020:  ERROR 5                 ;IT AFFECTED "LKS"  --  ERROR
003754  005037  177546                  B0020:  CLR     LKS
003760  012737  003772  000004                  MOV     #T0021  ,4
003766  005077  175126                          CLR     @$GDADR         ;CLEAR OUT THE "CLOSE" ADDRESS
                                                .SBTTL  LINE CLOCK REGISTER ADDRESSING TEST
                                                ;LINE CLOCK REGISTER ADDRESSING TEST
003772  000004                          T0021:  SCOPE
003774  012737  004054  000004                  MOV     #A0021,4        ;SETUP VECTOR IN CASE IT IS NONEXISTANT
004002  012737  000340  000006                  MOV     #340,6
004010  005037  001124                          CLR     $GDDAT
004014  012737  004036  001106                  MOV     #R0021,$LPADR   ;INITIALIZE THE LOOPBACK ADDRESS IN CASE OF AN ERROR
004022  012737  177746  001120                  MOV     #177746,$GDADR  ;SAME AS "LKS" ADDRESS EXCEPT WITH BIT 7 SET
004030  012737  000340  177776                  MOV     #340,@#PS       ;NO INTERRUPTS NOW
004036  012706  001000                  R0021:  MOV     #1000,SP        ;SETUP THE STACK
004042  005037  177546                          CLR     LKS
004046  012777  000100  175044          I0021:  MOV     #100,@$GDADR    ;SET THE "CLOSE" ADDRESS TO = 100
004054  032737  000100  177546          A0021:  BIT     #100,LKS        ;MAKE SURE THAT "LKS" WAS NOT AFFECTED
004062  001401                                  BEQ     B0021
004064  104005                          E0021:  ERROR 5                 ;IT AFFECTED "LKS"  --  ERROR
004066  005037  177546                  B0021:  CLR     LKS
004072  012737  004104  000004                  MOV     #T0022  ,4
004100  005077  175014                          CLR     @$GDADR         ;CLEAR OUT THE "CLOSE" ADDRESS
                                                .SBTTL  LINE CLOCK REGISTER HIGH BYTE TEST
                                                ;LINE CLOCK REGISTER HIGH BYTE TEST
                                                ;MAKE SURE THE LKS REGISTER LOW BYTE RESPONDS TO THE HIGH BYTES ADDRESS
004104  000004                          T0022:  SCOPE
004106  012737  005674  000004                  MOV     #TRAP0,@#4      ;SETUP VECOR IN CASE OF UNFORSEEN PROBLEMS
(1) 004114  012737  000340  000006              MOV     #340,@#6        ;NO INTERRUPTS WHILE PRINTING FATAL MESSAGE
004122  012737  000100  001124                  MOV     #100,$GDDAT
004130  012737  004152  001106                  MOV     #R0022,$LPADR   ;INITIALIZE THE LOOPBACK ADDRESS IN CASE OF AN ERROR
004136  012737  177547  001120                  MOV     #177547,$GDADR  ;HIGH BYTE OF THE LKS REGISTER
004144  012737  000340  177776                  MOV     #340,@#PS       ;NO INTERRUPTS NOW
004152  012706  001000                  R0022:  MOV     #1000,SP        ;SETUP THE STACK
004156  005037  177546                          CLR     LKS
004162  112777  000100  174730          I0022:  MOVB    #100,@$GDADR    ;SET THE HIGH BYTE ADDRESS TO = 100
```

```
                                                BIT     #100,LKS        ;MAKE SURE THAT "LKS" LOW BYTE WAS AFFECTED
                                                BNE     A0022
                                        E0022:  ERROR 5                 ;SHOULD HAVE SET BIT 7.  ERROR
                                        A0022:  CLR     LKS
                                                RESET


                                        .SBTTL  CLOCK FLAG BIT TEST
                                        ;CLOCK FLAG BIT TEST
                                        T0023:  SCOPE
                                                MOV     #TRAP0,@#4      ;SETUP VECOR IN CASE OF UNFORSEEN PROBLEMS
(1)                                             MOV     #340,@#6        ;NO INTERRUPTS WHILE PRINTING FATAL MESSAGE
                                                MOV     #200,$GDDAT
                                                MOV     #R0023,$LPADR   ;INITIALIZE THE LOOPBACK ADDRESS IN CASE OF AN ERROR
                                        R0023:  CLR     LKS
                                                CLR     R3              ;INITIALIZE A COUNTER LOCATION
(1)                                     A0023:  TSTB    LKS             ;IS THE CLOCK FLAG SET?
(1)                                             BMI     B0023           ;IF SO, CONTINUE ON WITH THE TEST
(1)                                             INC     R3              ;IF NOT INCREMENT THE COUNTER LOCATION
(1)                                             BNE     A0023           ;AND GO TEST THE CLOCK FLAG AGAIN. UNLESS...
(1)                                     E0023:  ERROR 1                 ;CLOCK FLAG DID NOT SET AFTER A WAITING PERIOD > 20 MS
(1)                                             BR      T0024
(1)                                     B0023:
                                        I0023:  MOV     #200,LKS        ;MOVE A 1 INTO THE CLOCK FLAG BIT
                                                CMP     LKS,$GDDAT      ;SHOULD NOT AFFECT THE FLAG BIT
                                                BEQ     T0024
                                        E10023: ERROR 1                 ;CLOCK FLAG DID NOT CLEAR


                                        .SBTTL  INTERRUPT TEST
                                        T0024:  SCOPE
                                                MOV     #TRAP0,@#4      ;SETUP VECOR IN CASE OF UNFORSEEN PROBLEMS
(1)                                             MOV     #340,@#6        ;NO INTERRUPTS WHILE PRINTING FATAL MESSAGE
                                                MOV     #200,$GDDAT     ;HAVE GOOD DATA INFO READY FOR TYPEOUT IN CASE OF AN ERR
                                                MOV     #R0024,$LPADR   ;INITIALIZE THE LOOPBACK ADDRESS IN CASE OF AN ERROR
                                                MOV     #E0024,100
                                                CLR     LKS             ;ALLOW CLOCK INTERRUPTS
                                        R0024:  MOV     #340,@#PS       ;NO INTERRUPTS NOW
                                                MOV     #300,LKS
                                        A0024:  INC     #0              ;WAIT FOR 20+ MS
                                                BNE     A0024           ;LOOP BACK IF NOT DONE WAITING
                                                RESET                   ;RESET SHOULD CLEAR INTERRUPT ENABLE
                                                CMP     $GDDAT,LKS      ;AND LEAVE THE CLOCK FLAG SET
                                                BEQ     T0025           ;GO ON TO THE NEXT TEST IF IT DID
                                        E0024:  ERROR 1                 ;RESET SET INTERRUPT ENABLE OR CLEARED CLOCK FLAG


                                        .SBTTL  NO SACK TIMEOUT TEST
                                        ;NO SACK TIMEOUT TEST
                                        T0025:  SCOPE
                                                MOV     #TRAP0,@#4      ;SETUP VECOR IN CASE OF UNFORSEEN PROBLEMS
```

```
   (1)  004420  012737  000340  000006          MOV     #340,@#6        ;NO INTERRUPTS WHILE PRINTING FATAL MESSAGE
   507  004426  012737  000300  001124          MOV     #300,$GDDAT     ;HAVE GOOD DATA INFO READY FOR TYPEOUT IN CASE OF AN ERR
   508  004434  012737  004456  001106          MOV     #R0025,$LPADR   ;INITIALIZE THE LOOPBACK ADDRESS IN CASE OF AN ERROR
   509  004442  012737  000340  000102          MOV     #340,102        ;NO INTERRUPTS AFTER THE FIRST ONE
   510  004450  012737  004530  000100          MOV     #C0025,100
   511  004456  005037  177546          R0025:  CLR     LKS
   512  004462  012737  000360  177776          MOV     #360,PS
   513  004470  005003                          CLR     R3              ;INITIALIZE A COUNTER LOCATION
   (1)  004472  105737  177546          A0025:  TSTB    LKS             ;IS THE CLOCK FLAG SET?
   (1)  004476  100404                          BMI     B0025           ;IF SO, CONTINUE ON WITH THE TEST
   (1)  004500  005203                          INC     R3              ;IF NOT INCREMENT THE COUNTER LOCATION
   (1)  004502  001373                          BNE     A0025           ;AND GO TEST THE CLOCK FLAG AGAIN. UNLESS...
   (1)  004504  104001          E0025:  ERROR 1                         ;CLOCK FLAG DID NOT SET AFTER A WAITING PERIOD > 20 MS
   (1)  004506  000415                          BR      T0026
   (1)  004510                  B0025:
   514  004510  005037  177776                  CLR     PS              ;MAKE ALL INTERRUPTS OK
   515  004514  012737  000100  177546          MOV     #100,LKS        ;ENABLE CLOCK INTERRUPTS
   516  004522  000001                          WAIT                    ;THE ONLY WAY TO LEAVE HERE WITHOUT ERROR IS TO INTERRUP
   517  004524  104006          E1002S: ERROR 6                         ;IF IT ERROR IS HERE ITS BECAUSE OF A "NO-SACK" TIMEOUT
   518  004526  000405                          BR      T0026
   519  004530  022737  000300  177546  C0025:  CMP     #300,LKS
   520  004536  001401                          BEQ     T0026           ;FIND OUT WHAT THE INTERRUPT DID TO THE CLOCK STATUS REG
   521  004540  104001          E2002S: ERROR 1                         ;IT CLEARED THE INTERRUPT ENABLE OR THE FLAG BIT
   522
   523
   524
   525                                  .SBTTL  RESET TEST
   526                                  ;RESET TEST
   527  004542  000004          T0026:  SCOPE
   528  004544  012737  005674  000004          MOV     #TRAP0,@#4      ;SETUP VECOR IN CASE OF UNFORSEEN PROBLEMS
   (1)  004552  012737  000340  000006          MOV     #340,@#6        ;NO INTERRUPTS WHILE PRINTING FATAL MESSAGE
   529  004560  012737  000200  001124          MOV     #200,$GDDAT     ;HAVE GOOD DATA INFO READY FOR TYPEOUT IN CASE OF AN ERR
   530  004566  012737  004616  001106          MOV     #R0026,$LPADR   ;INITIALIZE THE LOOPBACK ADDRESS IN CASE OF AN ERROR
   531  004574  012737  000340  177776          MOV     #340,@#PS       ;NO INTERRUPTS NOW
   532  004602  012737  004654  000100          MOV     #E0026,100
   533  004610  012737  000140  000102          MOV     #140,102        ;SETUP STATUS FOR AFTER THE INTERRUPT
   534  004616  005037  177546          R0026:  CLR     LKS
   535  004622  012706  001000                  MOV     #1000,SP        ;SETUP THE STACK
   536  004626  012737  000100  177546          MOV     #100,LKS        ;SET INTERRUPT ENABLE BIT NOW
   537  004634  005227  000000          A0026:  INC     #0              ;WAIT FOR CLOCK FLAG TO SET
   538  004640  001375                          BNE     A0026
   539  004642  000005          I0026:  RESET                           ;RESET SHOULD NOT CLEAR THE FLAG
   540  004644  023737  177546  001124          CMP     LKS,$GDDAT      ;FIND OUT IF ID DIT DID OR NOT
   541  004652  001401                          BEQ     T0027
   542  004654  104001          E0026:  ERROR 1                         ;RESET DID NOT INITIALIZE THE LKS WORD CORRECTLY
   543
   544
   545
   546                                  .SBTTL  CLOCK FLAG BIT TEST
   547                                  ;CLOCK FLAG BIT TEST
   548                                  ;MAKE SURE IT DOESNT CLEAR WHEN YOU TRY TO SET IT VIA A 'MOV' INSTRUCTION
   549  004656  000004          T0027:  SCOPE
   550  004660  012737  005674  000004          MOV     #TRAP0,@#4      ;SETUP VECOR IN CASE OF UNFORSEEN PROBLEMS
   (1)  004666  012737  000340  000006          MOV     #340,@#6        ;NO INTERRUPTS WHILE PRINTING FATAL MESSAGE
```

```
                                                MOV     #200,$GDDAT     ;HAVE GOOD DATA INFO READY FOR TYPEOUT IN CASE OF AN ERR
                                                MOV     #R0027,$LPADR   ;INITIALIZE THE LOOPBACK ADDRESS IN CASE OF AN ERROR
                                                CLR     102
                                                MOV     #T0030  ,100
                                        R0027:  CLR     LKS
                                                MOV     #340,PS
                                                MOV     #1000,SP        ;SETUP THE STACK
                                                CLR     WORD            ;SETUP A COUNTER LOCATION TO = 0
                                        A0027:  INC     WORD
                                                BNE     A0027           ;WASTE TIME LOOPING UNTIL THE COUNTER REACHES 0
                                                MOV     #300,LKS        ;SET INTERRUPT ENABLE AND TRY TO SET THE CLOCK FLAG
                                                MOV     #200,LKS        ;TRY TO SET IT AGAIN
                                                CMP     LKS,$GDDAT      ;DID THE CLOCK FLAG STAY SET?
                                                BEQ     T0030           ;IF NOT GO ON TO THE NEXT TEST
                                        E0027:  ERROR 1                 ;ERROR - MOVED A '1' INTO THE CLOCK FLAG BIT AND IT STAY


                                        .SBTTL  CLOCK FLAG AFTER INTERRUPT TEST
                                        ;SEE IF AN INTERRUPT CLEARS THE CLOCK FLAG
                                        T0030:  SCOPE
                                                MOV     #TRAP0,@#4      ;SETUP VECOR IN CASE OF UNFORSEEN PROBLEMS
(1)                                             MOV     #340,@#6        ;NO INTERRUPTS WHILE PRINTING FATAL MESSAGE
                                                CLR     LKS             ;NO CLOCK INTERRUPTS BEFORE WE ARE READY
                                                MOV     #100,$GDDAT     ;HAVE GOOD DATA INFO READY FOR TYPEOUT IN CASE OF AN ERR
                                                MOV     #R0030,$LPADR   ;INITIALIZE THE LOOPBACK ADDRESS IN CASE OF AN ERROR
                                        R0030:  MOV     #A0030,100      ;SETUP CLOCK INTERRUPT VECTOR
                                                CLR     102             ;PRIORITY LEVEL WILL ALLOW FURTHER INTERRUPTS
                                                CLR     PS
                                                MOV     #1000,SP        ;SETUP THE STACK
                                                CLR     LKS
                                                CLRB    WORD            ;CLEAR OUT A COUNTER LOCATION
                                                MOV     #100,LKS        ;ENABLE CLOCK INTERRUPTS NOW
                                                WAIT                    ;WAIT FOR AN INTERRUPT
                                        A0030:  MOV     #E0030,100      ;ERROR IF WE INTERRUPT AGAIN
                                                CLR     PS              ;LET INTERRUPTS HAPPEN NOW
                                        B0030:  INCB    WORD            ;DO A NOTHING LOOP FOR A VERY SHORT PERIOD OF TIME
                                                BNE     B0030           ;WE SHOULD INCREMENT TO 0 LONG BEFORE AN INTERRUPT COMES
                                                CLR     LKS
                                                BR      T0031
                                        E0030:  ERROR 1                 ;INTERRUPT DID NOT CLEAR THE CLOCK FLAG


                                        .SBTTL  NO INTERRUT AT PRIORITY 7 TEST
                                        ;TEST THAT CLOCK WILL NOT INTERRUPT WITH PROCESSR AT PRIORITY 7
                                        T0031:  SCOPE
                                                MOV     #TRAP0,@#4      ;SETUP VECOR IN CASE OF UNFORSEEN PROBLEMS
(1)                                             MOV     #340,@#6        ;NO INTERRUPTS WHILE PRINTING FATAL MESSAGE
                                                MOV     #R0031,$LPADR   ;SETUP LOOPBACK ADDRESS IN CASE OF AN ERROR
                                        R0031:  CLR     LKS
                                                CLR     R3              ;INITIALIZE A COUNTER LOCATION
(1)                                     A0031:  TSTB    LKS             ;IS THE CLOCK FLAG SET?
(1)                                             BMI     B0031           ;IF SO, CONTINUE ON WITH THE TEST
```

```
(1)  005170  005203                          INC     R3              ;IF NOT INCREMENT THE COUNTER LOCATION
(1)  005172  001373                          BNE     A0031           ;AND GO TEST THE CLOCK FLAG AGAIN. UNLESS...
(1)  005174  104001                E0031:    ERROR 1                 ;CLOCK FLAG DID NOT SET AFTER A WAITING PERIOD > 20 MS
(1)  005176  000427                          BR      T0032
(1)  005200                        B0031:
601  005200  012706  001000                 MOV     #1000,SP        ;INITIALIZE THE STACK POINTER
602  005204  012737  000340  177776         MOV     #340,PS         ;SET PRIORITY 7
603  005212  012737  005174  000100         MOV     #E0031,100      ;SET UP VECTOR RETURN
604  005220  012737  000100  177546         MOV     #100,LKS        ;ENABLE INTERRUPT
605  005226  005003                          CLR     R3              ;INITIALIZE A COUNTER LOCATION
(1)  005230  105737  177546        C0031:    TSTB    LKS             ;IS THE CLOCK FLAG SET?
(1)  005234  100404                          BMI     D0031           ;IF SO, CONTINUE ON WITH THE TEST
(1)  005236  005203                          INC     R3              ;IF NOT INCREMENT THE COUNTER LOCATION
(1)  005240  001373                          BNE     C0031           ;AND GO TEST THE CLOCK FLAG AGAIN. UNLESS...
(1)  005242  104001                E1O031:   ERROR 1                 ;CLOCK FLAG DID NOT SET AFTER A WAITING PERIOD > 20 MS
(1)  005244  000404                          BR      T0032
(1)  005246                        D0031:
606  005246  000240                          NOP
607  005250  000240                          NOP                     ;GIVE CLOCK EXTRA TIME TO INTERRUPT
608  005252  000401                          BR      T0032
609  005254  104003                E20031:   ERROR 3                 ;ERROR, CLOCK INTERRUPTED WITHOUT HAVING PRIORITY
610
611
612
613                                .SBTTL  CC PUSH TEST FOR CLOCK INTERRUPTS
614                                ;TEST THAT CLOCK INTERRUPT PUSHES CONDITION CODES ONTO STACK
615  005256  000004                T0032:    SCOPE
616  005260  012737  005674  000004         MOV     #TRAP0,@#4      ;SETUP VECOR IN CASE OF UNFORSEEN PROBLEMS
(1)  005266  012737  000340  000006         MOV     #340,@#6        ;NO INTERRUPTS WHILE PRINTING FATAL MESSAGE
617  005274  012737  005302  001106         MOV     #R0032,$LPADR   ;SETUP LOOPBACK ADDRESS IN CASE OF AN ERROR
618  005302  005037  177546        R0032:    CLR     LKS
619  005306  005003                          CLR     R3              ;INITIALIZE A COUNTER LOCATION
(1)  005310  105737  177546        A0032:    TSTB    LKS             ;IS THE CLOCK FLAG SET?
(1)  005314  100404                          BMI     B0032           ;IF SO, CONTINUE ON WITH THE TEST
(1)  005316  005203                          INC     R3              ;IF NOT INCREMENT THE COUNTER LOCATION
(1)  005320  001373                          BNE     A0032           ;AND GO TEST THE CLOCK FLAG AGAIN. UNLESS...
(1)  005322  104001                E0032:    ERROR 1                 ;CLOCK FLAG DID NOT SET AFTER A WAITING PERIOD > 20 MS
(1)  005324  000432                          BR      T0033
(1)  005326                        B0032:
620  005326  012706  001000                 MOV     #1000,SP        ;INITIALIZE THE STACK POINTER
621  005332  005037  000776                 CLR     BUF1
622  005336  005037  000774                 CLR     BUF2
623  005342  012737  005370  000100         MOV     #C0032,100      ;SET UP VECTOR RETURN
624  005350  012737  000100  177546         MOV     #100,LKS        ;ENABLE INTERRUPT
625  005356  012737  000200  177776         MOV     #200,PS         ;SET PRIORITY 4
626  005364  000277                          +SEC!SEV!SEZ!SEN        ;SET ALL CONDITION CODES
627  005366  000001                          WAIT                    ;WAIT FOR INTERRUPT
628  005370  022737  000217  000776 C0032:   CMP     #217,BUF1
629  005376  001405                          BEQ     T0033
630  005400  012737  000017  001124         MOV     #17,$GDDAT
631  005406  104007                          ERROR 7                 ;ERROR DID NOT PUSH CORRECT PS ONTO STACK
632  005410  000722                          BR      T0032
633
634
```

```
                                        .SBTTL  PC PUSH TEST FOR CLOCK INTERRUPTS
                                        ;TEST THAT CLOCK INTERRUPT PUSHES THE PROGRAM COUNTER ONTO STACK
005412  000004                  T0033:  SCOPE
005414  012737  005674  000004          MOV     #TRAP0,@#4      ;SETUP VECOR IN CASE OF UNFORSEEN PROBLEMS
(1) 005422  012737  000340  000006      MOV     #340,@#6        ;NO INTERRUPTS WHILE PRINTING FATAL MESSAGE
005430  012737  005436  001106          MOV     #R0033,$LPADR   ;SETUP LOOPBACK ADDRESS IN CASE OF AN ERROR
005436  005037  177546          R0033:  CLR     LKS
005442  005003                          CLR     R3              ;INITIALIZE A COUNTER LOCATION
(1) 005444  105737  177546          A0033:  TSTB    LKS             ;IS THE CLOCK FLAG SET?
(1) 005450  100404                          BMI     B0033           ;IF SO, CONTINUE ON WITH THE TEST
(1) 005452  005203                          INC     R3              ;IF NOT INCREMENT THE COUNTER LOCATION
(1) 005454  001373                          BNE     A0033           ;AND GO TEST THE CLOCK FLAG AGAIN. UNLESS...
(1) 005456  104001                  E0033:  ERROR 1                 ;CLOCK FLAG DID NOT SET AFTER A WAITING PERIOD > 20 MS
(1) 005460  000431                          BR      T0034
(1) 005462                          B0033:
005462  012706  001000                      MOV     #1000,SP        ;INITIALIZE THE STACK POINTER
005466  005037  000776                      CLR     BUF1
005472  005037  000774                      CLR     BUF2
005476  012737  005524  000100              MOV     #C0033,100      ;SET UP VECTOR RETURN
005504  012737  000100  177546              MOV     #100,LKS        ;ENABLE INTERRUPT
005512  012737  000200  177776              MOV     #200,PS         ;SET PRIORITY 4
005520  000277                              +SEC!SEV!SEZ!SEN        ;SET ALL CONDITION CODES
005522  000001                              WAIT                    ;WAIT FOR INTERRUPT
005524  022737  005524  000774  C0033:      CMP     #C0033,BUF2
005532  001404                              BEQ     T0034
005534  012737  005524  001124  E10033:     MOV     #C0033,$GDDAT
005542  104010                              ERROR 10                ;ERROR, DID NOT PUSH CORRECT PC ONTO STACK


                                        .SBTTL  END OF PASS INDICATING
005544  000004                  T0034:  SCOPE
005546  005037  177546                  CLR     LKS             ;TURN THE CLOCK OFF
005552  000005                          RESET                   ;TURN EVERYTHING OFF

                                        ;;**************************************************************
(1)
(1)                                     .SBTTL  END OF PASS ROUTINE
(1)
(1)                                     ;*INCREMENT THE PASS NUMBER ($PASS)
(1)                                     ;*TYPE "END PASS #XXXXX" (WHERE XXXXX IS A DECIMAL NUMBER)
(1)                                     ;*IF THERES A MONITOR GO TO IT
(1)                                     ;*IF THERE ISN'T JUMP TO T0001
(1)
(1) 005554                          $EOP:
(1) 005554  000004                          SCOPE
(1) 005556  005037  001102                  CLR     $TSTNM          ;ZERO THE TEST NUMBER
(1) 005562  005037  001214                  CLR     $TIMES          ;ZERO THE NUMBER OF ITERATIONS
(1) 005566  005237  001100                  INC     $PASS           ;INCREMENT THE PASS NUMBER
(1) 005572  042737  100000  001100          BIC     #100000,$PASS   ;DON'T ALLOW A NEG. NUMBER
(1) 005600  005327                          DEC     (PC)+           ;LOOP?
(1) 005602  000001                  $EOPCT: .WORD   1
(1) 005604  003021                          BGT     $DOAGN          ;YES
```

```
(1)  005606  012737                          MOV     (PC)+,@(PC)+            ;RESTORE COUNTER
(1)  005610  000001                SENDCT:   .WORD   1
(1)  005612  005602                          $EOPCT
(1)  005614  104400  005654                  TYPE    ,$ENDMG                 ;TYPE "END PASS #"
(2)  005620  013746  001100                  MOV     $PASS,-(SP)             ;SAVE $PASS FOR TYPEOUT
(2)  005624  104410                          TYPOS                           ;GO TYPE--DECIMAL ASCII WITH SIGN
(1)  005626  104400  005671                  TYPE    ,$ENULL                 ;TYPE A NULL CHARACTER
(1)  005632  013700  000042        $GET42:   MOV     @#42,R0                 ;GET MONITOR ADDRESS
(1)  005636  001404                          BEQ     $DOAGN                  ;IF NONE
(1)  005640  004710                $ENDAD:   JSR     PC,(R0)                 ;GO TO MONITOR
(1)  005642  000240                          NOP                             ;SAVE ROOM
(1)  005644  000240                          NOP                             ;FOR
(1)  005646  000240                          NOP                             ;ACT11
(1)  005650  000137  001550        $DOAGN:   JMP     @#T0001                 ;RETURN
(1)  005654  005015  047105  020104 $ENDMG:   .ASCIZ  <15><12>/END PASS #/
(1)  005662  040520  051523  021440
(1)  005670     000
(1)  005671     377     377     000 $ENULL:   .BYTE   -1,-1,0                 ;NULL CHARACTER STRING
664  005674  005046                TRAP0:    CLR     -(SP)
665  005676  004737  006216                  JSR     PC,$TYPE        ;PRINTOUT"TRAPPED TO 4 FROM "
666  005702  007661                          TRPMES                  ;ADDRESS OF THE MESSAGE
667  005704  011646                          MOV     (SP),-(SP)      ;GET THE ADDRESS WHERE THE TRAP OCCURED
668  005706  162716  000002                  SUB     #2,(SP)         ;MAKE IT RIGHT
669  005712  104402                          TYPOC                   ;TYPE OUT ADDRESS IN OCTAL
670  005714  104400  001225                  TYPE    ,$CRLF          ;PRINTOUT A CARRIAGE RETURN-LINE FEED
671  005720  005046                          CLR     -(SP)
672  005722  004737  006216                  JSR     PC,$TYPE        ;PRINTOUT RESTARTING MESSAGE
673  005726  007722                          TRPM2S                  ;ADDRESS OF RESTART MESSAGE
674  005730  000240                          NOP
675  005732  000240                          NOP
676  005734  000240                          NOP
677  005736  000240                          NOP
678  005740  000005                          RESET
679  005742  000137  001416                  JMP     START
680                                ;;*****************************************************************
(1)                                
(1)                                .SBTTL  SCOPE HANDLER ROUTINE
(1)                                
(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)                                ;*THE TEST NUMBER ($TSTNM) IS INCREMENTED AND DISPLAYED IN DISPLAY<7:0>
(1)                                ;*AND THE ERROR FLAG ($ERFLG) IS DISPLAYED IN DISPLAY<15:08>
(1)                                
(1)  005746                        $SCOPE:
(2)  005746  000240                          NOP
(1)  005750  006137  177570                  ROL     @#SWR                   ;LOOP ON PRESENT TEST?
(1)  005754  100511                          BMI     $OVER                   ;YES IF SW14=1
(1)                                          ;#####START OF CODE FOR THE XOR TESTER#####
(1)  005756  000416                $XTSTR:   BR      6$                      ;IF RUNNING ON THE "XOR" TESTER CHANGE
(1)                                                                          ;THIS INSTRUCTION TO A "NOP" (NOP=240)
(1)  005760  013746  000004                  MOV     @#ERRVEC,-(SP)          ;SAVE THE CONTENTS OF THE ERROR VECTOR
(1)  005764  012737  006004  000004          MOV     #5$,@#ERRVEC            ;SET FOR TIMEOUT
```

```
(1)  005772  005737  177060            TST     @#177060                ;TIME OUT ON XOR?
(1)  005776  012637  000004            MOV     (SP)+,@#ERRVEC          ;RESTORE THE ERROR VECTOR
(1)  006002  000463                    BR      $SVLAD                  ;GO TO THE NEXT TEST
(1)  006004  022626                5$: CMP     (SP)+,(SP)+             ;CLEAR THE STACK AFTER A TIME OUT
(1)  006006  012637  000004            MOV     (SP)+,@#ERRVEC          ;RESTORE THE ERROR VECTOR
(1)  006012  000423                    BR      7$                      ;LOOP ON THE PRESENT TEST
(1)  006014                        6$: ;#####END OF CODE FOR THE XOR TESTER#####
(1)  006014  032737  000400  177570    BIT     #SW08,@#SWR             ;LOOP ON SPEC. TEST?
(1)  006022  001404                    BEQ     2$                      ;BR IF NO
(1)  006024  123737  177570  001102    CMPB    @#SWR,$TSTNM            ;ON THE RIGHT TEST?     SWR<7:0>
(1)  006032  001462                    BEQ     $OVER                   ;BR IF YES
(1)  006034  105737  001103        2$: TSTB    $ERFLG                  ;HAS AN ERROR OCCURRED?
(1)  006040  001421                    BEQ     3$                      ;BR IF NO
(1)  006042  123737  001115  001103    CMPB    $ERMAX,$ERFLG           ;MAX. ERRORS FOR THIS TEST OCCURRED?
(1)  006050  101015                    BHI     3$                      ;BR IF NO
(1)  006052  032737  001000  177570    BIT     #SW09,@#SWR             ;LOOP ON ERROR?
(1)  006060  001404                    BEQ     4$                      ;BR IF NO
(1)  006062  013737  001110  001106 7$: MOV     $LPERR,$LPADR           ;SET LOOP ADDRESS TO LAST SCOPE
(1)  006070  000443                    BR      $OVER
(1)  006072  105037  001103        4$: CLRB    $ERFLG                  ;ZERO THE ERROR FLAG
(1)  006076  005037  001214            CLR     $TIMES                  ;CLEAR THE NUMBER OF ITERATIONS TO MAKE
(1)  006102  000415                    BR      1$                      ;ESCAPE TO THE NEXT TEST
(1)  006104  032737  004000  177570 3$: BIT     #SW11,@#SWR             ;INHIBIT ITERATIONS?
(1)  006112  001011                    BNE     1$                      ;BR IF YES
(1)  006114  005737  001100            TST     $PASS                   ;IF FIRST PASS OF PROGRAM
(1)  006120  001406                    BEQ     1$                      ;       INHIBIT ITERATIONS
(1)  006122  005237  001104            INC     $ICNT                   ;INCREMENT ITERATION COUNT
(1)  006126  023737  001214  001104    CMP     $TIMES,$ICNT            ;CHECK THE NUMBER OF ITERATIONS MADE
(1)  006134  002021                    BGE     $OVER                   ;BR IF MORE ITERATION REQUIRED
(1)  006136  012737  000001  001104 1$: MOV     #1,$ICNT                ;REINITIALIZE THE ITERATION COUNTER
(1)  006144  013737  006214  001214    MOV     $MXCNT,$TIMES           ;SET NUMBER OF ITERATIONS TO DO
(1)  006152  105237  001102        $SVLAD: INCB    $TSTNM              ;COUNT TEST NUMBERS
(1)  006156  011637  001106            MOV     (SP),$LPADR             ;SAVE SCOPE LOOP ADDRESS
(1)  006162  011637  001110            MOV     (SP),$LPERR             ;SAVE ERROR LOOP ADDRESS
(1)  006166  005037  001216            CLR     $ESCAPE                 ;CLEAR THE ESCAPE FROM ERROR ADDRESS
(1)  006172  112737  000001  001115    MOVB    #1,$ERMAX               ;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
(1)  006200  013737  001102  177570 $OVER: MOV     $TSTNM,@#DISPLAY        ;DISPLAY TEST NUMBER
(1)  006206  013716  001106            MOV     $LPADR,(SP)             ;FUDGE RETURN ADDRESS
(1)  006212  000002                    RTI                             ;FIXES PS
(1)  006214  000010                $MXCNT: 10                          ;MAX. NUMBER OF ITERATIONS
681                                ;;**************************************************************
(1)
(1)                                .SBTTL  TYPE ROUTINE
(1)
(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)                                     ;*2) USING A JSR INSTRUCTION
(1)                                     ;*      MOV     PS,-(SP)        ;PUSH PROCESSOR STATUS WORD ON THE STACK
(1)                                     ;*      JSR     PC,$TYPE        ;CALL TYPE ROUTINE
(1)                                     ;*      MESADDR                 ;FIRST ADRESS OF MESSAGE
(1)
(1)  006216  105737  001151             $TYPE:  TSTB    $TPFLG          ;IS THERE A TERMINAL?
(1)  006222  100002                             BPL     1$              ;BR IF YES
(1)  006224  000000                             HALT                    ;HALT HERE IF NO TERMINAL
(1)  006226  000407                             BR      3$              ;LEAVE
(1)  006230  010046                     1$:     MOV     R0,-(SP)        ;SAVE R0
(1)  006232  017600  000002                     MOV     @2(SP),R0       ;GET ADDRESS OF ASCIZ STRING
(1)  006236  112046                     2$:     MOVB    (R0)+,-(SP)     ;PUSH CHARACTER TO BE TYPED ONTO STACK
(1)  006240  001005                             BNE     4$              ;BR IF IT ISN'T THE TERMINATOR
(1)  006242  005726                             TST     (SP)+           ;IF TERMINATOR POP IT OFF THE STACK
(1)  006244  012600                             MOV     (SP)+,R0        ;RESTORE R0
(1)  006246  062716  000002             3$:     ADD     #2,(SP)         ;ADJUST RETURN PC
(1)  006252  000002                             RTI                     ;RETURN
(1)  006254  004737  006306             4$:     JSR     PC,7$           ;GO TYPE THIS CHARACTER
(1)  006260  123726  001150             5$:     CMPB    $FILLC,(SP)+    ;IS IT TIME FOR FILLER CHARS.?
(1)  006264  001364                             BNE     2$              ;IF NO GO GET NEXT CHAR.
(1)  006266  013746  001146                     MOV     $NULL,-(SP)     ;GET # OF FILLER CHARS. NEEDED
(1)                                                                     ;AND THE NULL CHAR.
(1)  006272  105366  000001             6$:     DECB    1(SP)           ;DOES A NULL NEED TO BE TYPED?
(1)  006276  002770                             BLT     5$              ;BR IF NO--GO POP THE NULL OFF OF STACK
(1)  006300  004737  006306                     JSR     PC,7$           ;GO TYPE A NULL
(1)  006304  000772                             BR      6$              ;LOOP
(1)  006306  105777  172630             7$:     TSTB    @$TPS           ;WAIT UNTIL PRINTER IS READY
(1)  006312  100375                             BPL     7$
(1)  006314  116677  000002  172622             MOVB    2(SP),@$TPB     ;LOAD CHAR TO BE TYPED INTO DATA REG.
(1)  006322  000207                             RTS     PC
682                                     ;;**************************************************************
(1)
(1)                                     .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
(1)
(1)                                     ;#CALL:
(1)                                     ;#      MOV     NUM,-(SP)       ;PUT THE BINARY NUMBER ON THE STACK
(1)                                     ;#      TYPDS                   ;GO TO THE ROUTINE
(1)
(2)  006324                             $TYPDS:
(3)  006324  010046                             MOV     R0,-(SP)        ;PUSH R0 ON STACK
(3)  006326  010146                             MOV     R1,-(SP)        ;PUSH R1 ON STACK
(3)  006330  010246                             MOV     R2,-(SP)        ;PUSH R2 ON STACK
(3)  006332  010346                             MOV     R3,-(SP)        ;PUSH R3 ON STACK
(3)  006334  010546                             MOV     R5,-(SP)        ;PUSH R5 ON STACK
(1)  006336  012746  020200                     MOV     #20200,-(SP)    ;SET BLANK SWITCH AND SIGN
(1)  006342  016605  000020                     MOV     20(SP),R5       ;GET THE INPUT NUMBER
(1)  006346  100004                             BPL     1$              ;BR IF INPUT IS POS.
(1)  006350  005405                             NEG     R5              ;MAKE THE BINARY NUMBER POS.
(1)  006352  112766  000055  000001             MOVB    #'-,1(SP)       ;MAKE THE ASCII NUMBER NEG.
(1)  006360  005000                     1$:     CLR     R0              ;ZERO THE CONSTANTS INDEX
(1)  006362  012703  006540                     MOV     #$DBLK,R3       ;SETUP THE OUTPUT POINTER
```

```
(1)  006366  112723  000040                    MOVB    #' ,(R3)+       ;SET THE FIRST CHARACTER TO A BLANK
(1)  006372  005002                      2$:   CLR     R2              ;CLEAR THE BCD NUMBER
(1)  006374  016001  006530                    MOV     $DTBL(R0),R1    ;GET THE CONSTANT
(1)  006400  160105                      3$:   SUB     R1,R5           ;FORM THIS BCD DIGIT
(1)  006402  002402                            BLT     4$              ;BR IF DONE
(1)  006404  005202                            INC     R2              ;INCREASE THE BCD DIGIT BY 1
(1)  006406  000774                            BR      3$
(1)  006410  060105                      4$:   ADD     R1,R5           ;ADD BACK THE CONSTANT
(1)  006412  005702                            TST     R2              ;CHECK IF BCD DIGIT=0
(1)  006414  001002                            BNE     5$              ;FALL THROUGH IF 0
(1)  006416  105716                            TSTB    (SP)            ;STILL DOING LEADING 0'S?
(1)  006420  100407                            BMI     7$              ;BR IF YES
(1)  006422  106316                      5$:   ASLB    (SP)            ;MSD?
(1)  006424  103003                            BCC     6$              ;BR IF NO
(1)  006426  116663  000001  177777            MOVB    1(SP),-1(R3)    ;YES--SET THE SIGN
(1)  006434  052702  000060              6$:   BIS     #'0,R2          ;MAKE THE BCD DIGIT ASCII
(1)  006440  052702  000040              7$:   BIS     #' ,R2          ;MAKE IT A SPACE IF NOT ALREADY A DIGIT
(1)  006444  110223                            MOVB    R2,(R3)+        ;PUT THIS CHARACTER IN THE OUTPUT BUFFER
(1)  006446  005720                            TST     (R0)+           ;JUST INCREMENTING
(1)  006450  020027  000010                    CMP     R0,#10          ;CHECK THE TABLE INDEX
(1)  006454  002746                            BLT     2$              ;GO DO THE NEXT DIGIT
(1)  006456  003002                            BGT     8$              ;GO TO EXIT
(1)  006460  010502                            MOV     R5,R2           ;GET THE LSD
(1)  006462  000764                            BR      6$              ;GO CHANGE TO ASCII
(1)  006464  105726                      8$:   TSTB    (SP)+           ;WAS THE LSD THE FIRST NON-ZERO?
(1)  006466  100003                            BPL     9$              ;BR IF NO
(1)  006470  116663  177777  177776            MOVB    -1(SP),-2(R3)   ;YES--SET THE SIGN FOR TYPING
(1)  006476  105013                      9$:   CLRB    (R3)            ;SET THE TERMINATOR
(3)  006500  012605                            MOV     (SP)+,R5        ;POP STACK INTO R5
(3)  006502  012603                            MOV     (SP)+,R3        ;POP STACK INTO R3
(3)  006504  012602                            MOV     (SP)+,R2        ;POP STACK INTO R2
(3)  006506  012601                            MOV     (SP)+,R1        ;POP STACK INTO R1
(3)  006510  012600                            MOV     (SP)+,R0        ;POP STACK INTO R0
(1)  006512  104400  006540                    TYPE    ,$DBLK          ;NOW TYPE THE NUMBER
(1)  006516  016666  000002  000004            MOV     2(SP),4(SP)     ;ADJUST THE STACK
(1)  006524  012616                            MOV     (SP)+,(SP)
(1)  006526  000002                            RTI                     ;RETURN TO USER
(1)  006530  023420                      $DTBL:  10000.
(1)  006532  001750                              1000.
(1)  006534  000144                              100.
(1)  006536  000012                              10.
(1)  006540  000004                      $DBLK:  .BLKW   4
683                                      ;;*****************************************************************
(1)
(1)                                      .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
(1)
(1)                                      ;*$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)                                     ;*STYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
(1)                                     ;*STYPOS OR STYPOC
(1)                                     ;*CALL:
(1)                                     ;*      MOV     NUM,-(SP)               ;NUMBER TO BE TYPED
(1)                                     ;*      TYPON                           ;CALL FOR TYPEOUT
(1)                                     ;*
(1)                                     ;*STYPOC---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)  006550  017646  000000          STYPOS: MOV     @(SP),-(SP)             ;PICKUP THE MODE
(1)  006554  116637  000001  006773          MOVB    1(SP),$OFILL            ;LOAD ZERO FILL SWITCH
(1)  006562  112637  006775                   MOVB    (SP)+,$OMODE+1          ;NUMBER OF DIGITS TO TYPE
(1)  006566  062716  000002                   ADD     #2,(SP)                 ;ADJUST RETURN ADDRESS
(1)  006572  000406                           BR      $TYPON
(1)  006574  112737  000001  006773  STYPOC: MOVB    #1,$OFILL               ;SET THE ZERO FILL SWITCH
(1)  006602  112737  000006  006775          MOVB    #6,$OMODE+1             ;SET FOR SIX(6) DIGITS
(1)  006610  112737  000005  006772  STYPON: MOVB    #5,$OCNT                ;SET THE ITERATION COUNT
(1)  006616  010346                           MOV     R3,-(SP)                ;SAVE R3
(1)  006620  010446                           MOV     R4,-(SP)                ;SAVE R4
(1)  006622  010546                           MOV     R5,-(SP)                ;SAVE R5
(1)  006624  113704  006775                   MOVB    $OMODE+1,R4             ;GET THE NUMBER OF DIGITS TO TYPE
(1)  006630  005404                           NEG     R4
(1)  006632  062704  000006                   ADD     #6,R4                   ;SUBTRACT IT FOR MAX. ALLOWED
(1)  006636  110437  006774                   MOVB    R4,$OMODE               ;SAVE IT FOR USE
(1)  006642  113704  006773                   MOVB    $OFILL,R4               ;GET THE ZERO FILL SWITCH
(1)  006646  016605  000012                   MOV     12(SP),R5               ;PICKUP THE INPUT NUMBER
(1)  006652  005003                           CLR     R3                      ;CLEAR THE OUTPUT WORD
(1)  006654  006105                    1$:    ROL     R5                      ;ROTATE MSB INTO "C"
(1)  006656  000404                           BR      3$                      ;GO DO MSB
(1)  006660  006105                    2$:    ROL     R5                      ;FORM THIS DIGIT
(1)  006662  006105                           ROL     R5
(1)  006664  006105                           ROL     R5
(1)  006666  010503                           MOV     R5,R3
(1)  006670  006103                    3$:    ROL     R3                      ;GET LSB OF THIS DIGIT
(1)  006672  105337  006774                   DECB    $OMODE                  ;TYPE THIS DIGIT?
(1)  006676  100016                           BPL     7$                      ;BR IF NO
(1)  006700  042703  177770                   BIC     #177770,R3              ;GET RID OF JUNK
(1)  006704  001002                           BNE     4$                      ;TEST FOR 0
(1)  006706  005704                           TST     R4                      ;SUPPRESS THIS 0?
(1)  006710  001403                           BEQ     5$                      ;BR IF YES
(1)  006712  005204                    4$:    INC     R4                      ;DON'T SUPPRESS ANYMORE 0'S
(1)  006714  052703  000060                   BIS     #'0,R3                  ;MAKE THIS DIGIT ASCII
(1)  006720  052703  000040             5$:   BIS     #' ,R3                  ;MAKE ASCII IF NOT ALREADY
(1)  006724  110337  006770                   MOVB    R3,8$                   ;SAVE FOR TYPING
(1)  006730  104400  006770                   TYPE    ,8$                     ;GO TYPE THIS DIGIT
(1)  006734  105337  006772             7$:   DECB    $OCNT                   ;COUNT BY 1
(1)  006740  003347                           BGT     2$                      ;BR IF MORE TO DO
(1)  006742  002402                           BLT     6$                      ;BR IF DONE
(1)  006744  005204                           INC     R4                      ;INSURE LAST DIGIT ISN'T A BLANK
(1)  006746  000744                           BR      2$                      ;GO DO THE LAST DIGIT
(1)  006750  012605                    6$:    MOV     (SP)+,R5                ;RESTORE R5
```

```
(1)  006752  012604                        MOV     (SP)+,R4        ;RESTORE R4
(1)  006754  012603                        MOV     (SP)+,R3        ;RESTORE R3
(1)  006756  016666  000002  000004        MOV     2(SP),4(SP)     ;SET THE STACK FOR RETURNING
(1)  006764  012616                        MOV     (SP)+,(SP)
(1)  006766  000002                        RTI                     ;RETURN
(1)  006770     000                 BS:     .BYTE   0               ;STORAGE FOR ASCII DIGIT
(1)  006771     000                         .BYTE   0               ;TERMINATOR FOR TYPE ROUTINE
(1)  006772     000                 $OCNT:  .BYTE   0               ;OCTAL DIGIT COUNTER
(1)  006773     000                 $OFILL: .BYTE   0               ;ZERO FILL SWITCH
(1)  006774  000000                 $OMODE: 0                       ;NUMBER OF DIGITS TO TYPE
684                                 ;;*****************************************************************
(1)
(1)                                 .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE
(1)
(1)                                 ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($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)  006776                         $ERRTYP:
(1)  006776  104400  001225                 TYPE    ,$CRLF          ;"CARRIAGE RETURN" & "LINE FEED"
(1)  007002  010046                         MOV     R0,-(SP)        ;SAVE R0
(1)  007004  005000                         CLR     R0              ;PICKUP THE ITEM INDEX
(1)  007006  153700  001114                 BISB    @#$ITEMB,R0
(1)  007012  001004                         BNE     1$              ;IF ITEM NUMBER IS ZERO, JUST
(1)                                                                 ;TYPE THE PC OF THE ERROR
(2)  007014  013746  001116                 MOV     $ERRPC,-(SP)    ;SAVE $ERRPC FOR TYPEOUT
(2)                                                                 ;ERROR ADDRESS
(2)  007020  104402                         TYPOC                   ;GO TYPE--OCTAL ASCII(ALL DIGITS)
(1)  007022  000426                         BR      6$              ;GET OUT
(1)  007024  005300                 1$:     DEC     R0              ;ADJUST THE INDEX SO THAT IT WILL
(1)  007026  006300                         ASL     R0              ;       WORK FOR THE ERROR TABLE
(1)  007030  006300                         ASL     R0
(1)  007032  006300                         ASL     R0
(1)  007034  062700  001232                 ADD     #$ERRTB,R0      ;FORM TABLE POINTER
(1)  007040  012037  007050                 MOV     (R0)+,2$        ;PICKUP "ERROR MESSAGE" POINTER
(1)  007044  001404                         BEQ     3$              ;SKIP TYPEOUT IF NO POINTER
(1)  007046  104400                         TYPE                    ;TYPE THE "ERROR MESSAGE"
(1)  007050  000000                 2$:     .WORD   0               ;"ERROR MESSAGE" POINTER GOES HERE
(1)  007052  104400  001225                 TYPE    ,$CRLF          ;"CARRIAGE RETURN" & "LINE FEED"
(1)  007056  012037  007066         3$:     MOV     @(R0)+,4$       ;PICKUP "DATA HEADER" POINTER
(1)  007062  001404                         BEQ     5$              ;SKIP TYPEOUT IF 0
(1)  007064  104400                         TYPE                    ;TYPE THE "DATA HEADER"
(1)  007066  000000                 4$:     .WORD   0               ;"DATA HEADER" POINTER GOES HERE
(1)  007070  104400  001225                 TYPE    ,$CRLF          ;"CARRIAGE RETURN" & "LINE FEED"
(1)  007074  011000                 5$:     MOV     @(R0),R0        ;PICKUP "DATA TABLE" POINTER
(1)  007076  001004                         BNE     7$              ;GO TYPE THE DATA
(1)  007100  012600                 6$:     MOV     (SP)+,R0        ;RESTORE R0
(1)  007102  104400  001225                 TYPE    ,$CRLF          ;"CARRIAGE RETURN" & "LINE FEED"
(1)  007106  000207                         RTS     PC              ;RETURN
(2)  007110                         7$:
(2)  007110  013046                         MOV     @(R0)+,-(SP)    ;SAVE @(R0)+ FOR TYPEOUT
(2)  007112  104402                         TYPOC                   ;GO TYPE--OCTAL ASCII(ALL DIGITS)
(1)  007114  005710                         TST     (R0)            ;IS THERE ANOTHER NUMBER?
(1)  007116  001770                         BEQ     6$              ;BR IF NO
```

```
(1)  007120  104400  007126                TYPE    ,8$                     ;TYPE TWO(2) SPACES
(1)  007124  000771                        BR      7$                      ;LOOP
(1)  007126  020040     000        8$:     .ASCIZ  /  /                    ;TWO(2) SPACES
(1)          007132                        .EVEN
693                                 ;;*****************************************************************
(1)
(1)                                 .SBTTL  ERROR HANDLER ROUTINE
(1)
(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)                                 ;*GO TO SERRTYP ON ERROR
(1)
(1)  007132                         $ERROR:
(3)  007132  010637  001170                MOV     SP,$REG6        ;GET THE CURRENT STACK POINTER VALUE
(3)  007136  162737  000004  001170        SUB     #4,$REG6                ;RESTORE IT TO ITS "PRE ERROR TRAP" VALUE FOR PR
(3)  007144  016637  000002  001172        MOV     2(SP),$REG7             ;GET THE PS OFF OF THE STACK
(3)  007152  005037  001166                CLR     $REG5                   ;PREPARE "$REG5" TO HOLD THE TEST #
(3)  007156  113737  001102  001166        MOVB    $TSTNM,$REG5            ;TEST # IS HELD IN THE LOW BYTE OF "TSTNM"
(3)  007164  010037  001154                MOV     R0,$REG0                ;MOST OF THE TIME R0 HAS GOOD STUFF IN IT ALSO
(1)  007170  105237  001103        7$:     INCB    $ERFLG                  ;SET THE ERROR FLAG
(1)  007174  001775                        BEQ     7$                      ;DON'T LET THE FLAG GO TO ZERO
(1)  007176  013737  001102  177570        MOV     $TSTNM,@#DISPLAY        ;DISPLAY TEST NUMBER AND ERROR FLAG
(1)  007204  032737  002000  177570        BIT     #SW10,@#SWR             ;BELL ON ERROR?
(1)  007212  001402                        BEQ     1$                      ;NO - SKIP
(1)  007214  104400  001220                TYPE    ,$BELL                  ;RING BELL
(1)  007220  005237  001112        1$:     INC     $ERTTL                  ;COUNT THE NUMBER OF ERRORS
(1)  007224  011637  001116                MOV     (SP),$ERRPC             ;GET ADDRESS OF ERROR INSTRUCTION
(1)  007230  162737  000002  001116        SUB     #2,$ERRPC
(1)  007236  117737  171654  001114        MOVB    @$ERRPC,$ITEMB          ;STRIP AND SAVE THE ERROR ITEM CODE
(1)  007244  032737  020000  177570        BIT     #SW13,@#SWR             ;SKIP TYPEOUT IF SET
(1)  007252  001004                        BNE     2$                      ;SKIP TYPEOUTS
(1)  007254  004737  006776                JSR     PC,@#$ERRTYP            ;GO TO USER ERROR ROUTINE
(1)  007260  104400  001225                TYPE    ,$CRLF
(1)  007264  005737  177570        2$:     TST     @#SWR                   ;HALT ON ERROR
(1)  007270  100001                        BPL     3$                      ;SKIP IF CONTINUE
(1)  007272  000000                        HALT                            ;HALT ON ERROR!
(1)  007274  032737  001000  177570  3$:   BIT     #SW09,@#SWR             ;LOOP ON ERROR SWITCH SET?
(1)  007302  001402                        BEQ     4$                      ;BR IF NO
(1)  007304  013716  001110                MOV     $LPERR,(SP)             ;FUDGE RETURN FOR LOOPING
(1)  007310  005737  001216        4$:     TST     $ESCAPE                 ;CHECK FOR AN ESCAPE ADDRESS
(1)  007314  001402                        BEQ     5$                      ;BR IF NONE
(1)  007316  013716  001216                MOV     $ESCAPE,(SP)            ;FUDGE RETURN ADDRESS FOR ESCAPE
(1)  007322  000002                 5$:    RTI                             ;RETURN
694                                 ;;*****************************************************************
(1)
(1)                                 .SBTTL  TRAP DECODER
(1)
(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)  007324  010046                $TRAP:  MOV     R0,-(SP)            ;SAVE R0
(1)  007326  016600  000002                MOV     2(SP),R0            ;GET TRAP ADDRESS
(1)  007332  005740                        TST     -(R0)               ;BACKUP BY 2
(1)  007334  111000                        MOVB    (R0),R0             ;GET RIGHT BYTE OF TRAP
(1)  007336  016000  007344                MOV     $TRPAD(R0),R0       ;INDEX TO TABLE
(1)  007342  000200                        RTS     R0                  ;GO TO ROUTINE
(1)
(3)
(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)  007344                         $TRPAD:
(3)  007344  006216                        $TYPE           ;CALL=TYPE      TRAP+0(104400)  TTY TYPEOUT ROUTINE
(3)  007346  006574                        $TYPOC          ;CALL=TYPOC     TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING
(3)  007350  006550                        $TYPOS          ;CALL=TYPOS     TRAP+4(104404)  TYPE OCTAL NUMBER (NO LEADING ZE
(3)  007352  006610                        $TYPON          ;CALL=TYPON     TRAP+6(104406)  TYPE OCTAL NUMBER (AS PER LAST C
(3)  007354  006324                        $TYPDS          ;CALL=TYPDS     TRAP+10(104410) TYPE DECIMAL NUMBER (WITH SIGN)
 695                                ;;**************************************************************
(1)
(1)                                 .SBTTL  POWER DOWN AND UP ROUTINES
(1)
(1)                                 ;POWER DOWN ROUTINE
(1)  007356  012737  007520  000024 $PWRDN: MOV     #$ILLUP,@#PWRVEC    ;SET FOR FAST UP
(1)  007364  012737  000340  000026        MOV     #340,@#PWRVEC+2     ;PRIO:7
(3)  007372  010046                        MOV     R0,-(SP)            ;PUSH R0 ON STACK
(3)  007374  010146                        MOV     R1,-(SP)            ;PUSH R1 ON STACK
(3)  007376  010246                        MOV     R2,-(SP)            ;PUSH R2 ON STACK
(3)  007400  010346                        MOV     R3,-(SP)            ;PUSH R3 ON STACK
(3)  007402  010446                        MOV     R4,-(SP)            ;PUSH R4 ON STACK
(3)  007404  010546                        MOV     R5,-(SP)            ;PUSH R5 ON STACK
(3)  007406  013746  007746                MOV     POWPUS,-(SP)        ;PUSH POWPUS ON STACK
(3)  007412  013746  007750                MOV     POWPOP,-(SP)        ;PUSH POWPOP ON STACK
(3)  007416  013746  007536                MOV     POWMES,-(SP)        ;PUSH POWMES ON STACK
(3)  007422  013746  001550                MOV     T0001,-(SP)         ;PUSH T0001 ON STACK
(1)  007426  010637  007524                MOV     SP,$SAVR6           ;SAVE SP
(1)  007432  012737  007444  000024        MOV     #$PWRUP,@#PWRVEC    ;SET UP VECTOR
(1)  007440  000000                        HALT
(1)  007442  000776                        BR      .-2                 ;HANG UP
(1)
(1)                                 ;POWER UP ROUTINE
(1)  007444  013706  007524         $PWRUP: MOV     $SAVR6,SP           ;GET SP
(1)  007450  005037  007524                 CLR     $SAVR6              ;WAIT LOOP FOR THE TTY
(1)  007454  005237  007524         1$:     INC     $SAVR6              ;WAIT FOR THE INC
(1)  007460  001375                        BNE     1$                  ;OF <POWPUS>,<POWPOP>,<POWMES>,<T0001> WORD
(3)  007462  012605                        MOV     (SP)+,R5            ;POP STACK INTO R5
(3)  007464  012604                        MOV     (SP)+,R4            ;POP STACK INTO R4
(3)  007466  012603                        MOV     (SP)+,R3            ;POP STACK INTO R3
(3)  007470  012602                        MOV     (SP)+,R2            ;POP STACK INTO R2
(3)  007472  012601                        MOV     (SP)+,R1            ;POP STACK INTO R1
(3)  007474  012600                        MOV     (SP)+,R0            ;POP STACK INTO R0
```

```
(1)  007476  012737  007356  000024          MOV     #$PWRDN,@#PWRVEC        ;SET UP THE POWER DOWN VECTOR
(1)  007504  012737  000340  000026          MOV     #340,@#PWRVEC+2         ;PRIO:7
(1)  007512  104400  007526                  TYPE    ,$POWER                 ;POWER FAIL MESSAGE
(1)  007516  000002                          RTI
(1)  007520  000000                $ILLUP:   HALT                            ;THE POWER UP SEQUENCE WAS STARTED
(1)  007522  000776                          BR      .-2                     ; BEFORE THE POWER DOWN WAS COMPLETE
(1)  007524  000000                $SAVR6:   0                               ;PUT THE SP HERE
(1)  007526  005015  047520  042527  $POWER:  .ASCIZ  <15><12>"POWER"
(1)  007534  000122
(1)                                          .EVEN
696  007536  005015  042522  052123  POWMES:  .ASCIZ <15> <12> "RESTARTING AFTER A POWER FIALURE" <15> <12> <12>
     007544  051101  044524  043516
     007552  040440  052106  051105
     007560  040440  050040  053517
     007566  051105  043040  040511
     007574  052514  042522  005015
     007602  000012
697  007604  005015  046412  026504  STMES:   .ASCIZ  <15><12><12>"MD-11-DZKWA-D LINE FREQUENCY CLOCK TEST"<15><12>
     007612  030461  042055  045532
     007620  040527  042055  046040
     007626  047111  020105  051106
     007634  050505  042525  041516
     007642  020131  046103  041517
     007650  020113  042524  052123
     007656  005015     000
698
699  007661     124  040522  050120  TRPMES:  .ASCIZ  "TRAPPED TO LOC 4 FROM LOCATION  "
     007666  042105  052040  020117
     007674  047514  020103  020064
     007702  051106  046517  046040
     007710  041517  052101  047511
     007716  020116  000040
700  007722  042522  052123  051101  TRPM2S:  .ASCIZ  "RESTARTING PROGRAM"
     007730  044524  043516  050040
     007736  047522  051107  046501
     007744     000
701          007746                          .EVEN
702  007746  001230                  POWPUS:  WORD
703  007750  001230                  POWPOP:  WORD
704  007752  020040  020040  020040  EM1:     .ASCIZ  "                                LKS     LKS     "
     007760  020040  020040  020040
     007766  020040  020040  020040
     007774  020040  020040  020040
     010002  020040  020040  020040
     010010  020040  045514  020123
     010016  020040  020040  045514
     010024  020123  020040  020040
     010032     000
705  010033     050  041520  020051  DH1:     .ASCIZ  "(PC)    (PS)    (SP)    TEST#   WAS     S/B     "
     010040  020040  024040  051520
     010046  020051  020040  024040
     010054  050123  020051  020040
     010062  052040  051505  021524
     010070  020040  053440  051501
```

```
        010076  020040  020040  051440
        010104  041057  020040  020040
        010112  000040
  706                                   .EVEN
  707   010114  001116  001172  001170  DT1:    $ERRPC,$REG7,$REG6,$REG5,LKS,$GDDAT
        010122  001166  177546  001124
  708   010130  000000                  0
  709   010132  046103  041517  020113  EM2:    .ASCIZ  "CLOCK FAILED TO INTERRUPT"
        010140  040506  046111  042105
        010146  052040  020117  047111
        010154  042524  051122  050125
        010162  000124
  710   010164  050050  024503  020040  DH2:    .ASCIZ  "(PC)    (PS)    (SP)    TEST#   (LKS)   "
        010172  020040  050050  024523
        010200  020040  020040  051450
        010206  024520  020040  020040
        010214  042524  052123  020043
        010222  020040  046050  051513
        010230  020051  020040     000
  711           010236                  .EVEN
  712   010236  001116  001172  001170  DT2:    $ERRPC,$REG7,$REG6,$REG5,LKS
        010244  001166  177546
  713   010250  000000                  0
  714   010252  046103  041517  020113  EM3:    .ASCIZ  "CLOCK INTERRUPTED WHEN THE PROCESSOR PRIORITY WAS TOO HIGH"
        010260  047111  042524  051122
        010266  050125  042524  020104
        010274  044127  047105  052040
        010302  042510  050040  047522
        010310  042503  051523  051117
        010316  050040  044522  051117
        010324  052111  020131  040527
        010332  020123  047524  020117
        010340  044510  044107     000
  715   010345     050  041520  020051  DH3:    .ASCIZ  "(PC)    (PS)    (SP)    TEST#   (LKS)   "
        010352  020040  024040  051520
        010360  020051  020040  024040
        010366  050123  020051  020040
        010374  052040  051505  021524
        010402  020040  024040  045514
        010410  024523  020040  000040
  716                                   .EVEN
  717   010416  001116  001172  001170  DT3:    $ERRPC,$REG7,$REG6,$REG5,LKS
        010424  001166  177546
  718   010430  000000                  0
  719   010432  046103  041517  020113  EM4:    .ASCIZ  "CLOCK GIVES UNEQUAL # OF PULSES OVER TWO EQUAL PERIODS OF TIME"
        010440  044507  042526  020123
        010446  047125  050505  040525
        010454  020114  020043  043117
        010462  050040  046125  042523
        010470  020123  053117  051105
        010476  052040  047527  042440
        010504  052521  046101  050040
        010512  051105  047511  051504
        010520  047440  020106  044524
```

```
                010526  042515  000
  720           010531  050  041520  020051  DH4:    .ASCIZ  "(PC)    (PS)    (SP)    TEST#   1ST     2ND"<15><12>"
                010536  020040  024040  051520
                010544  020051  020040  024040
                010552  [illegible]  020051  020040
                010560  [illegible]  051505  021524
                010566  020040  030440  052123
                010574  020040  020040  031040
                010602  042116  005015  020040
                010610  020040  020040  020040
                010616  020040  020040  020040
                010624  020040  020040  020040
                010632  020040  020040  020040
                010640  020040  020040  020040
                010646  042520  044522  042117
                010654  020040  042520  044522
                010662  042117  000
  721                   010666                  .EVEN
  722           010666  001116  001172  001170  DT4:    SERRPC,$REG7,$REG6,$REG5,$REG1,$REG2
                010674  001166  001156  001160
  723           010702  000000                  0
  724           010704  045514  020123  042522  EM5:    .ASCIZ  "LKS REGISTER RESPONDS TO ANOTHER ADDRESS"
                010712  044507  052123  051105
                010720  051040  051505  047520
                010726  042116  020123  047524
                010734  040440  047516  044124
                010742  051105  040440  042104
                010750  042522  051523  000
  725           010755  050  041520  020051  DH5:    .ASCIZ  "(PC)    (PS)    (SP)    TEST#   ADDRESS"
                010762  020040  024040  051520
                010770  020051  020040  024040
                010776  050123  020051  020040
                011004  052040  051505  021524
                011012  020040  040440  042104
                011020  042522  051523  000
  726                   011026                  .EVEN
  727           011026  001116  001172  001170  DT5:    SERRPC,$REG7,$REG6,$REG5,$GOADR
                011034  001166  001120
  728           011040  000000                  0
  729           011042  020101  047516  051440  EM6:    .ASCIZ  "A NO SACK TIMEOUT HAS OCCURED"
                011050  041501  020113  044524
                011056  042515  052517  020124
                011064  040510  020123  041517
                011072  052503  042522  000104
  730           011100  050050  024503  020040  DH6:    .ASCIZ  "(PC)    (PS)    (SP)    TEST#   (LKS)   "
                011106  020040  050050  024523
                011114  020040  020040  051450
                011122  024520  020040  020040
                011130  042524  052123  020043
                011136  020040  046050  051513
                011144  020051  020040  000
  731                   011152                  .EVEN
  732           011152  001116  001172  001170  DT6:    SERRPC,$REG7,$REG6,$REG5,LKS
                011160  001166  177546
```

```
   733  011164  000000                    0
   734  011166  051127  047117  020107  EM7:    .ASCIZ  "WRONG CONDITION CODES WERE PUT ONTO STACK BY INTERRUPT"
        011174  047503  042116  052111
        011202  047511  020116  047503
        011210  042504  020123  042527
        011216  042522  050040  052125
        011224  047440  052116  020117
        011232  052123  041501  020113
        011240  054502  044440  052116
        011246  051105  052522  052120
        011254     000
   735  011255     050  041520  020051  DH7:    .ASCIZ  "(PC)    (PS)    (SP)    TEST#   CC WAS  CC S/B"
        011262  020040  024040  051520
        011270  020051  020040  024040
        011276  050123  020051  020040
        011304  052040  051505  021524
        011312  020040  041440  020103
        011320  040527  020123  041440
        011326  020103  027523  000102
   736                             .EVEN
   737  011334  001116  001172  001170  DT7:    $ERRPC,$REG7,$REG6,$REG5,BUF1,$GDDAT
        011342  001166  000776  001124
   738  011350  000000                    0
   739  011352  051127  047117  020107  EM10:   .ASCIZ  "WRONG PC PUT ONTO THE STACK BY AN INTERRUPT"
        011360  041520  050040  052125
        011366  047440  052116  020117
        011374  044124  020105  052123
        011402  041501  020113  054502
        011410  040440  020116  047111
        011416  042524  051122  050125
        011424  000124
   740  011426  050050  024503  020040  DH10:   .ASCIZ  "(PC)    (PS)    (SP)    TEST#  @(SP)WAS @(SP)S/B  "
        011434  020040  050050  024523
        011442  020040  020040  051450
        011450  024520  020040  020040
        011456  042524  052123  020043
        011464  040040  051450  024520
        011472  040527  020123  024100
        011500  050123  051451  041057
        011506  020040     000
   741          011512                  .EVEN
   742  011512  001116  001172  001170  DT10:   $ERRPC,$REG7,$REG6,$REG5,BUF2,$GDDAT
        011520  001166  000774  001124
   743  011526  000000                    0
   744  011530  051124  042531  020104  EM11:   .ASCIZ  "TRYED TO ACCESS THE LKS REGISTER, AND TRAPPED"
        011536  047524  040440  041503
        011544  051505  020123  044124
        011552  020105  045514  020123
        011560  042522  044507  052123
        011566  051105  020054  047101
        011574  020104  051124  050101
        011602  042520  000104
   745  011606  050050  024503  020040  DH11:   .ASCIZ  "(PC)    (PS)    (SP)    TEST#"
        011614  020040  050050  024523
```

```
        011622  020040  020040  051450
        011630  024520  020040  020040
        011636  042524  052123  000043
   746                          .EVEN
   747  011644  001116  001172  001170  DT11:   $ERRPC,$REG7,$REG6,$REG5
        011652  001166
   748  011654  000000                   0
   749          000001                   .END
```

```
A0004   001762      A0005   002044      A0006   002150      A0007   002300
A0010   002430      A0011   002532      A0012   002666      A0013   002752
A0014   003272      A0015   003404      A0016   003516      A0017   003630
A0020   003742      A0021   004054      A0022   004202      A0023   004250
A0024   004366      A0025   004472      A0026   004634      A0027   004744
A0030   005100      A0031   005162      A0032   005310      A0033   005444
BIT0  = 000001      BIT00 = 000001      BIT01 = 000002      BIT02 = 000004
BIT03 = 000010      BIT04 = 000020      BIT05 = 000040      BIT06 = 000100
BIT07 = 000200      BIT08 = 000400      BIT09 = 001000      BIT1  = 000002
BIT10 = 002000      BIT11 = 004000      BIT12 = 010000      BIT13 = 020000
BIT14 = 040000      BIT15 = 100000      BIT2  = 000004      BIT3  = 000010
BIT4  = 000020      BIT5  = 000040      BIT6  = 000100      BIT7  = 000200
BIT8  = 000400      BIT9  = 001000      BPTVEC= 000014      BUF1  = 000776
BUF2  = 000774      B0005   002062      B0006   002166      B0007   002316
B0010   002446      B0011   002550      B0012   002704      B0013   002770
B0014   003304      B0015   003416      B0016   003530      B0017   003642
B0020   003754      B0021   004066      B0023   004266      B0025   004510
B0030   005112      B0031   005200      B0032   005326      B0033   005462
C0007   002326      C0010   002456      C0011   002600      C0013   003004
C0025   004530      C0031   005230      C0032   005370      C0033   005524
DH1     010033      DH10    011426      DH11    011606      DH2     010164
DH3     010345      DH4     010531      DH5     010755      DH6     011100
DH7     011255      DISPLA= 177570      DT1     010114      DT10    011512
DT11    011644      DT2     010236      DT3     010416      DT4     010666
DT5     011026      DT6     011152      DT7     011334      D0007   002340
D0010   002470      D0011   002616      D0013   003022      D0031   005246
EMTVEC= 000030      EM1     007752      EM10    011352      EM11    011530
EM2     010132      EM3     010252      EM4     010432      EM5     010704
EM6     011042      EM7     011166      ERRVEC= 000004      E0001   001606
E0002   001654      E0003   001720      E0004   001774      E0005   002056
E0006   002162      E0007   002312      E0010   002442      E0011   002544
E0012   002700      E0013   002764      E0014   003302      E0015   003414
E0016   003526      E0017   003640      E0020   003752      E0021   004064
E0022   004200      E0023   004262      E0024   004406      E0025   004504
E0026   004654      E0027   004776      E0030   005126      E0031   005174
E0032   005322      E0033   005456      E10006  002210      E10007  002334
E10010  002464      E10012  002720      E10013  003016      E10023  004304
E10025  004524      E10031  005242      E10033  005534      E1005   002100
E1011   002612      E20007  002346      E20010  002476      E20011  002624
E20013  003100      E20025  004540      E20031  005254      E30013  003132
E40013  003174      F0013   003026      G0013   003042      H0013   003066
IOTVEC= 000020      I0001   001600      I0014   003264      I0015   003376
I0016   003510      I0017   003622      I0020   003734      I0021   004046
I0022   004162      I0023   004266      I0026   004642      J0013   003104
KSTART  001400      K0013   003120      LKS   = 177546      L0013   003136
M0013   003150      NOP   = 000240      N0013   003164      PC    =X000007
PIRQ  = 177772      PIRQVE= 000240      POWMES  007536      POWPOP  007750
POWPUS  007746      PS    = 177776      PSW   = 177776      PWRVEC= 000024
RESVEC= 000010      R0    =X000000      R0001   001574      R0002   001634
R0003   001710      R0004   001746      R0005   002036      R0006   002142
R0007   002260      R0010   002410      R0011   002524      R0012   002660
R0013   002724      R0014   003254      R0015   003366      R0016   003500
R0017   003612      R0020   003724      R0021   004036      R0022   004152
R0023   004242      R0024   004352      R0025   004456      R0026   004616
```

DZKWAD.P11 SYMBOL TABLE

```
R0027   004722    R0030   005036    R0031   005154    R0032   005302
R0033   005436    R1     =X000001   R1013   002736    R2     =X000002
R2013   002770    R3     =X000003   R3013   003052    R4     =X000004
R4013   003104    R5     =X000005   R6     =X000006   R7     =X000007
SP     =X000006   STACK  = 001100   START   001416   STKLMT= 177774
STMES   007604    SWR    = 177570   SW0    = 000001   SW00   = 000001
SW01   = 000002   SW02   = 000004   SW03   = 000010   SW04   = 000020
SW05   = 000040   SW06   = 000100   SW07   = 000200   SW08   = 000400
SW09   = 001000   SW1    = 000002   SW10   = 002000   SW11   = 004000
SW12   = 010000   SW13   = 020000   SW14   = 040000   SW15   = 100000
SW2    = 000004   SW3    = 000010   SW4    = 000020   SW5    = 000040
SW6    = 000100   SW7    = 000200   SW8    = 000400   SW9    = 001000
TBITVE= 000014    TKVEC  = 000060   TPVEC  = 000064   TRAPVE= 000034
TRAP0   005674    TRPMES  007661    TRPM2S  007722    TRTVEC= 000014
TYP0S  = 104410   TYPE   = 104400   TYP0C  = 104402   TYP0N  = 104406
TYP0S  = 104404   T0001    001550   T0002    001610   T0003    001656
T0004    001722   T0005    001776   T0006    002102   T0007    002212
T0010    002350   T0011    002500   T0012    002626   T0013    002722
T0014    003210   T0015    003352   T0016    003434   T0017    003546
T0020    003660   T0021    003772   T0022    004104   T0023    004210
T0024    004306   T0025    004410   T0026    004542   T0027    004656
T0030    005000   T0031    005130   T0032    005256   T0033    005412
T0034    005544   WORD     001230   $ERR0R   001122   $ERRDAT  001126
$CELL    001220   $CMTAG   001100   $CM1    = 000010   $CM2    = 000020
$CM3   = 000010   $CMM     000010   $CRLF    001225   $CRBLK   001540
$DORGN   005650   $DTBL    001530   $ENDAD   001540   $ENDCT   001510
$ENDMG   005654   $ENULL   001671   $ERP     001554   $EOPCT   001502
$ERRFLG  001103   $ERMAX   001115   $ERROR   007132   $ERRPC   001116
$ERRTB   001232   $ERRTY   006776   $ERTTL   001112   $ESCAP   001216
$FILLC   001150   $FILLS   001147   $GDADR   001120   $GDDAT   001124
$GET42   005632   $HD    = 000000   $ICNT    001104   $ILLUP   007520
$ITEMB   001114   $LF      001226   $LPADR   001106   $LPERR   001110
$MCONT   005214   $NULL    001146   $OVCNT   006772   $OVMODE  006774
$OVER    001200   $PARS    001100   $PWRER   007526   $PWRDN   007356
$PWRUP   007444   $QUES    001224   $REGAD   001152   $REGO    001154
$REG1    001156   $REG2    001160   $REG3    001162   $REG4    001164
$REG5    001166   $REG6    001170   $REG7    001172   $SAVR6   007524
$SCOPE   005746   $SETUP=  000037   $STUP  = 177777   $SVLAD   006152
$SWR   = 167400   $TIMES   001214   $TKB      001140   $TKS     001136
$TMP0    001174   $TMP1    001176   $TMP2    001200   $TMP3    001202
$TMP4    001204   $TMP5    001206   $TMP6    001210   $TMP7    001212
$TN    = 000000   $TPB     001144   $TPFLG   001151   $TPS     001142
$TRAP    007324   $TRP   = 000012   $TRPAD   007344   $TSTNM   001102
$TYPDS   006324   $TYPE    006216   $TYPOC   006574   $TYPON   006610
$TYPOS   006550   $XTSTR   005756   $OFILL   006773   .      = 011656
```

ERRORS DETECTED: 0

#DZKWAD,DZKWAD=DZKWAD.P11
RUN-TIME: 40 21 0 SECONDS
CORE USED: 14K

C06

Spooler runtime 11 Seconds, 27 [illegible], 271 disk reads, 0 disk writes, 67 pages

[illegible]

[illegible] 1111111111111111111111111111111110

[illegible] 0011111111112222222222233312

[illegible] 901234567890123456789012 **

[illegible] 1111111111111111111111111111111110

[illegible] 0000000001111111111222222222233312