DRV11

DRV11 TEST
CVKAFD0

AH-8207D-MC

FICHE 1 OF 1

JAN 1979
digital
MADE IN USA

IDENTIFICATION

PRODUCT CODE:	AC-8206D-MC
PRODUCT NAME:	CVKAFDO DRV11 TEST
PRODUCT DATE:	AUGUST 1978
MAINTAINER:	DIAGNOSTIC ENGINEERING

1. ABSTRACT

THIS IS A LOGIC TEST OF THE DRV11. TO ALLOW TESTING OF THE DATA LINES AND INTERRUPTS, A SPECIAL MAINTENANCE CABLE (BC08R) IS USED BY DEFAULT. ALSO, A SPECIAL TEST MODULE IS REQUIRED BY OPTION TO TEST THE NEWDATA RDY AND DATATRANS SIGNALS.
NOTE: THE SPECIAL TEST MODULE IS FOR USE BY IN HOUSE MANUFACTURING ONLY.
SEE SECTION 5.2

THIS TEST WILL OPERATE ON ONE DRV11. SPECIAL OPERATIONAL PROCEDURES ARE REQUIRED TO OPERATE ON OTHER THAN THE PRIMARY DRV11. SEE SEC. 5.4

2. REQUIREMENTS

2.1 EQUIPMENT

LSI-11

DRV11

TEST CABLE (BC08R) (BY OPTION)

TEST MODULE (BY OPTION)
(FOR IN HOUSE MANUFACTURING ONLY)

2.2 STORAGE

2.2.1 PROGRAM STORAGE - 4K

3. LOADING PROCEDURE

3.1 METHOD

ABSOLUTE LOADER

4. STARTING PROCEDURE

200 - NORMAL ENTRY TO TEST ONE DEVICE

TO LOAD AND EXECUTE

1. LOAD PROGRAM WITH THE ABSOLUTE LOADER.
2. IF ANY PROGRAM OPTIONS ARE REQUIRED, SET THE APPROPRIATE BIT IN THE SOFTWARE SWITCH REGISTER AT LOCATION 422. (REF. SECTION 5.1)
3. START PROGRAM AT 200.
4. PROGRAM WILL PRINT "END OF PASS" FOLLOWING EACH PASS.

4.1 CONTROL SWITCH SETTING

THIS PROGRAM CONTAINS A SOFTWARE SWITCH REGISTER FOR
OPTION SELECTION. FOR IT TO OPERATE THE OPERATOR MUST
SELECT THE APPROPRIATE OPTION BY SETTING OR
RESETTING THE RESPECTIVE BIT IN THE WORD.

TO DO THIS , THE LSI-11 MUST BE IN ODT MODE.

4.2 STARTING ADDRESS OR ADDRESSES

200 = START OF TEST--FOR NORMAL TESTING

5. OPERATING PROCEDURE

1. THE PROGRAM WILL CYCLE CONTINUOUSLY UNLESS HALTED
   BY THE OPERATOR, OR SOME ERROR CONDITION.
2. TO HALT THE PROGRAM, DEPRESS THE BREAK KEY. ODT
   WILL DISPLAY THE PC AT WHICH IT WAS HALTED.
3. IF NEW OPTIONS ARE TO BE SELECTED IN THE SWR, THEY MUST
   BE SET AT THIS TIME.
4. CONTINUE THE PROGRAM VIA A 'P'' OR A 'G'' COMMAND.

5.1 SOFTWARE SWITCH SETTINGS

BIT15 - CONTINUE ON ERROR	(100000)
BIT14 - LOOP ON CURRENT ERROR	(040000)
BIT13 - NOT USED	(020000)
BIT12 - NOT USED	(010000)
BIT11 - NOT USED	(004000)
BIT10 - LOOP ON CURRENT TEST	(002000)
BIT9 - RUN TEST MODULE	(001000)
BIT8 - INHIBIT WRAP CABLE	(000400)
BIT7 - NOT USED	(000200)
BIT6 - NOT USED	(000100)
BIT5 - NOT USED	(000040)
BIT4 - NOT USED	(000020)
BIT3 - NOT USED	(000010)
BIT2 - NOT USED	(000004)
BIT1 - NOT USED	(000002)
BIT0 - NOT USED	(000001)

5.2 SELECTION OF TEST OPTIONS

1. TO TEST NEWDATA RDY AND DATATRANS SIGNALS, THE
   SPECIAL WRAP MODULE MUST BE INSTALLED. THE OPERATOR
   MUST ALSO SET BIT9 IN THE SWITCH REGISTER (LOC. 422).
   NOTE: THE SPECIAL MODULE IS FOR USE BY IN HOUSE
   MANUFACTURING ONLY.

2. THIS TEST WILL RUN WITH THE WRAP CABLE BY DEFAULT.
   TO INHIBIT TESTING WITH THE WRAP CABLE, THE OPERATOR
   MUST SET BIT8 IN THE SWITCH REGISTER (LOC. 422).

5.3 WRAP CABLE

THE WRAP CABLE IS REQUIRED TO TEST TRANSFER OF DATA
INTO AND OUT OF THE INPUT BUFFER, AND THE DEVICE
INTERRUPTS.

NOTE !!!!! THIS DIAGNOSTIC IS APPROXIMATELY 95% EFFECTIVE
WHEN RUN WITH THE WRAP CABLE, AND APPROXIMATELY

60-70% EFFECTIVE WHEN RUN WITHOUT IT.

5.4 TESTING OTHER DRV11 MODULES

TO TEST A DRV11 NOT ADDRESSED AS 167770, OR
VECTORED AT 300, THE OPERATOR MUST SUPPLY THE NEW ADDRESSES
AND VECTORS TO THE PROGRAM BY DEPOSITING THEM AT
THE LOCATIONS TAGGED BY 'RCSR'' IN THE BEGINNING OF THE LISTING.
THE ORDER IS AS FOLLOWS:

```
RCSR:   CSR ADDRESS
        OUTPUT BUFFER ADDRESS
        INPUT BUFFER ADDRESS
        HIGH BYTE ADDR. OF OUTPUT BUFFER OR
                (OUTPUT BUFFER ADDR -1)
        'A' INTERRUPT VECTOR ADDRESS
        'A' ADDRESS + 2
        'B' INTERRUPT VECTOR ADDRESS
        'B' ADDRESS + 2
```

5.5 EXECUTION TIME

TYPICAL RUN TIMES (ONE PASS)

QUICK VERIFY	1 SEC.
WITH WRAP CABLE	10 SEC.

6. ERRORS

ALL ERROR REPORTS WITHIN THIS TEST ARE IN THE FORM
OF AN ERROR HALT. ON THE LSI-11, A HALT WILL FORCE
ODT TO DISPLAY THE PC+2 OF THE HALT. THIS IS
THE PRIMARY ERROR INDICATOR WITHIN THE PROGRAM.
UPON DETECTION OF AN ERROR,THE PROGRAM WILL PLACE THE
CURRENT ERROR NUMBER AND THE CURRENT TEST IN THE MAILBOX
(SEE IMPORTANT TAGS SEC. 8)
TO DETERMINE THE TYPE OF ERROR, THE OPERATOR MUST REFER-
ENCE THE LISTING.

6.1 ERROR RECOVERY

IN ORDER TO CONTINUE, THE OPERATOR MUST ISSUE A 'P'' TO
CONTINUE THE PROGRAM, OR MAY SET THE ERROR LOOP SWITCH
PRIOR TO CONTINUING.

6. ERRORS

6.1 ERROR REPORTING

ALL ERROR REPORTS WILL BE DONE VIA A HALT WITHIN THE
PROGRAM. THIS WILL CAUSE ODT TO DISPLAY THE PC+2 OF THE
ERROR HALT. AT THIS TIME THE OPERATOR MUST REFERENCE
THE LISTING TO DETERMINE THE ERROR DESCRIPTION.
THE NUMBER AT TAG $FATAL IN THE APT MAILBOX CONTAINS
THE ERROR NUMBER AND MAY BE USED TO REFERENCE THE DE-
SCRIPTION IN THE TABLE OF CONTENTS.

6.2 ERROR RECOVERY

IN ORDER TO CONTINUE, THE OPERATOR MUST ISSUE A 'P'' TO

CONTINUE THE PROGRAM, OR MAY SET THE ERROR LOOP SWITCH
PRIOR TO CONTINUING.

8. IMPORTANT TAGS

FOLLOWING IS A LIST OF IMPORTANT TAGS WITHIN THE LISTING

TAG	COMMENT
$MAIL	START OF THE PROGRAM MAILBOX. MANY CLUES TO PROBLEMS CAN BE FOUND HERE
$FATAL	ERROR NUMBER. USE THE TABLE OF CONTENTS TO LOCATE THE ERROR INFORMATION AND/OR CODE
$TESTN	CURRENT TEST NUMBER
$PASS	PASS COUNT OF THE PROGRAM WHEN ERROR WAS DETECTED OR PROGRAM HALTED
$SWREG	SOFTWARE SWITCH REGISTER
RCSR	START OF UNIT UNDER TEST ADDRESSES

10. LISTING

```
                                .TITLE CVKAFD
                                .ENABLE ABS
                                .NLIST  MD,MC,CND
                                .LIST   ME


000001                  X=1
000001                  N=1
```

```
                                ;REVISION       B
                                ;       1.      CHANGED TEST REQUIRING WRAP CABLE TO BE
                                ;               INHIBITED BY BIT 8 OF THE SWR
                                ;       2.      INITIALIZED A & B INTERRUPT VECTOR LOCATIONS
                                ;               WITH TRAP CATCHER

                                ;REVISION       C
                                ;       1.      DOCUMENTATION CHANGE ONLY

                                ;REVISION       D
                                ;       1.      ADDED CODE TO PRINT OUT PROGRAMS TITLE
```

```
                        ;GENERAL REGISTER LOGIC TEST

104000                  HLT=104000
167770                  CSR=167770
001200                  STKPTR=1200
                        ;REGISTER DEFINITIONS
000000                          R0=%0
000001                          R1=%1
000002                          R2=%2
000003                          R3=%3
000004                          R4=%4
000005                          R5=%5
000006                          SP=%6
000007                          PC=%7

                        ;SWITCHES

001000                          SW9=1000
002000                          SW10=2000
004000                          SW11=4000
020000                          SW13=20000
040000                          SW14=40000

                        .SBTTL  BASIC DEFINITIONS

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

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

                        ;*GENERAL PURPOSE REGISTER DEFINITIONS
000000                  R0=     %0              ;;GENERAL REGISTER
000001                  R1=     %1              ;;GENERAL REGISTER
000002                  R2=     %2              ;;GENERAL REGISTER
000003                  R3=     %3              ;;GENERAL REGISTER
000004                  R4=     %4              ;;GENERAL REGISTER
000005                  R5=     %5              ;;GENERAL REGISTER
000006                  R6=     %6              ;;GENERAL REGISTER
000007                  R7=     %7              ;;GENERAL REGISTER
000006                  SP=     %6              ;;STACK POINTER
000007                  PC=     %7              ;;PROGRAM COUNTER

                        ;*PRIORITY LEVEL DEFINITIONS
000000                  PR0=    0               ;;PRIORITY LEVEL 0
```

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

                        ;*''SWITCH REGISTER'' SWITCH DEFINITIONS
100000                  SW15=   100000
040000                  SW14=   40000
020000                  SW13=   20000
010000                  SW12=   10000
004000                  SW11=   4000
002000                  SW10=   2000
001000                  SW09=   1000
000400                  SW08=   400
000200                  SW07=   200
000100                  SW06=   100
000040                  SW05=   40
000020                  SW04=   20
000010                  SW03=   10
000004                  SW02=   4
000002                  SW01=   2
000001                  SW00=   1
                        .EQUIV  SW09,SW9
                        .EQUIV  SW08,SW8
                        .EQUIV  SW07,SW7
                        .EQUIV  SW06,SW6
                        .EQUIV  SW05,SW5
                        .EQUIV  SW04,SW4
                        .EQUIV  SW03,SW3
                        .EQUIV  SW02,SW2
                        .EQUIV  SW01,SW1
                        .EQUIV  SW00,SW0

                        ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
100000                  BIT15=  100000
040000                  BIT14=  40000
020000                  BIT13=  20000
010000                  BIT12=  10000
004000                  BIT11=  4000
002000                  BIT10=  2000
001000                  BIT09=  1000
000400                  BIT08=  400
000200                  BIT07=  200
000100                  BIT06=  100
000040                  BIT05=  40
000020                  BIT04=  20
000010                  BIT03=  10
000004                  BIT02=  4
000002                  BIT01=  2
000001                  BIT00=  1
                        .EQUIV  BIT09,BIT9
                        .EQUIV  BIT08,BIT8
                        .EQUIV  BIT07,BIT7
```

```
                                .EQUIV  BIT06,BIT6
                                .EQUIV  BIT05,BIT5
                                .EQUIV  BIT04,BIT4
                                .EQUIV  BIT03,BIT3
                                .EQUIV  BIT02,BIT2
                                .EQUIV  BIT01,BIT1
                                .EQUIV  BIT00,BIT0

                                ;*BASIC ''CPU'' TRAP VECTOR ADDRESSES
            000004              ERRVEC= 4                   ;;TIME OUT AND OTHER ERRORS
            000010              RESVEC= 10                  ;;RESERVED AND ILLEGAL INSTRUCTIONS
            000014              TBITVEC=14                  ;;''T'' BIT
            000014              TRTVEC= 14                  ;;TRACE TRAP
            000014              BPTVEC= 14                  ;;BREAKPOINT TRAP (BPT)
            000020              IOTVEC= 20                  ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
            000024              PWRVEC= 24                  ;;POWER FAIL
            000030              EMTVEC= 30                  ;;EMULATOR TRAP (EMT) **ERROR**
            000034              TRAPVEC=34                  ;;''TRAP'' TRAP
            000060              TKVEC=  60                  ;;TTY KEYBOARD VECTOR
            000064              TPVEC=  64                  ;;TTY PRINTER VECTOR
            000240              PIRQVEC=240                 ;;PROGRAM INTERRUPT REQUEST VECTOR
                                        .ENABLE ABS
            000000                      .=0
000000  000002                          .+2
000002  000000                          HALT
000004  000006                          .+2
000006  000000                          HALT
000010  000012                          .+2
000012  000000                          HALT
000014  000016                          .+2
000016  000000                          HALT
000020  000022                          .+2
000022  000000                          HALT
000024  000026                          .+2
000026  000000                          HALT
000030  000032                          .+2
000032  000000                          HALT
000034  000036                          .+2
000036  000000                          HALT
000040  000042                          .+2
000042  000000                          HALT
000044  000046                          .+2
000046  000000                          HALT
000050  000052                          .+2
000052  000000                          HALT
000054  000056                          .+2
000056  000000                          HALT
000060  000062                          .+2
000062  000000                          HALT
000064  000066                          .+2
000066  000000                          HALT
            000100                      .=100
000100  000102                          102
000102  000002                          RTI             ; RTI FOR POSSIBLE CLOCK INTERRUPT
000104  000106                          .+2
000106  000000                          HALT
```

```
  194  000110  000112                        .+2
  195  000112  000000                        HALT
  196  000114  000116                        .+2
  197  000116  000000                        HALT
  198  000120  000122                        .+2
  199  000122  000000                        HALT
  200  000124  000126                        .+2
  201  000126  000000                        HALT
  202  000130  000132                        .+2
  203  000132  000000                        HALT
  204  000134  000136                        .+2
  205  000136  000000                        HALT
  206  000140  000142                        .+2
  207  000142  000000                        HALT
  208  000144  000146                        .+2
  209  000146  000000                        HALT
  210  000150  000152                        .+2
  211  000152  000000                        HALT
  212  000154  000156                        .+2
  213  000156  000000                        HALT
  214  000160  000162                        .+2
  215  000162  000000                        HALT
  216  000164  000166                        .+2
  217  000166  000000                        HALT
  218  000170  000172                        .+2
  219  000172  000000                        HALT
  220          000200                        .=200
  221  000200  005067  000202                CLR     $PASS           ; CLEAR PASS COUNT
  222  000204  005067  000172                CLR     $FATAL
  223  000210  005067  000170                CLR     $TESTN
  224  000214  000137  001246                JMP     @#START1        ;INITIAL START
  225          000300                        .=300                   ;DEVICE INTERRUPT VECTORS
  226  000300  000302                        .+2
  227  000302  000000                        HALT
  228  000304  000306                        .+2
  229  000306  000000                        HALT
  230          000400                        .=400
  231                              .SBTTL  APT MAILBOX-ETABLE
  232
  233                              ;;*****************************************************************
  234                              .EVEN
  235  000400                      $MAIL:                          ;;APT MAILBOX
  236  000400  000000              $MSGTY: .WORD   AMSGTY          ;;MESSAGE TYPE CODE
  237  000402  000000              $FATAL: .WORD   AFATAL          ;;FATAL ERROR NUMBER
  238  000404  000000              $TESTN: .WORD   ATESTN          ;;TEST NUMBER
  239  000406  000000              $PASS:  .WORD   APASS           ;;PASS COUNT
  240  000410  000000              $DEVCT: .WORD   ADEVCT          ;;DEVICE COUNT
  241  000412  000000              $UNIT:  .WORD   AUNIT           ;;I/O UNIT NUMBER
  242  000414  000000              $MSGAD: .WORD   AMSGAD          ;;MESSAGE ADDRESS
  243  000416  000000              $MSGLG: .WORD   AMSGLG          ;;MESSAGE LENGTH
  244  000420                      $ETABLE:                        ;;APT ENVIRONMENT TABLE
  245  000420      000             $ENV:   .BYTE   AENV            ;;ENVIRONMENT BYTE
  246  000421      000             $ENVM:  .BYTE   AENVM           ;;ENVIRONMENT MODE BITS
  247  000422  000000              $SWREG: .WORD   ASWREG          ;;APT SWITCH REGISTER
  248  000424  000000              $USWR:  .WORD   AUSWR           ;;USER SWITCHES
  249  000426  000000              $CPUOP: .WORD   ACPUOP          ;;CPU TYPE,OPTIONS
```

```
                                ;*                      BITS 15-11=CPU TYPE
                                ;*                           11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
                                ;*                           11/70=06,PDQ=07,Q=10
                                ;*                      BIT 10=REAL TIME CLOCK
                                ;*                      BIT  9=FLOATING POINT PROCESSOR
                                ;*                      BIT  8=MEMORY MANAGEMENT
000430                          $ETEND:
                                .MEXIT
                                .SBTTL  APT PARAMETER BLOCK

                                ;;*****************************************************************
                                ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
                                ;;*****************************************************************
        000430                          .$X=.   ;;SAVE CURRENT LOCATION
        000024                          .=24    ;;SET POWER FAIL TO POINT TO START OF PROGRAM
000024  000200                          200     ;;FOR APT START UP
        000044                          .=44    ;;POINT TO APT INDIRECT ADDRESS PNTR.
000044  000430                          $APTHDR ;;POINT TO APT HEADER BLOCK
        000430                          .=.$X   ;;RESET LOCATION COUNTER
                                ;;*****************************************************************
                                ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
                                ;INTERFACE SPEC.

000430                          $APTHD:
000430  000000                  $HIBTS: .WORD   0       ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
000432  000400                  $MBADR: .WORD   $MAIL   ;;ADDRESS OF APT MAILBOX (BITS 0-15)
000434  000010                  $TSTM:  .WORD   10      ;;RUN TIM OF LONGEST TEST
000436  000010                  $PASTM: .WORD   10      ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
000440  000000                  $UNITM: .WORD   0       ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
000442  000014                          .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
        001200                          .=1200
                                DEVCNT=$DEVCT
        000410                  ERRNUM=$FATAL
        000402                  OPTION=$CPUOP
        000426
                                ;THIS TABLE CONTAINS INITIAL REGISTER AND VECTOR ADDRESSES

001200  167770                  RCSR:   CSR
001202  167772                          CSR+2
001204  167774                          CSR+4
001206  167773                          CSR+3
001210  000300                  RCSR1:  300
001212  000302                          302
001214  000304                          304
001216  000306                          306

                                ;THIS TABLE CONTAINS REGISTER AND VECTOR ADDRESSES OF THE DR11-C UNDER TEST

001220  167770                  DRCSR:  167770          ;ADDRESS OF DR11-C STATUS REGISTER
001222  167772                  DROBUF: 167772          ;ADDRESS OF DR OUTPUT BUFFER REG.
001224  167774                  DRIBUF: 167774          ;ADDRESS OF DR INPUT BUFFER REG.
001226  167773                  DRBHIO: 167773          ;HIGH BYTE OF OUTPUT BUFFER REG.

001230  000300                  DRVECA: 300             ;INTERRUPT VECTOR OF UNIT UNDER TEST
001232  000302                  DRLVLA: 302
001234  000304                  DRVECB: 304             ;INTERRUPT VECTOR
```

```
306  001236  000306                  DRLVLB: 306
307  001240  000000                  XORFLC: 0
308
309  001242  000000                  COUNT:  0                       ;COUNT LOCATION
310  001244  000240                  PL:     240                     ;PRIORITY LEVEL
311
312  001246  012706  001200          START1: MOV     #STKPTR,SP
313  001252  000137  001256                  JMP     @#START
314                                  ;+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
315
316                                  ;+              ROUTINE TO PRINT DIAGNOSTIC'S TITLE IF  ALLOWED BY  APT MODE
317
318                                  ;  ALL CODE ADDED FOR TITLE PRINTOUT IS NOTED
319                                  ; WITH   A ;+ IN THE  COMMENT SECTION.
320                                  ;+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
321  001256  132767  000040  177135  START:  BITB  #40, $ENVM           ;+ WILL APT ALLOW PRINTING?
322  001264  001010                   BNE AROUND                      ;+ NO BRANCH AROUND
323  001266  012700  004602           MOV #TITLE1,R0                  ;+ GET STARTING ADDRESS OF TITLE MSG
324  001272  105737  177564  LOOP1:   TSTB @#TPS                      ;+ CHECK IF TERMINAL READY
325  001276  100375                   BPL     LOOP1                   ;+ LOOP IF NOT
326  001300  112037  177566           MOVB (R0)+,@#TPB                ;+ PRINT A CHARACTER
327  001304  001372                   BNE LOOP1                       ;+ BRANCH BACK IF NOT DONE PRINTING
328                                  ;+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
329
330                                  ; INITIALIZE ADDRESS AND VECTORS
331
332  001306  012700  001200  AROUND: MOV #RCSR, R0                    ; GET ADDRESS OF UNIT UNDER TEST
333  001312  012701  001220           MOV #DRCSR,R1
334
335  001316  012737  004656  000024           MOV     #PFAIL,@#24
336  001324  012021                           MOV     (R0)+,(R1)+     ;LOAD INITIAL TEST ADDRESSES
337  001326  012021                           MOV     (R0)+,(R1)+
338  001330  012021                           MOV     (R0)+,(R1)+
339  001332  012021                           MOV     (R0)+,(R1)+
340  001334  012021                           MOV     (R0)+,(R1)+
341  001336  012021                           MOV     (R0)+,(R1)+
342  001340  012021                           MOV     (R0)+,(R1)+
343
344                                  ;DOES RESET CLEAR REGISTER?
345  001342                          TST1:
346  001342  012767  000001  177034           MOV     #1,$TESTN       ; MOVE TEST NUMBER TO MAILBOX
347  001350  016705  177644  LP1:     MOV     DRCSR,R5                ;GET ADDRESS OF STATUS REGISTER
348  001354  106427  000200           MTPS    #200                    ;TURN OFF INTERRUPTS
349  001360  016737  000056  000004   MOV     ERR1,@#4                        ;SET TIME OUT TRAP VECTOR
350  001366  012777  177777  177626   MOV     #-1,@DROBUF             ;PRESET OUTPUT BUFFER
351  001374  000005                   RESET                           ;CLEAR DATA REGISTER
352  001376  017700  177620           MOV     @DROBUF,R0              ;GET RESULT OF RESET
353  001402  001450                   BEQ     CON
354  001404  032767  040000  177010   BIT     #BIT14,$SWREG           ; CHECK FOR LOOP ON ERROR
355  001412  001356                   BNE     LP1                     ; GO TO LOOP ERROR
356  001414  012767  000001  176760   MOV     #1,$FATAL
357  001422  012767  000001  176750   MOV     #1,$MSGTY               ; MOVE ERROR NUM TO MAILBOX
358  001430  005767  176766           TST     $SWREG                  ; CHECK FOR HALT ON ERROR
359  001434  100401                   BMI     1$                      ; CONTINUE IF SET
360  001436  000000                   HALT                    ;<DATA REG DID NOT CLEAR>
361  001440                          1$:
```

```
 362  001440  000431                          BR      CON
 363  001442                          ERR1:
 364  001442  032767  040000  176752          BIT     #BIT14,$SWREG   ; CHECK FOR LOOP ON ERROR
 365  001450  001337                          BNE     LP1             ; GO TO LOOP ERROR
 366  001452  012767  000002  176722          MOV     #2,$FATAL
 367  001460  012767  000001  176712          MOV     #1,$MSGTY       ; MOVE ERROR NUM TO MAILBOX
 368  001466  005767  176730                  TST     $SWREG          ; CHECK FOR HALT ON ERROR
 369  001472  100401                          BMI     1$              ; CONTINUE IF SET
 370  001474  000000                          HALT            ;<TIME WHEN ADDRESSING PLU>
 371  001476                          1$:
 372  001476  032767  002000  176716          BIT     #BIT10,$SWREG   ; CHECK FOR LOOP ON TEST
 373  001504  001316                          BNE     TST1            ; GO TO LOOP ON TEST
 374  001506  000407                          BR      TST2
 375  001510  012706  001200          5$:     MOV     #STKPTR,SP      ;RESET STACK POINTER
 376  001514  012737  000006  000004          MOV     #6,@#4          ;RESTORE TIME OUT TRAP
 377  001522  000765                          BR      1$
 378  001524  000772                  CON:    BR      .-12
 379                                  ; TEST 'NEWDATA RDY'' AND 'DATATRANS'' SIGNALS IN PLU
 380                                  ;  NOTE****** THE PLU TEST MODULE MUST BE INSTALLED
 381                                  ;               TO EXECUTE THIS TEST
 382                                  ;
 383  001526                          TST2:
 384  001526  012767  000002  176650          MOV     #2,$TESTN       ; MOVE TEST NUMBER TO MAILBOX
 385  001534  032767  001000  176660          BIT     #BIT9,$SWREG
 386  001542  001505                          BEQ     TST3            ; SKIP TEST IF NOT SELECTED
 387  001544  012706  001200                  MOV     #STKPTR,SP              ; SET UP STACK POINTER
 388  001550  000005                          RESET                           ; CLEAR EVERYTHING
 389                                                  ; THIS RESET SHOULD INITIALIZE THE
 390                                                  ; SIGNAL LATCHES IN THE TEST MODULE
 391  001552  012777  031460  177442          MOV     #31460,@DROBUF          ; PRIME THE LATCHES
 392  001560  000240                          NOP
 393  001562  000240                          NOP             ; TIMING ALLOWANCE
 394  001564  017700  177434                  MOV     @DRIBUF,R0              ; GET DATA
 395  001570  032700  000001                  BIT     #BIT0,R0                ; CHECK DATA TRANS SIG
 396  001574  001016                          BNE     T2CON           ; CONTINUE IF PRESENT
 397  001576  032767  040000  176616          BIT     #BIT14,$SWREG   ; CHECK FOR LOOP ON ERROR
 398  001604  001350                          BNE     TST2            ; GO TO LOOP ERROR
 399  001606  012767  000003  176566          MOV     #3,$FATAL
 400  001614  012767  000001  176556          MOV     #1,$MSGTY       ; MOVE ERROR NUM TO MAILBOX
 401  001622  005767  176574                  TST     $SWREG          ; CHECK FOR HALT ON ERROR
 402  001626  100401                          BMI     1$              ; CONTINUE IF SET
 403  001630  000000                          HALT            ;<NO DATA TRANS SIGNAL>
 404  001632                          1$:
 405  001632  032700  000002          T2CON:  BIT     #BIT1,R0        ; CHECK NEW DATA RDY SIGNAL
 406  001636  001016                          BNE     T2CN1           ; CONTINUE IF OK
 407  001640  032767  040000  176554          BIT     #BIT14,$SWREG   ; CHECK FOR LOOP ON ERROR
 408  001646  001371                          BNE     T2CON           ; GO TO LOOP ERROR
 409  001650  012767  000004  176524          MOV     #4,$FATAL
 410  001656  012767  000001  176514          MOV     #1,$MSGTY       ; MOVE ERROR NUM TO MAILBOX
 411  001664  005767  176532                  TST     $SWREG          ; CHECK FOR HALT ON ERROR
 412  001670  100401                          BMI     1$              ; CONTINUE IF SET
 413  001672  000000                          HALT            ;<NO NEW DATA RDY SIGNAL>
 414  001674                          1$:
 415  001674  000005                  T2CN1:  RESET                   ; CLEAR EVERYTHING
 416  001676  000240                          NOP
 417  001700  000240                          NOP
```

```
 418  001702  017700  177316                MOV     @DRIBUF,R0      ; CHECK SIGNAL LATCHES
 419  001706  005700                        TST     R0              ; SHOULD BE CLEAR
 420  001710  001416                        BEQ     1$              ; CONTINUE IF CLEAR
 421  001712  032767  040000  176502        BIT     #BIT14,$SWREG   ; CHECK FOR LOOP ON ERROR
 422  001720  001365                        BNE     T2CN1           ; GO TO LOOP ERROR
 423  001722  012767  000005  176452        MOV     #5,$FATAL
 424  001730  012767  000001  176442        MOV     #1,$MSGTY       ; MOVE ERROR NUM TO MAILBOX
 425  001736  005767  176460                TST     $SWREG          ; CHECK FOR HALT ON ERROR
 426  001742  100401                        BMI     1$              ; CONTINUE IF SET
 427  001744  000000                        HALT            ;<SIGNALS DID NOT CLEAR>
 428  001746                        1$:
 429  001746  032767  002000  176446        BIT     #BIT10,$SWREG   ; CHECK FOR LOOP ON TEST
 430  001754  001264                        BNE     TST2            ; GO TO LOOP ON TEST
 431  001756                        TST3:
 432  001756  012767  000003  176420        MOV     #3,$TESTN       ; MOVE TEST NUMBER TO MAILBOX
 433  001764  012777  177777  177230        MOV     #-1,@DROBUF     ;ALL ONES TO REGISTER
 434  001772  017700  177224                MOV     @DROBUF,R0
 435  001776  022700  177777                CMP     #-1,R0
 436  002002  001416                        BEQ     1$
 437  002004  032767  040000  176410        BIT     #BIT14,$SWREG   ; CHECK FOR LOOP ON ERROR
 438  002012  001361                        BNE     TST3            ; GO TO LOOP ERROR
 439  002014  012767  000006  176360        MOV     #6,$FATAL
 440  002022  012767  000001  176350        MOV     #1,$MSGTY       ; MOVE ERROR NUM TO MAILBOX
 441  002030  005767  176366                TST     $SWREG          ; CHECK FOR HALT ON ERROR
 442  002034  100401                        BMI     1$              ; CONTINUE IF SET
 443  002036  000000                        HALT            ;<REGISTER WILL NOT HOLD ALL ONES>
 444  002040                        1$:
 445  002040  032767  002000  176354        BIT     #BIT10,$SWREG   ; CHECK FOR LOOP ON TEST
 446  002046  001343                        BNE     TST3            ; GO TO LOOP ON TEST
 447
 448                        ;WRAP CABLE MUST BE INSTALLED TO EXECUTE THIS TEST
 449  002050                        TST4:
 450  002050  012767  000004  176326        MOV     #4,$TESTN       ; MOVE TEST NUMBER TO MAILBOX
 451  002056  032767  000400  176336        BIT     #BIT8,$SWREG
 452  002064  001031                        BNE     TST5            ;SKIP TEST IF NOT SELECTED
 453  002066  012777  177777  177126        MOV     #-1,@DROBUF
 454  002074  000005                        RESET                   ;SET DATA TO ALL ONES
 455  002076  005777  177122                TST     @DRIBUF         ;REGISTER SHOULD CLEAR
 456  002102  001416                        BEQ     1$
 457  002104  032767  040000  176310        BIT     #BIT14,$SWREG   ; CHECK FOR LOOP ON ERROR
 458  002112  001356                        BNE     TST4            ; GO TO LOOP ERROR
 459  002114  012767  000007  176260        MOV     #7,$FATAL
 460  002122  012767  000001  176250        MOV     #1,$MSGTY       ; MOVE ERROR NUM TO MAILBOX
 461  002130  005767  176266                TST     $SWREG          ; CHECK FOR HALT ON ERROR
 462  002134  100401                        BMI     1$              ; CONTINUE IF SET
 463  002136  000000                        HALT            ;<REGISTER DID NOT CLEAR BY RESET>
 464  002140                        1$:
 465  002140  032767  002000  176254        BIT     #BIT10,$SWREG   ; CHECK FOR LOOP ON TEST
 466  002146  001340                        BNE     TST4            ; GO TO LOOP ON TEST
 467
 468  002150                        TST5:
 469  002150  012767  000005  176226        MOV     #5,$TESTN       ; MOVE TEST NUMBER TO MAILBOX
 470  002156  012777  052525  177036        MOV     #52525,@DROBUF  ;LOAD TEST DATA INTO BUFFER
 471  002164  017700  177032                MOV     @DROBUF,R0      ;COPY DATA FROM BUFFER TO R0
 472  002170  022700  052525                CMP     #52525,R0       ;COMPARE DATA
 473  002174  001416                        BEQ     1$              ;BR IF DATA IS CORRECT
```

```
                                                        BIT     #BIT14,$SWREG   ; CHECK FOR LOOP ON ERROR
                                                        BNE     TST5            ; GO TO LOOP ERROR
                                                        MOV     #10,$FATAL
                                                        MOV     #1,$MSGTY       ; MOVE ERROR NUM TO MAILBOX
                                                        TST     $SWREG          ; CHECK FOR HALT ON ERROR
                                                        BMI     1$              ; CONTINUE IF SET
                                                        HALT            ;<INCORRECT DATA IN REG>
                                                1$:
                                                        BIT     #BIT10,$SWREG   ; CHECK FOR LOOP ON TEST
                                                        BNE     TST5            ; GO TO LOOP ON TEST

                                                TST6:
                                                        MOV     #6,$TESTN       ; MOVE TEST NUMBER TO MAILBOX
                                                        MOV     #125252,@DROBUF ;LOAD TEST DATA INTO BUFFER
                                                        MOV     @DROBUF,R0      ;COPY DATA FROM BUFFER TO R0
                                                        CMP     #125252,R0      ;COMPARE DATA
                                                        BEQ     1$              ;BR IF DATA IS CORRECT
                                                        BIT     #BIT14,$SWREG   ; CHECK FOR LOOP ON ERROR
                                                        BNE     TST6            ; GO TO LOOP ERROR
                                                        MOV     #11,$FATAL
                                                        MOV     #1,$MSGTY       ; MOVE ERROR NUM TO MAILBOX
                                                        TST     $SWREG          ; CHECK FOR HALT ON ERROR
                                                        BMI     1$              ; CONTINUE IF SET
                                                        HALT            ;<INCORRECT DATA IN REG>
                                                1$:
                                                        BIT     #BIT10,$SWREG   ; CHECK FOR LOOP ON TEST
                                                        BNE     TST6            ; GO TO LOOP ON TEST

                                                ;TEST RELIABILITY OF DR11-C OUTPUT BUFFER REGISTER
                                                TST7:
                                                        MOV     #7,$TESTN       ; MOVE TEST NUMBER TO MAILBOX
                                                BUFTST: MOV     R5,R2           ;GET ADDRESS OF DRCSR
                                                        TST     (R2)+           ;R2=ADDRESS OF OUTPUT BUFFER REG.
                                                        CLR     R3              ; INITIALIZE DATA REGISTER
                                                LP7:    MOV     R3,(R2)         ; SEND THE DATA
                                                        CMP     (R2),R3         ; CHECK THE RECEIVED DATA
                                                        BNE     5$              ; ERROR IF NOT THE SAME
                                                        INC     R3              ; INCREMENT THE DATA
                                                        TSTB    R3              ; CHECK FOR END OF DATA
                                                        BEQ     1$              ; CONTINUE IF END
                                                        BR      LP7             ; GO TO SEND DATA IF NOT
                                                5$:
                                                        BIT     #BIT14,$SWREG   ; CHECK FOR LOOP ON ERROR
                                                        BNE     LP7             ; GO TO LOOP ERROR
                                                        MOV     #12,$FATAL
                                                        MOV     #1,$MSGTY       ; MOVE ERROR NUM TO MAILBOX
                                                        TST     $SWREG          ; CHECK FOR HALT ON ERROR
                                                        BMI     1$              ; CONTINUE IF SET
                                                        HALT            ;<DATA INCORRECT IN REG>
                                                1$:
                                                        BIT     #BIT10,$SWREG   ; CHECK FOR LOOP ON TEST
                                                        BNE     TST7            ; GO TO LOOP ON TEST
                                                ;TEST THAT BYTE REFERENCE TO DROBUF AFFECT PROPER BYTE ONLY

                                                TST10:
                                                        MOV     #10,$TESTN      ; MOVE TEST NUMBER TO MAILBOX
```

```
                                TAG:    MOV     #-1,@DROBUF     ;SET ALL ONES IN BUFFER
                                        CLRB    @DROBUF         ;CLEAR LOW BYTE
                                        MOV     @DROBUF,R0      ;COPY DATA
                                        CMP     #177400,R0      ;VERIFY THAT LOW BYTE IS CLEAR
                                        BEQ     1$
                                        BIT     #BIT14,$SWREG   ; CHECK FOR LOOP ON ERROR
                                        BNE     TAG             ; GO TO LOOP ERROR
                                        MOV     #13,$FATAL
                                        MOV     #1,$MSGTY       ; MOVE ERROR NUM TO MAILBOX
                                        TST     $SWREG          ; CHECK FOR HALT ON FRROR
                                        BMI     1$              ; CONTINUE IF SET
                                        HALT            ;<LOW BYTE FAILED TO CLEAR>
                                1$:
                                        BIT     #BIT10,$SWREG   ; CHECK FOR LOOP ON TEST
                                        BNE     TST10           ; GO TO LOOP ON TEST

                                TST11:
                                        MOV     #11,$TESTN      ; MOVE TEST NUMBER TO MAILBOX
                                        MOV     #-1,@DROBUF     ;SET ALL ONES IN BUFFER
                                        CLRB    @DRBHIO         ;CLEAR HIGH BYTE
                                        MOV     @DROBUF,R0
                                        CMP     #377,R0         ;VERIFY THAT HIGH BYTE IS CLEAR
                                        BEQ     1$
                                        BIT     #BIT14,$SWREG   ; CHECK FOR LOOP ON ERROR
                                        BNE     TST11           ; GO TO LOOP ERROR
                                        MOV     #14,$FATAL
                                        MOV     #1,$MSGTY       ; MOVE ERROR NUM TO MAILBOX
                                        TST     $SWREG          ; CHECK FOR HALT ON ERROR
                                        BMI     1$              ; CONTINUE IF SET
                                        HALT            ;<HIGH BYTE FAILED TO CLEAR>
                                1$:
                                        BIT     #BIT10,$SWREG   ; CHECK FOR LOOP ON TEST
                                        BNE     TST11           ; GO TO LOOP ON TEST
                                TST12:
                                        MOV     #12,$TESTN      ; MOVE TEST NUMBER TO MAILBOX
```

```
565  002634  005067  000110              CLR     T12DAT          ;CLEAR DATA LOCATION
566  002640  012704  002750              MOV     #T12DAT,R4      ;STORE ADDRESS OF DATA LOCATION
567  002644  005077  176352              CLR     @DROBUF         ;CLEAR OUTPUT BUFFER
568  002650  105077  176352  T12CON:     CLRB    @DRBHIO         ;CLEAR HIGH BYTE
569  002654  105277  176346  3$:         INCB    @DRBHIO         ;INCREMENT HIGH BYTE
570  002660  105264  000001              INCB    1(R4)           ;INCREMENT COMPARISON DATA
571  002664  027714  176332              CMP     @DROBUF,(R4)    ;COMPARE DATA
572  002670  001004                      BNE     6$              ;BRANCH ON ERROR
573  002672  105764  000001  4$:         TSTB    1(R4)           ;FINISHED?
574  002676  001417                      BEQ     1$              ;YES
575  002700  000765                      BR      3$              ;CONTINUE TESTING
576  002702                      6$:
577  002702  032767  040000  175512      BIT     #BIT14,$SWREG   ; CHECK FOR LOOP ON ERROR
578  002710  001346                      BNE     TST12           ; GO TO LOOP ERROR
579  002712  012767  000015  175462      MOV     #15,$FATAL
580  002720  012767  000001  175452      MOV     #1,$MSGTY       ; MOVE ERROR NUM TO MAILBOX
581  002726  005767  175470              TST     $SWREG          ; CHECK FOR HALT ON ERROR
582  002732  100401                      BMI     1$              ; CONTINUE IF SET
583  002734  000000                      HALT            ;<DATA INCORRECT IN REG>
584  002736                      1$:
585  002736  032767  002000  175456      BIT     #BIT10,$SWREG   ; CHECK FOR LOOP ON TEST
586  002744  001330                      BNE     TST12           ; GO TO LOOP ON TEST
587  002746  000401                      BR      TST13
588  002750  000000              T12DAT: .WORD   0
589                              ;CONTROL STATUS REGISTER (DRCSR) TESTS.
590  002752                      TST13:
591  002752  012767  000013  175424      MOV     #13,$TESTN      ; MOVE TEST NUMBER TO MAILBOX
592  002760  005015                      CLR     (R5)            ;CLEAR STATUS REGISTER
593  002762  011500                      MOV     (R5),R0         ;COPY DATA
594  002764  032700  000143              BIT     #143,R0         ;VERIFY THAT IE AND CSR BITS ARE CLEAR
595  002770  001416                      BEQ     T13CON          ;IF YES, CONTINUE
596  002772  032767  040000  175422      BIT     #BIT14,$SWREG   ; CHECK FOR LOOP ON ERROR
597  003000  001364                      BNE     TST13           ; GO TO LOOP ERROR
598  003002  012767  000016  175372      MOV     #16,$FATAL
599  003010  012767  000001  175362      MOV     #1,$MSGTY       ; MOVE ERROR NUM TO MAILBOX
600  003016  005767  175400              TST     $SWREG          ; CHECK FOR HALT ON ERROR
601  003022  100401                      BMI     1$              ; CONTINUE IF SET
602  003024  000000                      HALT            ;<STATUS REG DID NOT CLEAR>
603  003026                      1$:
604  003026  012715  000140  T13CON:     MOV     #140,@R5        ;INTERRUPT ENABLE FOR A+B
605  003032  011500                      MOV     @R5,R0
606  003034  032700  000140              BIT     #140,R0         ;INTERRUPT ENABLE BITS SET?
607  003040  001016                      BNE     1$              ;CONTINUE IF YES
608  003042  032767  040000  175352      BIT     #BIT14,$SWREG   ; CHECK FOR LOOP ON ERROR
609  003050  001366                      BNE     T13CON          ; GO TO LOOP ERROR
610  003052  012767  000017  175322      MOV     #17,$FATAL
611  003060  012767  000001  175312      MOV     #1,$MSGTY       ; MOVE ERROR NUM TO MAILBOX
612  003066  005767  175330              TST     $SWREG          ; CHECK FOR HALT ON ERROR
613  003072  100401                      BMI     1$              ; CONTINUE IF SET
614  003074  000000                      HALT            ;<ENABLE BITS NOT ON>
615  003076                      1$:
616  003076  032767  002000  175316      BIT     #BIT10,$SWREG   ; CHECK FOR LOOP ON TEST
617  003104  001322                      BNE     TST13           ; GO TO LOOP ON TEST
618
619  003106                      TST14:
620  003106  012767  000014  175270      MOV     #14,$TESTN      ; MOVE TEST NUMBER TO MAILBOX
```

```
621  003114  012715  000140            MOV     #140,@R5        ;SET INTERRUPT ENABLE FLOPS
622  003120  000005                    RESET                   ;CLEAR THOSE FLOPS
623  003122  011500                    MOV     @R5,R0          ;COPY CONTENTS OF DRCSR TO R0
624  003124  032700  000140            BIT     #140,R0         ;TEST INTERRUPT ENABLE BITS
625  003130  001416                    BEQ     1$              ;BR IF CLEARED
626  003132  032767  040000  175262    BIT     #BIT14,$SWREG   ; CHECK FOR LOOP ON ERROR
627  003140  001362                    BNE     TST14           ; GO TO LOOP ERROR
628  003142  012767  000020  175232    MOV     #20,$FATAL
629  003150  012767  000001  175222    MOV     #1,$MSGTY       ; MOVE ERROR NUM TO MAILBOX
630  003156  005767  175240            TST     $SWREG          ; CHECK FOR HALT ON ERROR
631  003162  100401                    BMI     1$              ; CONTINUE IF SET
632  003164  000000                    HALT            ;<INTERRUPT ENABLE DID NOT CLEAR>
633  003166                  1$:
634  003166  032767  002000  175226    BIT     #BIT10,$SWREG   ; CHECK FOR LOOP ON TEST
635  003174  001344                    BNE     TST14           ; GO TO LOOP ON TEST
636
637  003176                  TST15:
638  003176  012767  000015  175200    MOV     #15,$TESTN      ; MOVE TEST NUMBER TO MAILBOX
639  003204  052715  000001            BIS     #1,@R5          ;SHOULD SET REQ A ALSO
640  003210  032715  000201            BIT     #201,@R5        ;VERIFY THAT REQ A IS SET
641  003214  001402                    BEQ     5$              ;FLAG ERROR MESSAGE IF NO
642  003216  005015                    CLR     @R5             ;CLEAR STATUS REGISTER
643  003220  000416                    BR      1$              ;GO TO NEXT TEST
644  003222                  5$:
645  003222  032767  040000  175172    BIT     #BIT14,$SWREG   ; CHECK FOR LOOP ON ERROR
646  003230  001362                    BNE     TST15           ; GO TO LOOP ERROR
647  003232  012767  000021  175142    MOV     #21,$FATAL
648  003240  012767  000001  175132    MOV     #1,$MSGTY       ; MOVE ERROR NUM TO MAILBOX
649  003246  005767  175150            TST     $SWREG          ; CHECK FOR HALT ON ERROR
650  003252  100401                    BMI     1$              ; CONTINUE IF SET
651  003254  000000                    HALT            ;<A REQ DID NOT SET>
652  003256                  1$:
653
```

```
654  003256                                TST16:
655  003256  012767  000016  175120              MOV     #16,$TESTN      ; MOVE TEST NUMBER TO MAILBOX
656  003264  052715  000002                      BIS     #2,@R5          ;SHOULD SET REQ B
657  003270  032715  100002                      BIT     #100002,@R5     ;VERIFY THAT REQ B IS SET
658  003274  001402                              BEQ     5$              ;FLAG ERROR MESSAGE IF NO
659  003276  005015                              CLR     @R5             ;CLEAR STATUS REGISTER
660  003300  000416                              BR      1$              ;GO TO NEXT TEST
661  003302                                5$:
662  003302  032767  040000  175112              BIT     #BIT14,$SWREG   ; CHECK FOR LOOP ON ERROR
663  003310  001362                              BNE     TST16           ; GO TO LOOP ERROR
664  003312  012767  000022  175062              MOV     #22,$FATAL
665  003320  012767  000001  175052              MOV     #1,$MSGTY       ; MOVE ERROR NUM TO MAILBOX
666  003326  005767  175070                      TST     $SWREG          ; CHECK FOR HALT ON ERROR
667  003332  100401                              BMI     1$              ; CONTINUE IF SET
668  003334  000000                              HALT            ;<B REQ DID NOT SET>
669  003336                                1$:
670
671  003336                                TST17:
672  003336  012767  000017  175040              MOV     #17,$TESTN      ; MOVE TEST NUMBER TO MAILBOX
673  003344  106427  000340                      MTPS    #340            ;LOCK OUT INTERRUPTS
674  003350  052715  177777                      BIS     #-1,@R5         ;LOAD ALL ONES IN STATUS REGISTER
675  003354  022715  100343                      CMP     #100343,(R5)    ;VERIFY THAT ALL WRITE BITS ARE SET IN DRCSR
676  003360  001416                              BEQ     T17CON          ;BR IF SET
677  003362  032767  040000  175032              BIT     #BIT14,$SWREG   ; CHECK FOR LOOP ON ERROR
678  003370  001362                              BNE     TST17           ; GO TO LOOP ERROR
679  003372  012767  000023  175002              MOV     #23,$FATAL
680  003400  012767  000001  174772              MOV     #1,$MSGTY       ; MOVE ERROR NUM TO MAILBOX
681  003406  005767  175010                      TST     $SWREG          ; CHECK FOR HALT ON ERROR
682  003412  100401                              BMI     1$              ; CONTINUE IF SET
683  003414  000000                              HALT            ;<INCORRECT DATA IN REG>
684  003416                                1$:
685  003416  042715  000003              T17CON: BIC     #3,@R5          ;CLEAR CSR BITS
686  003422  032715  000140                      BIT     #140,@R5        ;TEST INTERRUPT ENABLE BITS
687  003426  001016                              BNE     1$              ;COTINUE IF STILL SET
688  003430  032767  040000  174764              BIT     #BIT14,$SWREG   ; CHECK FOR LOOP ON ERROR
689  003436  001367                              BNE     T17CON          ; GO TO LOOP ERROR
690  003440  012767  000024  174734              MOV     #24,$FATAL
691  003446  012767  000001  174724              MOV     #1,$MSGTY       ; MOVE ERROR NUM TO MAILBOX
692  003454  005767  174742                      TST     $SWREG          ; CHECK FOR HALT ON ERROR
693  003460  100401                              BMI     1$              ; CONTINUE IF SET
694  003462  000000                              HALT            ;<WRONG BITS SET>
695  003464                                1$:
696  003464  032767  002000  174730              BIT     #BIT10,$SWREG   ; CHECK FOR LOOP ON TEST
697  003472  001321                              BNE     TST17           ; GO TO LOOP ON TEST
698
699  003474                                TST20:
700  003474  012767  000020  174702              MOV     #20,$TESTN      ; MOVE TEST NUMBER TO MAILBOX
701  003502  106427  000340                      MTPS    #340            ;LOCK OUT INTERRUPTS
702  003506  052715  000003                      BIS     #3,@R5          ;SET CSR BITS
703  003512  000005                              RESET                   ;SHOULD CLEAR INTERRUPT ENABLE FLOPS
704  003514  032715  000140                      BIT     #140,@R5        ;VERIFY THAT FLOPS ARE CLEARED
705  003520  001416                              BEQ     1$              ;BR IF YES
706  003522  032767  040000  174672              BIT     #BIT14,$SWREG   ; CHECK FOR LOOP ON ERROR
707  003530  001361                              BNE     TST20           ; GO TO LOOP ERROR
708  003532  012767  000025  174642              MOV     #25,$FATAL
709  003540  012767  000001  174632              MOV     #1,$MSGTY       ; MOVE ERROR NUM TO MAILBOX
```

```
   710  003546  005767  174650          TST     $SWREG          ; CHECK FOR HALT ON ERROR
   711  003552  100401                  BMI     1$              ; CONTINUE IF SET
   712  003554  000000                  HALT            ;<RESET DID NOT CLEAR BITS>
   713  003556                  1$:
   714  003556  032767  002000  174636  BIT     #BIT10,$SWREG   ; CHECK FOR LOOP ON TEST
   715  003564  001343                  BNE     TST20           ; GO TO LOOP ON TEST
   716
   717                          ;NOTE: THE WRAP CABLE MUST BE INSTALLED TO EXECUTE
   718                          ;      TESTS 21-27
   719  003566                  TST21:
   720  003566  012767  000021  174610  MOV     #21,$TESTN      ; MOVE TEST NUMBER TO MAILBOX
   721  003574  032767  000400  174620  BIT     #BIT8,$SWREG
   722  003602  001402                  BEQ     LP21            ;DO TESTS IF NOT INHIBITED
   723  003604  000167  000710          JMP     TST999          ;IF INHIBITED
   724  003610  005015                  LP21:   CLR     @R5             ;CLEAR STATUS REGISTER
   725  003612  005215                  INC     @R5             ;SET CSR0
   726  003614  105715                  TSTB    @R5             ;CHECK FOR REQ A FLAG
   727  003616  100416                  BMI     1$              ;BR IF SET
   728  003620  032767  040000  174574  BIT     #BIT14,$SWREG   ; CHECK FOR LOOP ON ERROR
   729  003626  001357                  BNE     TST21           ; GO TO LOOP ERROR
   730  003630  012767  000026  174544  MOV     #26,$FATAL
   731  003636  012767  000001  174534  MOV     #1,$MSGTY       ; MOVE ERRCR NUM TO MAILBOX
   732  003644  005767  174552          TST     $SWREG          ; CHECK FOR HALT ON ERROR
   733  003650  100401                  BMI     1$              ; CONTINUE IF SET
   734  003652  000000                  HALT            ;<BIT0 DID NOT SET BIT7>
   735  003654                  1$:
   736  003654  032767  002000  174540  BIT     #BIT10,$SWREG   ; CHECK FOR LOOP ON TEST
   737  003662  001341                  BNE     TST21           ; GO TO LOOP ON TEST
   738
```

```
                                        TST22:
003664                                          MOV     #22,$TESTN      ; MOVE TEST NUMBER TO MAILBOX
003664  012767  000022  174512                  MOV     #2,@R5          ;SET CSR1
003672  012715  000002                          TST     @R5             ;CHECK FOR REQ B FLAG
003676  005715                                  BMI     1$              ;BR IF SET
003700  100416                                  BIT     #BIT14,$SWREG   ; CHECK FOR LOOP ON ERROR
003702  032767  040000  174512                  BNE     TST22           ; GO TO LOOP ERROR
003710  001365                                  MOV     #27,$FATAL
003712  012767  000027  174462                  MOV     #1,$MSGTY       ; MOVE ERROR NUM TO MAILBOX
003720  012767  000001  174452                  TST     $SWREG          ; CHECK FOR HALT ON ERROR
003726  005767  174470                          BMI     1$              ; CONTINUE IF SET
003732  100401                                  HALT            ;<BIT1 DID NOT SET BIT15>
003734  000000                          1$:
003736                                  BIT     #BIT10,$SWREG   ; CHECK FOR LOOP ON TEST
003736  032767  002000  174456                  BNE     TST22           ; GO TO LOOP ON TEST
003744  001347                          TST23:
003746                                  MOV     #23,$TESTN      ; MOVE TEST NUMBER TO MAILBOX
003746  012767  000023  174430                  MOV     #3,@R5          ;CSR0 AND CSR1
003754  012715  000003                          MOV     (R5),R0         ;COPY DATA
003760  011500                                  CMP     #100203,R0      ;COMPARE DATA
003762  022700  100203                          BEQ     1$              ;BR IF GOOD DATA IS RECEIVED
003766  001416                                  BIT     #BIT14,$SWREG   ; CHECK FOR LOOP ON ERROR
003770  032767  040000  174424                  BNE     TST23           ; GO TO LOOP ERROR
003776  001363                                  MOV     #30,$FATAL
004000  012767  000030  174374                  MOV     #1,$MSGTY       ; MOVE ERROR NUM TO MAILBOX
004006  012767  000001  174364                  TST     $SWREG          ; CHECK FOR HALT ON ERROR
004014  005767  174402                          BMI     1$              ; CONTINUE IF SET
004020  100401                                  HALT            ;<INCORRECT BITS SET IN REG>
004022  000000                          1$:
004024                                  BIT     #BIT10,$SWREG   ; CHECK FOR LOOP ON TEST
004024  032767  002000  174370                  BNE     TST23           ; GO TO LOOP ON TEST
004032  001345
                                        ;CAN WE RAISE INTERRUPT "A"
                                        TST24:
004034                                  MOV     #24,$TESTN      ; MOVE TEST NUMBER TO MAILBOX
004034  012767  000024  174342                  MTPS    #340            ;LOCK OUT INTERRUPTS
004042  106427  000340                          MOV     #STKPTR,SP      ;INITIALIZE STACK POINTER
004046  012706  001200                          MOV     #4$,@DRVECA     ;INTERRUPT RETURN POINTER
004052  012777  004074  175150                  MOV     #101,@R5        ;INTERRUPT ENABLE AND CSR0
004060  012715  000101                          MTPS    #0              ;ALLOW INTERRUPTS
004064  106427  000000                          NOP
004070  000240                                  BR      5$              ;NO INTERRUPT
004072  000402
                                        4$:     CLR     @R5             ; CLEAR INTERRUPT ENABLE
004074  005015                                  BR      1$              ;GO TO NEXT TEST
004076  000416                          5$:
004100                                  BIT     #BIT14,$SWREG   ; CHECK FOR LOOP ON ERROR
004100  032767  040000  174314                  BNE     TST24           ; GO TO LOOP ERROR
004106  001352                                  MOV     #31,$FATAL
004110  012767  000031  174264                  MOV     #1,$MSGTY       ; MOVE ERROR NUM TO MAILBOX
004116  012767  000001  174254                  TST     $SWREG          ; CHECK FOR HALT ON ERROR
004124  005767  174272                          BMI     1$              ; CONTINUE IF SET
004130  100401                                  HALT            ;<NO A INTERRUPT>
004132  000000                          1$:
004134                                  BIT     #BIT10,$SWREG   ; CHECK FOR LOOP ON TEST
004134  032767  002000  174260                  BNE     TST24           ; GO TO LOOP ON TEST
004142  001334
```

```
795
796                                     ;RAISE INTERRUPT 'B'
797  004144                             TST25:
798  004144  012767  000025  174232             MOV     #25,$TESTN      ; MOVE TEST NUMBER TO MAILBOX
799  004152  012706  001200                     MOV     #STKPTR,SP      ;INITIALIZE STACK POINTER
800  004156  106427  000340                     MTPS    #340            ;LOCK OUT INTERRUPTS
801  004162  012777  004204  175044             MOV     #4$,@DRVECB     ;INTERRUPT RETURN POINTER
802  004170  012715  000042                     MOV     #42,@R5         ;IE AND CSR1
803  004174  106427  000000                     MTPS    #0              ;ALLOW INTERRUPTS
804  004200  000240                             NOP
805  004202  000402                             BR      5$              ;NO INTERRUPT
806  004204  005015                     4$:     CLR     @R5             ;CLEAR INTERRUPT ENABLE
807  004206  000416                             BR      1$              ;GO TO NEXT TEST
808  004210                             5$:
809  004210  032767  040000  174204             BIT     #BIT14,$SWREG   ; CHECK FOR LOOP ON ERROR
810  004216  001352                             BNE     TST25           ; GO TO LOOP ERROR
811  004220  012767  000032  174154             MOV     #32,$FATAL
812  004226  012767  000001  174144             MOV     #1,$MSGTY       ; MOVE ERROR NUM TO MAILBOX
813  004234  005767  174162                     TST     $SWREG          ; CHECK FOR HALT ON ERROR
814  004240  100401                             BMI     1$              ; CONTINUE IF SET
815  004242  000000                             HALT            ;<NO B INTERRUPT>
816  004244                             1$:
817  004244  032767  002000  174150             BIT     #BIT10,$SWREG   ; CHECK FOR LOOP ON TEST
818  004252  001334                             BNE     TST25           ; GO TO LOOP ON TEST
819
820                                     ;TEST FOR INTERRUPT FROM DEVICE
821  004254                             TST26:
822  004254  012767  000026  174122             MOV     #26,$TESTN      ; MOVE TEST NUMBER TO MAILBOX
823  004262  017702  174744                     MOV     @DRLVLA,R2      ;SAVE INTERRUPT PSW
824  004266  016777  174752  174736     LP26:   MOV     PL,@DRLVLA      ;LOCK OUT SUCCESSIVE INTERRUPTS
825  004274  012706  001200                     MOV     #STKPTR,SP      ;INITIALIZE STACK POINTER
826  004300  012777  004354  174722             MOV     #1$,@DRVECA     ;INTERRUPT RETURN POINTER
827  004306  012715  000101                     MOV     #101,@R5        ;SET INTERRUPT ENABLE-AND CSR0
828  004312  106427  000000                     MTPS    #0              ;ALLOW INTERRUPTS
829  004316  000240                             NOP
830  004320                             5$:
831  004320  032767  040000  174074             BIT     #BIT14,$SWREG   ; CHECK FOR LOOP ON ERROR
832  004326  001352                             BNE     TST26           ; GO TO LOOP ERROR
833  004330  012767  000033  174044             MOV     #33,$FATAL
834  004336  012767  000001  174034             MOV     #1,$MSGTY       ; MOVE ERROR NUM TO MAILBOX
835  004344  005767  174052                     TST     $SWREG          ; CHECK FOR HALT ON ERROR
836  004350  100401                             BMI     1$              ; CONTINUE IF SET
837  004352  000000                             HALT            ;<NO DEVICE INTERRUPT>
838  004354                             1$:
839  004354  032767  002000  174040             BIT     #BIT10,$SWREG   ; CHECK FOR LOOP ON TEST
840  004362  001341                             BNE     LP26            ; GO TO LOOP ON TEST
841  004364  005015                             CLR     @R5             ;CLEAR INTERRUPT ENABLE
842  004366  010277  174640                     MOV     R2,@DRLVLA      ;RESTORE INTERRUPT PSW
843                                     ; PLU WRAP TEST
844  004372                             TST27:
845  004372  012767  000027  174004             MOV     #27,$TESTN      ; MOVE TEST NUMBER TO MAILBOX
846  004400  005000                             CLR     R0              ;SET UP STARTING DATA
847  004402  010077  174614             WLOOP:  MOV     R0,@DROBUF      ; SEND DATA
848  004406  027700  174612                     CMP     @DRIBUF,R0      ; CHECK THE DATA
849  004412  001020                             BNE     5$              ; ERROR IF NOT RIGHT
850  004414  005200                             INC     R0              ; CHANGE DATA
```

```
851  004416  001434                          BEQ   1$                ;NEXT TEST IF END
852  004420  022700  031460          3$:     CMP   #31460,R0         ; CHECK FOR TEST MODULE CODE
853  004424  001411                          BEQ   4$
854  004426  022700  031461                  CMP   #31461,R0
855  004432  001406                          BEQ   4$
856  004434  022700  031462                  CMP   #31462,R0
857  004440  001403                          BEQ   4$
858  004442  022700  031463                  CMP   #31463,R0
859  004446  001355                          BNE   WLOOP
860  004450  005200                  4$:     INC   R0
861  004452  000762                          BR    3$                ; RECHECK DATA CODE
862  004454                          5$:
863  004454  032767  040000  173740          BIT   #BIT14,$SWREG     ; CHECK FOR LOOP ON ERROR
864  004462  001347                          BNE   WLOOP             ; GO TO LOOP ERROR
865  004464  012767  000034  173710          MOV   #34,$FATAL
866  004472  012767  000001  173700          MOV   #1,$MSGTY         ; MOVE ERROR NUM TO MAILBOX
867  004500  005767  173716                  TST   $SWREG            ; CHECK FOR HALT ON ERROR
868  004504  100401                          BMI   1$                ; CONTINUE IF SET
869  004506  000000                          HALT          ;<WRAP DATA DID NOT COMPARE>
870  004510                          1$:
871  004510  032767  002000  173704          BIT   #BIT10,$SWREG     ; CHECK FOR LOOP ON TEST
872  004516  001325                          BNE   TST27             ; GO TO LOOP ON TEST
```

```
873
874
875  004520                                  TST999:
876  004520  005237  000406                          INC     @#$PASS         ; INCREMENT PASS COUNT
877  004524  132767  000040  173667                  BITB    #40,$ENVM       ; WILL APT ALLOW PRINTING?
878  004532  001010                                  BNE     ACT             ; NO
879  004534  012700  004640                          MOV     #MSG,R0         ; GET MESSAGE ADDRESS
880  004540  105737  177564          WAIT:   TSTB    @#TPS           ; CHECK IF TTY READY
881  004544  100375                          BPL     WAIT            ; IF NOT
882  004546  112037  177566                  MOVB    (R0)+,@#TPB     ; PRINT THE CHARACTER
883  004552  001372                          BNE     WAIT            ; NEXT IF NOT DONE
884  004554  013700  000042          ACT:    MOV     @#42,R0         ; CHECK ACT
885  004560  001405                          BEQ     GOAGIN          ; KEEP GOING
886  004562  000005                          RESET
887  004564  004710                  $ENDAD: JSR     PC,(R0)         ; ACT HOOKS
888  004566  000240                          NOP
889  004570  000240                          NOP
890  004572  000240                          NOP
891  004574  000167  174506          GOAGIN: JMP     AROUND ;+ DO ANOTHER PASS
892                                          ;+ ON PASSES >1 THE TITLE PRINTOUT WILL BE SUPPRESSED
893                                  .EVEN
894  004600                          STORE:
895          177564                  TPS=177564
896          177566                  TPB=177566
897  004600  177777                  PASSPT: -1
898  004602  053103  040513  026506  TITLE1:.ASCIZ .CVKAF-DO   DRV11 DIAGNOSTIC. <15><12>
899  004610  030104  020040  042040
900  004616  053122  030461  042040
901  004624  040511  047107  051517
902  004632  044524  006503  000012
903  004640  047105  020104  043117  MSG:    .ASCIZ  .END OF PASS.<15><12>
904  004646  050040  051501  006523
905  004654  000012
906                                  .EVEN
```

```
907
908                                     ;ENTER HERE FOR POWER FAIL
909
910  004656  010046                    PFAIL:  MOV     %0,-(6)         ;SAVE REGISTER OR STACK
911  004660  010146                            MOV     %1,-(6)         ;WHEN POWERING DOWN
912  004662  010246                            MOV     %2,-(6)
913  004664  010346                            MOV     %3,-(6)
914  004666  010446                            MOV     %4,-(6)
915  004670  010546                            MOV     %5,-(6)
916  004672  016746  173126                    MOV     24,-(6)
917  004676  010637  004712                    MOV     %6,@#SAVR6      ;STORE STACK POSITION
918  004702  012737  004714  000024            MOV     #RESTAR,@#24
919  004710  000000                            HALT                    ;HALT ON POWER DOWN NORMAL
920  004712  000000                    SAVR6:  0                       ;STACK IS SAVED HERE
921  004714  016706  177772            RESTAR:MOV      SAVR6,%6        ;RESTORE REGISTER OFF STACK
922  004720  012667  173100                    MOV     (6)+,24         ;WHEN POWERING UP
923  004724  012605                            MOV     (6)+,%5
924  004726  012604                            MOV     (6)+,%4
925  004730  012603                            MOV     (6)+,%3
926  004732  012602                            MOV     (6)+,%2
927  004734  012601                            MOV     (6)+,%1
928  004736  012600                            MOV     (6)+,%0
929  004740  000137  001256                    JMP     @#START
930          000001                            .END
```

```
ABASE = 000000             234
ACDW1 = 000000             234
ACDW2 = 000000             234
ACPUOP= 000000             234      249
ACT     004554             878      884#
ADDW0 = 000000             234
ADDW1 = 000000             234
ADDW10= 000000             234
ADDW11= 000000             234
ADDW12= 000000             234
ADDW13= 000000             234
ADDW14= 000000             234
ADDW15= 000000             234
ADDW2 = 000000             234
ADDW3 = 000000             234
ADDW4 = 000000             234
ADDW5 = 000000             234
ADDW6 = 000000             234
ADDW7 = 000000             234
ADDW8 = 000000             234
ADDW9 = 000000             234
ADEVCT= 000000             234      240
ADEVM = 000000             234
AENV  = 000000             234      245
AENVM = 000000             234      246
AFATAL= 000000             234      237
AMADR1= 000000             234
AMADR2= 000000             234
AMADR3= 000000             234
AMADR4= 000000             234
AMAMS1= 000000             234
AMAMS2= 000000             234
AMAMS3= 000000             234
AMAMS4= 000000             234
AMSGAD= 000000             234      242
AMSGLG= 000000             234      243
AMSGTY= 000000             234      236
AMTYP1= 000000             234
AMTYP2= 000000             234
AMTYP3= 000000             234
AMTYP4= 000000             234
APASS = 000000             234      239
APRIOR= 000000             234
AROUND  001306             322      332#     891
ASWREG= 000000             234      247
ATESTN= 000000             234      238
AUNIT = 000000             234      241
AUSWR = 000000             234      248
AVECT1= 000000             234
AVECT2= 000000             234
BIT0  = 000001             144#     395
BIT00 = 000001             134#     144
BIT01 = 000002             133#     143
BIT02 = 000004             132#     142
BIT03 = 000010             131#     141
BIT04 = 000020             130#     140
```

```
BIT05 = 000040           129#    139
BIT06 = 000100           128#    138
BIT07 = 000200           127#    137
BIT08 = 000400           126#    136
BIT09 = 001000           125#    135
BIT1  = 000002           143#    405
BIT10 = 002000           124#    372     429     445     465     482     499     524     543     561     585     616     634
                         696     714     736     752     768     793     817     839     871
BIT11 = 004000           123#
BIT12 = 010000           122#
BIT13 = 020000           121#
BIT14 = 040000           120#    354     364     397     407     421     437     457     474     491     516     535     553
                         577     596     608     626     645     662     677     688     706     728     744     760     785
                         809     831     863
BIT15 = 100000           119#
BIT2  = 000004           142#
BIT3  = 000010           141#
BIT4  = 000020           140#
BIT5  = 000040           139#
BIT6  = 000100           138#
BIT7  = 000200           137#
BIT8  = 000400           136#    451     721
BIT9  = 001000           135#    385
BPTVEC= 000014           151#
BUFTST  002342           505#
CON     001524           353     362     378#
COUNT   001242           309#
CR    = 000015            59#
CRLF  = 000200            60#
CSR   = 167770            29#    287     288     289     290
DDISP = 177570            66#
DEVCNT= 000410           281#
DRBHIO  001226           301#    549*    568*    569*
DRCSR   001220           298#    333     347
DRIBUF  001224           300#    394     418     455     848
DRLVLA  001232           304#    823     824*    842*
DRLVLB  001236           306#
DROBUF  001222           299#    350*    352     391*    433*    434     453*    470*    471     487*    488     530*    531*
                         532     548*    550     567*    571     847*
DRVECA  001230           303#    776*    826*
DRVECB  001234           305#    801*
DSWR  = 177570            65#
EMTVEC= 000030           154#
ERRNUM= 000402           282#
ERRVEC= 000004           147#
ERR1    001442           349     363#
GOAGIN  004574           885     891#
HLT   = 104000            28#
HT    = 000011            57#
IOTVEC= 000020           152#
LF    = 000012            58#
LOOP1   001272           324#    325     327
LP1     001350           347#    355     365
LP21    003610           722     724#
LP26    004266           824#    840
LP7     002350           508#    514     517
```

```
MSG      004640              879     903#
N      = 000035               10#    356     357#    366     367#    399     400#    409     410#    423     424#    439     440#
                              459     460#    476     477#    493     494#    518     519#    537     538#    555     556#    579
                              580#    598     599#    610     611#    628     629#    647     648#    664     665#    679     680#
                              690     691#    708     709#    730     731#    746     747#    762     763#    787     788#    811
                              812#    833     834#    865     866#
OPTION=  000426              283#
PASSPT   004600              897#
PFAIL    004656              335     910#
PIRQ   = 177772               64#
PIRQVE=  000240              158#
PL       001244              310#    824
PR0    = 000000               81#
PR1    = 000040               82#
PR2    = 000100               83#
PR3    = 000140               84#
PR4    = 000200               85#
PR5    = 000240               86#
PR6    = 000300               87#
PR7    = 000340               88#
PS     = 177776               61#     62
PSW    = 177776               62#
PWRVEC=  000024              153#
RCSR     001200              287#    332
RCSR1    001210              291#
RESTAR   004714              918     921#
RESVEC=  000010              148#
SAVR6    004712              917*    920#    921
STACK  = 001100               52#
START    001256              313     321#    929
START1   001246              224     312#
STKLMT=  177774               63#
STKPTR=  001200               30#    312     375     387     775     799     825
STORE    004600              894#
SW0    = 000001              116#
SW00   = 000001              106#    116
SW01   = 000002              105#    115
SW02   = 000004              104#    114
SW03   = 000010              103#    113
SW04   = 000020              102#    112
SW05   = 000040              101#    111
SW06   = 000100              100#    110
SW07   = 000200               99#    109
SW08   = 000400               98#    108
SW09   = 001000               97#    107
SW1    = 000002              115#
SW10   = 002000               44#     96#
SW11   = 004000               45#     95#
SW12   = 010000               94#
SW13   = 020000               46#     93#
SW14   = 040000               47#     92#
SW15   = 100000               91#
SW2    = 000004              114#
SW3    = 000010              113#
SW4    = 000020              112#
SW5    = 000040              111#
```

```
SW6    = 000100             110#
SW7    = 000200             109#
SW8    = 000400             108#
SW9    = 001000              43#     107#
TAG      002440             530#     536
TBITVE= 000014              149#
TITLE1   004602             323      898#
TKVEC  = 000060             156#
TPB    = 177566             326*     882*     896#
TPS    = 177564             324      880      895#
TPVEC  = 000064             157#
TRAPVE= 000034              155#
TRTVEC= 000014              150#
TST1     001342             345#     373
TST10    002432             528#     544
TST11    002530             546#     554      562
TST12    002626             563#     578      586
TST13    002752             587      590#     597      617
TST14    003106             619#     627      635
TST15    003176             637#     646
TST16    003256             654#     663
TST17    003336             671#     678      697
TST2     001526             374      383#     398      430
TST20    003474             699#     707      715
TST21    003566             719#     729      737
TST22    003664             739#     745      753
TST23    003746             754#     761      769
TST24    004034             772#     786      794
TST25    004144             797#     810      818
TST26    004254             821#     832
TST27    004372             844#     872
TST3     001756             386      431#     438      446
TST4     002050             449#     458      466
TST5     002150             452      468#     475      483
TST6     002242             485#     492      500
TST7     002334             503#     525
TST999   004520             723      875#
T12CON   002650             568#
T12DAT   002750             565*     566      588#
T13CON   003026             595      604#     609
T17CON   003416             676      685#     689
T2CN1    001674             406      415#     422
T2CON    001632             396      405#     408
WAIT     004540             880#     881      883
WLOOP    004402             847#     859      864
X      = 000030               9#     346      347#     384      385#     432      433#     450      451#     469      470#     486      487#
                           504      505#     529      530#     547      548#     564      565#     591      592#     620      621#     638
                           639#     655      656#     672      673#     700      701#     720      721#     740      741#     755      756#
                           773      774#     798      799#     822      823#     845      846#
XORFLG   001240             307#
$APTHD   000430             267      273#
$CPUOP   000426             249#     283
$DEVCT   000410             240#     281
$ENDAD   004564             887#
$ENV     000420             245#
$ENVM    000421             246#     321      877
```

```
$ETABL  000420          244#
$ETEND  000430          256#  279
$FATAL  000402          222*  237#  282   356*  366*  399*  409*  423*  439*  459*  476*  493*  518*
                        537*  555*  579*  598*  610*  628*  647*  664*  679*  690*  708*  730*  746*
                        762*  787*  811*  833*  865*
$HIBTS  000430          274#
$MAIL   000400          235#  275   279
$MBADR  000432          275#
$MSGAD  000414          242#
$MSGLG  000416          243#
$MSGTY  000400          236#  357*  367*  400*  410*  424*  440*  460*  477*  494*  519*  538*  556*
                        580*  599*  611*  629*  648*  665*  680*  691*  709*  731*  747*  763*  788*
                        812*  834*  866*
$PASS   000406          221*  239#  876*
$PASTM  000436          277#
$SWREG  000422          247#  354   358   364   368   372   385   397   401   407   411   421   425
                        429   437   441   445   451   457   461   465   474   478   482   491   495
                        499   516   520   524   535   539   543   553   557   561   577   581   585
                        596   600   608   612   616   626   630   634   645   649   662   666   677
                        681   688   692   696   706   710   714   721   728   732   736   744   748
                        752   760   764   768   785   789   793   809   813   817   831   835   839
                        863   867   871
$TESTN  000404          223*  238#  346*  384*  432*  450*  469*  486*  504*  529*  547*  564*  591*
                        620*  638*  655*  672*  700*  720*  740*  755*  773*  798*  822*  845*
$TSTM   000434          276#
$UNIT   000412          241#
$UNITM  000440          278#
$USWR   000424          248#
.       = 004744        160#  161   163   165   167   169   171   173   175   177   179   181   183
                        185   187   189#  192   194   196   198   200   202   204   206   208   210
                        212   214   216   218   220#  225#  226   228   230#  263   264#  266#  268#
                        280#  378
.$X     = 000430        263#  268
```

```
COMMEN      1#      159#
ENDCOM      1#      159#
ERNUM      12#      356     366     399     409     423     439     459     476     493     518     537     555     579     598
          610       628     647     664     679     690     708     730     746     762     787     811     833     865
ERR        12#      354     363     397     407     421     437     457     474     491     515     535     553     576     596
          608       626     644     661     677     688     706     728     744     760     784     808     830     862
ERROR      53#
ESCAPE      1#      159#
GETPRI      1#      159#
GETSWR      1#      159#
MULT        1#      159#
NEWTST      1#      159#
POP         1#      159#
PUSH        1#      159#
REPORT      1#      159#
SCOPE      54#
SEQ        13#      345     383     431     449     468     485     503     528     546     563     590     619     637     654
          671       699     719     739     754     772     797     821     844
SETPRI      1#      159#
SETUP       1#      159#
SKIP        1#      159#
SLASH       1#      159#
SPACE     159#
STARS       1#      159#    233     260     262     269
SWRSU       1#      159#
TYPBIN      1#      159#
TYPDEC      1#      159#
TYPNAM      1#      159#
TYPNUM      1#      159#
TYPOCS      1#      159#
TYPOCT      1#      159#
TYPTXT      1#      159#
$TNUM      13#      346     384     432     450     469     486     504     529     547     564     591     620     638     655
          672       700     720     740     755     773     798     822     845
$$ESCA      1#      159#
$$NEWT      1#      159#
$$SKIP      1#      159#
.EQUAT      1#       49#
.HEADE      1#
.KT11       1#
.SETUP      1#
.SWRHI      1#
.$ACT1      1#
.$APTB      1#       49#    231
.$APTH      1#       49#    258
.$APTY      1#       49#
.$ASTA      1#
.$CATC      1#
.$CMTA      1#
.$DB2D      1#
.$DB2O      1#
.$DIV       1#
.$EOP       1#
.$ERRC      1#
.$ERRT      1#
.$MULT      1#
```

```
.$POWE      1#
.$RAND      1#
.$RDDE      1#
.$RDOC      1#
.$READ      1#
.$R2AZ      1#
.$SAVE      1#
.$SB2D      1#
.$SB2O      1#
.$SCOP      1#
.$SIZE      1#
.$SUPR      1#
.$TRAP      1#      49#
.$TYPB      1#
.$TYPD      1#
.$TYPE      1#      49#
.$TYPO      1#
.$40CA      1#
.1170       1#


. ABS.  004744      000


 ERRORS DETECTED:  0

 CVKAFD.BIN,CVKAFD.LST/CRF/SOL/NL:TOC=CVKAFD.SML,CVKAFD.P11
 RUN-TIME:  8  9  .5 SECONDS
 RUN-TIME RATIO: 75/18=4.1
 CORE USED:  32K  (63 PAGES)
```