

DRV11

TEST PRO ENH
CNKAFAO

AH-T468A-MC
FICHE 1 OF 1

MAY 1983
COPYRIGHT © 82-83
MADE IN USA



Microfiche grid containing multiple frames of data, including text and tables.



5675
5676
5677
5678
5679
5680
5681
5682
5683
5684
5685
5686
5687
5688
5689
5690
5691
5692
5693
5694
5695
5696
5697
5698
5699
5700
5701
5702
5703
5704
5705
5706
5707
5708
5709
5710
5711
5712
5713
5714
5715
5716
5717
5718
5719
5720
5721
5722
5723
5724
5725
5726
5727

.REM 8

IDENTIFICATION

PRODUCT CODE: AC-T467A-MC
PRODUCT NAME: CNKafAO DRV11 TEST PRO ENH
PRODUCT DATE: DECEMBER, 1982
MAINTAINER : DIAGNOSTIC SERVICES/ISS
AUTHOR: ELLIOT GERBERG

COPYRIGHT (C) 1982,1983 BY
DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS
ALL RIGHTS RESERVED

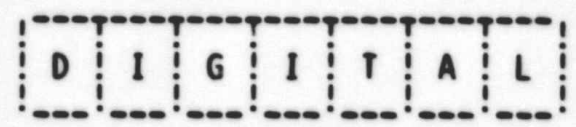
THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY
BE USED AND COPIED ONLY IN ACCORDANCE WITH THE TERMS
OF SUCH LICENSE AND WITH THE INCLUSION OF THE ABOVE
COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER COPIES
THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAIL-
ABLE TO ANY OTHER PERSON. NO TITLE TO AND OWNERSHIP
OF THE SOFTWARE IS HEREBY TRANSFERRED.

THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE
WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COM-
MITMENT BY DIGITAL EQUIPMENT CORPORATION.

DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR
RELIABILITY OF ITS SOFTWARE ON EQUIPMENT THAT IS
NOT SUPPLIED BY DIGITAL.

THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION:

DEC	PDP	UNIBUS	MASSBUS
DECUS	DECTAPE	VAX	



5729
5730
5731
5732
5733
5734
5735
5736
5737
5738
5739
5740
5741
5742
5743
5744
5745
5746
5747
5748
5749
5750
5751
5752
5753
5754
5755
5756
5757
5758
5759
5760
5761
5762
5763
5764
5765
5766
5767
5768
5769
5770
5771
5772
5773
5774
5775
5776
5777
5778
5779
5780
5781
5782
5783
5784

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

SBC 11/21

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

UNDER XXDP+ OR
ABSOLUTE LOADER

4. STARTING PROCEDURE

200 - NORMAL ENTRY TO TEST ONE DEVICE

TO LOAD AND EXECUTE

1. UNDER XXDP+ OR 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)

5785
5786
5787
5788
5789
5790
5791
5792
5793
5794
5795
5796
5797
5798
5799
5800
5801
5802
5803
5804
5805
5806
5807
5808
5809
5810
5811
5812
5813
5814
5815
5816
5817
5818
5819
5820
5821
5822
5823
5824
5825
5826
5827
5828
5829
5830
5831
5832
5833
5834
5835
5836
5837
5838
5839
5840

3. START PROGRAM AT 200.
4. PROGRAM WILL PRINT "END OF PASS" FOLLOWING EACH PASS.

4.1 CONTRL 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 RESETING 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

5841
5842
5843
5844
5845
5846
5847
5848
5849
5850
5851
5852
5853
5854
5855
5856
5857
5858
5859
5860
5861
5862
5863
5864
5865
5866
5867
5868
5869
5870
5871
5872
5873
5874
5875
5876
5877
5878
5879
5880
5881
5882
5883
5884
5885
5886
5887
5888
5889
5890
5891
5892
5893
5894
5895
5896

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 174500 OR VECTORED AT 340, 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

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 DESCRIPTION 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.

5897
5898
5899
5900
5901
5902
5903
5904
5905
5906
5907
5908
5909
5910
5911
5912
5913
5914
5915
5916
5917
5918
5919
5920
5921
5922
5923
5924
5925
5926
5927
5928
5929
5930
5931
5932
5933
5961

7. 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

8. LISTING

.TITLE CNKafa
.ENABLE ABS
.NLIST MD,MC,CND
.LIST ME

000001
000001

X=1
N=1

5972
5973
5974
5975
5976
5977
5978
5979
5980
5981
5982
5983
5984
5985
5986
5987
5988
5989
5990
5991
5992
5993
5994
5995
5996
5997
5998
5999
6000
6001
6002
6003
6004
6005
6006

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

:REVISION 1. C
: 1. DOCUMENTATION CHANGE ONLY

:REVISION 1. D
: 1. ADDED CODE TO PRINT OUT PROGRAMS TITLE

:REVISION 1. E
: 1. INCORPORATED PATCHES D1,D2 AND D3
D1 -- "INCORRECT PROGRAM COUNTER ADDRESSING AT
LOC. 1360 IN TEST 1, CAUSED WRONG ADDRESS
TO BE LOADED INTO LOC. 4. THIS RESULTS
IN PROGRAM HALTING AND THE FOLLOWING
MESSAGE TO BE DISPLAYED TO CONSOLE:
'ILLEGAL WORD 1'"
D2 -- "PROGRAM OVERPRINTS END PASS REPORT AND
PROGRAM TITLE."
D3 -- "RELEASED VERSION OF PATCH D2 ON THE MEDIA
IS INCORRECT CAUSING CPU TO HALT AT 314"
2. MADE PROGRAM APT COMPATIBLE

: CVKAFE DIAGNOSTIC WAS MODIFIED TO RUN ON 11/21 PROCESSOR
: BY LOWERING PRIORITY 7 TO 6 AND INITIALIALIZING BRKVEC
: AND LKVEC AND ALSO CHANGING DEFAULT CSR AND VECTOR
: ADDRESSES. THE PROGRAM WAS RENAMED TO CNKAFAD.

```
6008          ;GENERAL REGISTER LOGIC TEST
6009
6010          104000      HLT=104000
6011          174500      CSR=174500
6012          001200      STKPTR=1200
6013          ;REGISTER DEFINITIONS
6014          000000      R0=%0
6015          000001      R1=%1
6016          000002      R2=%2
6017          000003      R3=%3
6018          000004      R4=%4
6019          000005      R5=%5
6020          000006      SP=%6
6021          000007      PC=%7
6022
6023          ;SWITCHES
6024
6025          001000      SW9=1000
6026          002000      SW10=2000
6027          004000      SW11=4000
6028          020000      SW13=20000
6029          040000      SW14=40000
6030
6032          .SBTTL BASIC DEFINITIONS
(1)          ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
(1)          001100      STACK= 1100
(1)          .EQUIV EMT,ERROR      ;;BASIC DEFINITION OF ERROR CALL
(1)          .EQUIV IOT,SCOPE      ;;BASIC DEFINITION OF SCOPE CALL
(1)
(1)          ;*MISCELLANEOUS DEFINITIONS
(1)          000011      HT= 11      ;;CODE FOR HORIZONTAL TAB
(1)          000012      LF= 12      ;;CODE FOR LINE FEED
(1)          000015      CR= 15      ;;CODE FOR CARRIAGE RETURN
(1)          000200      CRLF= 200    ;;CODE FOR CARRIAGE RETURN-LINE FEED
(1)          177776      PS= 177776  ;;PROCESSOR STATUS WORD
(1)          .EQUIV PS,PSW
(1)          177774      STKLMT= 177774 ;;STACK LIMIT REGISTER
(1)          177772      PIRQ= 177772 ;;PROGRAM INTERRUPT REQUEST REGISTER
(1)          177570      DSWR= 177570 ;;HARDWARE SWITCH REGISTER
(1)          177570      DDISP= 177570 ;;HARDWARE DISPLAY REGISTER
(1)          ;***** THE FOLLOWING ODT START ADDRESS FOR SBC 11/21 IS ADDED
(1)          170000      ODTST= 170000
(1)          ;*GENERAL PURPOSE REGISTER DEFINITIONS
(1)          000000      R0= %0      ;;GENERAL REGISTER
(1)          000001      R1= %1      ;;GENERAL REGISTER
(1)          000002      R2= %2      ;;GENERAL REGISTER
(1)          000003      R3= %3      ;;GENERAL REGISTER
(1)          000004      R4= %4      ;;GENERAL REGISTER
(1)          000005      R5= %5      ;;GENERAL REGISTER
(1)          000006      R6= %6      ;;GENERAL REGISTER
(1)          000007      R7= %7      ;;GENERAL REGISTER
(1)          000006      SP= %6      ;;STACK POINTER
(1)          000007      PC= %7      ;;PROGRAM COUNTER
(1)
(1)          ;*PRIORITY LEVEL DEFINITIONS
```


(1)	000000	PR0=	0	::PRIORITY LEVEL 0
(1)	000040	PR1=	40	::PRIORITY LEVEL 1
(1)	000100	PR2=	100	::PRIORITY LEVEL 2
(1)	000140	PR3=	140	::PRIORITY LEVEL 3
(1)	000200	PR4=	200	::PRIORITY LEVEL 4
(1)	000240	PR5=	240	::PRIORITY LEVEL 5
(1)	000300	PR6=	300	::PRIORITY LEVEL 6
(1)	000340	PR7=	340	::PRIORITY LEVEL 7

::*"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

::*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) ;*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) ;***** THE FOLLOWING BREAK VECTOR AND LINE CLOCK VECTOR ARE INCLUDED
(1) 000100 LKVEC= 100 ;: LINE CLOCK VECTOR
(1) 000140 BRKVEC= 140 ;: BREAK VECTOR
(1) 000240 PIRQVEC=240 ;: PROGRAM INTERRUPT REQUEST VECTOR

6033 .ENABLE ABS
6034 .=0
6038 000000 .+2
(1) 000002 000000 HALT
(1) 000004 000006 .+2
(1) 000006 000000 HALT
(1) 000010 000012 .+2
(1) 000012 000000 HALT
(1) 000014 000016 .+2
(1) 000016 000000 HALT
(1) 000020 000022 .+2
(1) 000022 000000 HALT
(1) 000024 000026 .+2
(1) 000026 000000 HALT
(1) 000030 000032 .+2
(1) 000032 000000 HALT
(1) 000034 000036 .+2
(1) 000036 000000 HALT
(1) 000040 000042 .+2
(1) 000042 000000 HALT
(1) 000044 000046 .+2
(1) 000046 000000 HALT
(1) 000050 000052 .+2
(1) 000052 000000 HALT
(1) 000054 000056 .+2
(1) 000056 000000 HALT
(1) 000060 000062 .+2
(1) 000062 000000 HALT
(1) 000064 000066 .+2
(1) 000066 000000 HALT
6039 000100 .=100

```

```

6040 000100 000102 102
6041 000102 000002 RTI ; RTI FOR POSSIBLE CLOCK INTERRUPT
6045 000104 000106 .+2
(1) 000106 000000 HALT
(1) 000110 000112 .+2
(1) 000112 000000 HALT
(1) 000114 000116 .+2
(1) 000116 000000 HALT
(1) 000120 000122 .+2
(1) 000122 000000 HALT
(1) 000124 000126 .+2
(1) 000126 000000 HALT
(1) 000130 000132 .+2
(1) 000132 000000 HALT
(1) 000134 000136 .+2
(1) 000136 000000 HALT
(1) 000140 000142 .+2
(1) 000142 000000 HALT
(1) 000144 000146 .+2
(1) 000146 000000 HALT
(1) 000150 000152 .+2
(1) 000152 000000 HALT
(1) 000154 000156 .+2
(1) 000156 000000 HALT
(1) 000160 000162 .+2
(1) 000162 000000 HALT
(1) 000164 000166 .+2
(1) 000166 000000 HALT
(1) 000170 000172 .+2
(1) 000172 000000 HALT

```

```

6046 000200 000200 .=200
6047 000200 005067 000202 CLR $PASS ; CLEAR PASS COUNT
6048 000204 005067 000172 CLR $FATAL
6049 000210 005067 000170 CLR $TESTN
6050 000214 000137 001246 JMP @#START1 ; INITIAL START
6051 000300 000300 .=300 ; DEVICE INTERRUPT VECTORS
6052 000300 000302 .+2
6053 000302 000000 HALT
6054 000304 000306 .+2
6055 000306 000000 HALT
6056 000400 000400 .=400

```

.SBTTL APT MAILBOX-ETABLE

```

(1)
(2)
(1)
(1) 000400
(1) 000400 000000 $MAIL:
(1) 000402 000000 $MSGTY: .WORD AMSGTY ;; APT MAILBOX
(1) 000404 000000 $FATAL: .WORD AFATAL ;; MESSAGE TYPE CODE
(1) 000406 000000 $TESTN: .WORD ATESTN ;; FATAL ERROR NUMBER
(1) 000410 000000 $PASS: .WORD APASS ;; TEST NUMBER
(1) 000412 000000 $DEVCT: .WORD ADEVCT ;; PASS COUNT
(1) 000414 000000 $UNIT: .WORD AUNIT ;; DEVICE COUNT
(1) 000416 000000 $MSGAD: .WORD AMSGAD ;; I/O UNIT NUMBER
(1) 000420 000000 $MSGLG: .WORD AMSGLG ;; MESSAGE ADDRESS
(1) 000420 000 $ETABLE: ;; MESSAGE LENGTH
(1) 000420 000 $ENV: .BYTE AENV ;; APT ENVIRONMENT TABLE
;; ENVIRONMENT BYTE

```

(1) 000421 000
(1)
(1) 000422 000000
(1) 000424 000000
(1) 000426 000000
(1)
(1)
(1)
(1)
(1)
(1)
(1) 000430
(1)
6058
(1)
(2)
(1)
(2)
(1) 000430
(1) 000024 000024
(1) 000024 000200
(1) 000044 000044
(1) 000044 000430
(1) 000430
(2)
(1)
(1)
(1)
(1) 000430
(1) 000430 000000
(1) 000432 000400
(1) 000434 000010
(1) 000436 000010
(1) 000440 000000
(1) 000442 000014
6059 001200
6060 000410
6061 000402
6062 000426
6063
6064
6065
6066 001200 174500
6067 001202 174502
6068 001204 174504
6069 001206 174503
6070 001210 000340
6071 001212 000342
6072 001214 000344
6073 001216 000346
6074
6075
6076
6077 001220 174500
6078 001222 174502
6079 001224 174504

```
$ENVM: .BYTE AENVM
::ENVIRONMENT MODE BITS
$SWREG: .WORD ASWREG ::APT SWITCH REGISTER
$USWR: .WORD AUSWR ::USER SWITCHES
$CPUOP: .WORD ACPUOP ::CPU TYPE,OPTIONS
::*
::* BITS 15-11=CPU TYPE
::* 11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
::* 11/70=06,PDQ=07,Q=10
::* BIT 10=REAL TIME CLOCK
::* BIT 9=FLOATING POINT PROCESSOR
::* BIT 8=MEMORY MANAGEMENT

$ETEND:
.MEXIT
.SBTTL APT PARAMETER BLOCK

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

$APTHD:
$SHIBTS: .WORD 0 ::TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
$MADDR: .WORD $MAIL ::ADDRESS OF APT MAILBOX (BITS 0-15)
$STMT: .WORD 10 ::RUN TIM OF LONGEST TEST
$PASTM: .WORD 10 ::RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
$UNITM: .WORD 0 ::ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
.WORD $ETEND-$MAIL/2 ::LENGTH MAILBOX-ETABLE(WORDS)
.=1200

DEVcnt=$DEVCT
ERRNUM=$FATAL
OPTION=$CPUOP

:THIS TABLE CONTAINS INITIAL REGISTER AND VECTOR ADDRESSES

RCSR: CSR
CSR+2
CSR+4
CSR+3
RCSR1: 340
342
344
346

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

DRCSR: 174500 ;ADDRESS OF DR11-C STATUS REGISTER
DROBUF: 174502 ;ADDRESS OF DR OUTPUT BUFFER REG.
DRIBUF: 174504 ;ADDRESS OF DR INPUT BUFFER REG.
```

```

6080 001226 174503          DRBHIO: 174503          ;HIGH BYTE OF OUTPUT BUFFER REG.
6081
6082 001230 000340          DRVECA: 340           ;INTERRUPT VECTOR OF UNIT UNDER TEST
6083 001232 000342          DRLVLA: 342
6084 001234 000344          DRVECB: 344           ;INTERRUPT VECTOR
6085 001236 000346          DRLVLB: 346
6086 001240 000000          XORFLG: 0
6087
6088 001242 000000          COUNT: 0             ;COUNT LOCATION
6089 001244 000240          PL: 240              ;PRIORITY LEVEL
6090
6091 001246 012706 001200    START1: MOV #STKPTR,SP
6092 001252 000137 001256    JMP @#START
6093 ;+*****
6094
6095 ;+ ROUTINE TO PRINT DIAGNOSTIC'S TITLE IF ALLOWED BY APT MODE
6096
6097 ; ALL CODE ADDED FOR TITLE PRINTOUT IS NOTED
6098 ; WITH A ;+ IN THE COMMENT SECTION.
6099 ;+*****
6100 001256 132767 000040 177135 START: BITB #40, $ENVM ;+ WILL APT ALLOW PRINTING?
6101 001264 001013          BNE AROUND           ;+ NO BRANCH AROUND
6102 001266 012700 004616    MOV #TITLE1,R0       ;+ GET STARTING ADDRESS OF TITLE MSG
6103 001272 105737 177564    LOOP1: TSTB @#TPS    ;+ CHECK IF TERMINAL READY
6104 001276 100375          BPL LOOP1            ;+ LOOP IF NOT
6105 001300 112037 177566    MOVB (R0)+,@#TPB    ;+ PRINT A CHARACTER
6106 001304 001372          BNE LOOP1            ;+ BRANCH BACK IF NOT DONE PRINTING
6107 001306 105737 177564    LOOP: TSTB @#TPS
6108 001312 100375          BPL LOOP
6109 ;+*****
6110
6111 ; INITIALIZE ADDRESS AND VECTORS
6112
6113 001314 012700 001200    AROUND: MOV #RCSR, R0 ; GET ADDRESS OF UNIT UNDER TEST
6114 001320 012701 005000    MOV #5000, R1
6115 001324 077101          1$: SOB R1,1$
6116 001326 012701 001220    MOV #DRCSR,R1
6117
6118 001332 012737 004672 000024 MOV #PFAIL,@#24
6119 001340 012021          MOV (R0)+,(R1)+     ;LOAD INITIAL TEST ADDRESSES
6120 001342 012021          MOV (R0)+,(R1)+
6121 001344 012021          MOV (R0)+,(R1)+
6122 001346 012021          MOV (R0)+,(R1)+
6123 001350 012021          MOV (R0)+,(R1)+
6124 001352 012021          MOV (R0)+,(R1)+
6125 001354 012021          MOV (R0)+,(R1)+
6126
6127
6128 001356          ;DOES RESET CLEAR REGISTER?
(2) 001356 012767 000001 177020 TST1:
6129 001364 016705 177630    LP1: MOV #1,$TESTN ; MOVE TEST NUMBER TO MAILBOX
6130 001370 106427 000200    MOV DRCSR,R5 ;GET ADDRESS OF STATUS REGISTER
6131 001374 012737 001456 000004 MTPS #200 ;TURN OFF INTERRUPTS
6132 001402 012777 177777 177612 MOV #ERR1,@#4 ;SET TIME OUT TRAP VECTOR
6133 001410 000005          MOV #-1,@DROBUF ;PRESET OUTPUT BUFFER
6134 001412 017700 177604    RESET ;CLEAR DATA REGISTER
MOV @DROBUF,R0 ;GET RESULT OF RESET

```

```

6135 001416 001450      BEQ      CON
6136 001420 032767 040000 176774  BIT      #BIT14,$SWREG ; CHECK FOR LOOP ON ERROR
(1) 001426 001356      BNE      LP1          ; GO TO LOOP ERROR
(2) 001430 012767 000001 176744  MOV      #1,$FATAL
(1) 001436 012767 000001 176734  MOV      #1,$MSGTY ; MOVE ERROR NUM TO MAILBOX
(1) 001444 005767 176752  TST      $SWREG      ; CHECK FOR HALT ON ERROR
(1) 001450 100401      BMI      1$          ; CONTINUE IF SET
(1) 001452 000000      HALT      ;<DATA REG DID NOT CLEAR>
(1) 001454      1$:
6137 001454 000431      BR       CON
6138 001456      ERR1:
(1) 001456 032767 040000 176736  BIT      #BIT14,$SWREG ; CHECK FOR LOOP ON ERROR
(1) 001464 001337      BNE      LP1          ; GO TO LOOP ERROR
(2) 001466 012767 000002 176706  MOV      #2,$FATAL
(1) 001474 012767 000001 176676  MOV      #1,$MSGTY ; MOVE ERROR NUM TO MAILBOX
(1) 001502 005767 176714  TST      $SWREG      ; CHECK FOR HALT ON ERROR
(1) 001506 100401      BMI      1$          ; CONTINUE IF SET
(1) 001510 000000      HALT      ;<TIME WHEN ADDRESSING PLU>
(1) 001512      1$:
(1) 001512 032767 002000 176702  BIT      #BIT10,$SWREG ; CHECK FOR LOOP ON TEST
(1) 001520 001316      BNE      TST1         ; GO TO LOOP ON TEST
6139 001522 000407      BR       TST2
6140 001524 012706 001200      5$: MOV      #STKPTR,SP ; RESET STACK POINTER
6141 001530 012737 000006 000004  MOV      #6,@#4     ; RESTORE TIME OUT TRAP
6142 001536 000765      BR       1$
6143 001540 000772      CON: BR       -12
6144      ; TEST "NEWDATA RDY" AND "DATATRANS" SIGNALS IN PLU
6145      ; NOTE***** THE PLU TEST MODULE MUST BE INSTALLED
6146      ; TO EXECUTE THIS TEST
6147      ;
6148 001542      TST2:
(2) 001542 012767 000002 176634  MOV      #2,$TESTN ; MOVE TEST NUMBER TO MAILBOX
6149 001550 032767 001000 176644  BIT      #BIT9,$SWREG
6150 001556 001505      BEQ      TST3         ; SKIP TEST IF NOT SELECTED
6151 001560 012706 001200      MOV      #STKPTR,SP ; SET UP STACK POINTER
6152 001564 000005      RESET      ; CLEAR EVERYTHING
6153      ; THIS RESET SHOULD INITIALIZE THE
6154      ; SIGNAL LATCHES IN THE TEST MODULE
6155 001566 012777 031460 177426  MOV      #31460,@DROBUF ; PRIME THE LATCHES
6156 001574 000240      NOP
6157 001576 000240      NOP
6158 001600 017700 177420      @DRIBUF,R0 ; TIMING ALLOWANCE
6159 001604 032700 000001      BIT      #BIT0,R0 ; GET DATA
6160 001610 001016      BNE      T2CON       ; CHECK DATA TRANS SIG
6161 001612 032767 040000 176602  BIT      #BIT14,$SWREG ; CONTINUE IF PRESENT
(1) 001620 001350      BNE      TST2         ; CHECK FOR LOOP ON ERROR
(2) 001622 012767 000003 176552  MOV      #3,$FATAL ; GO TO LOOP ERROR
(1) 001630 012767 000001 176542  MOV      #1,$MSGTY ; MOVE ERROR NUM TO MAILBOX
(1) 001636 005767 176560  TST      $SWREG      ; CHECK FOR HALT ON ERROR
(1) 001642 100401      BMI      1$          ; CONTINUE IF SET
(1) 001644 000000      HALT      ;<NO DATA TRANS SIGNAL>
(1) 001646      1$:
6162 001646 032700 000002      T2CON: BIT      #BIT1,R0 ; CHECK NEW DATA RDY SIGNAL
6163 001652 001016      BNE      T2CN1       ; CONTINUE IF OK
6164 001654 032767 040000 176540  BIT      #BIT14,$SWREG ; CHECK FOR LOOP ON ERROR
(1) 001662 001371      BNE      T2CON       ; GO TO LOOP ERROR

```

NO NEW DATA RDY SIGNAL

SEQ 0014

```

(2) 001664 012767 000004 176510 MOV #4,$FATAL
(1) 001672 012767 000001 176500 MOV #1,$MSGTY ; MOVE ERROR NUM TO MAILBOX
(1) 001700 005767 176516 TST $$SWREG ; CHECK FOR HALT ON ERROR
(1) 001704 100401 BMI 1$ ; CONTINUE IF SET
(1) 001706 000000 HALT ;<NO NEW DATA RDY SIGNAL>
(1) 001710
1$:
6165 001710 000005 T2CN1: RESET ; CLEAR EVERYTHING
6166 001712 000240 NOP
6167 001714 000240 NOP
6168 001716 017700 177302 MOV @DRIBUF,RO ; CHECK SIGNAL LATCHES
6169 001722 005700 TST RO ; SHOULD BE CLEAR
6170 001724 001416 BEQ 1$ ; CONTINUE IF CLEAR
6171 001726 032767 040000 176466 BIT #BIT14,$SWREG ; CHECK FOR LOOP ON ERROR
(1) 001734 001365 BNE T2CN1 ; GO TO LOOP ERROR
(2) 001736 012767 000005 176436 MOV #5,$FATAL
(1) 001744 012767 000001 176426 MOV #1,$MSGTY ; MOVE ERROR NUM TO MAILBOX
(1) 001752 005767 176444 TST $$SWREG ; CHECK FOR HALT ON ERROR
(1) 001756 100401 BMI 1$ ; CONTINUE IF SET
(1) 001760 000000 HALT ;<SIGNALS DID NOT CLEAR>
(1) 001762
1$:
(1) 001762 032767 002000 176432 BIT #BIT10,$SWREG ; CHECK FOR LOOP ON TEST
(1) 001770 001264 BNE TST2 ; GO TO LOOP ON TEST
6172 001772
TST3:
(2) 001772 012767 000003 176404 MOV #3,$TESTN ; MOVE TEST NUMBER TO MAILBOX
6173 002000 012777 177777 177214 MOV #-1,@DROBUF ; ALL ONES TO REGISTER
6174 002006 017700 177210 MOV @DROBUF,RO
6175 002012 022700 177777 CMP #-1,RO
6176 002016 001416 BEQ 1$
6177 002020 032767 040000 176374 BIT #BIT14,$SWREG ; CHECK FOR LOOP ON ERROR
(1) 002026 001361 BNE TST3 ; GO TO LOOP ERROR
(2) 002030 012767 000006 176344 MOV #6,$FATAL
(1) 002036 012767 000001 176334 MOV #1,$MSGTY ; MOVE ERROR NUM TO MAILBOX
(1) 002044 005767 176352 TST $$SWREG ; CHECK FOR HALT ON ERROR
(1) 002050 100401 BMI 1$ ; CONTINUE IF SET
(1) 002052 000000 HALT ;<REGISTER WILL NOT HOLD ALL ONES>
(1) 002054
1$:
(1) 002054 032767 002000 176340 BIT #BIT10,$SWREG ; CHECK FOR LOOP ON TEST
(1) 002062 001343 BNE TST3 ; GO TO LOOP ON TEST
6178
6179 ;WRAP CABLE MUST BE INSTALLED TO EXECUTE THIS TEST
TST4:
6180 002064
(2) 002064 012767 000004 176312 MOV #4,$TESTN ; MOVE TEST NUMBER TO MAILBOX
6181 002072 032767 000400 176322 BIT #BIT8,$SWREG
6182 002100 001031 BNE TST5 ; SKIP TEST IF NOT SELECTED
6183 002102 012777 177777 177112 MOV #-1,@DROBUF
6184 002110 000005 RESET ; SET DATA TO ALL ONES
6185 002112 005777 177106 TST @DRIBUF ; REGISTER SHOULD CLEAR
6186 002116 001416 BEQ 1$
6187 002120 032767 040000 176274 BIT #BIT14,$SWREG ; CHECK FOR LOOP ON ERROR
(1) 002126 001356 BNE TST4 ; GO TO LOOP ERROR
(2) 002130 012767 000007 176244 MOV #7,$FATAL
(1) 002136 012767 000001 176234 MOV #1,$MSGTY ; MOVE ERROR NUM TO MAILBOX
(1) 002144 005767 176252 TST $$SWREG ; CHECK FOR HALT ON ERROR
(1) 002150 100401 BMI 1$ ; CONTINUE IF SET
(1) 002152 000000 HALT ;<REGISTER DID NOT CLEAR BY RESET>
(1) 002154
1$:
  
```

REGISTER DID NOT CLEAR BY RESET

SEQ 0015

```

(1) 002154 032767 002000 176240 BIT #BIT10,$SWREG ; CHECK FOR LOOP ON TEST
(1) 002162 001340 BNE TST4 ; GO TO LOOP ON TEST
6188
6189 002164 TST5: MOV #5,$TESTN ; MOVE TEST NUMBER TO MAILBOX
(2) 002164 012767 000005 176212 MOV #52525,@DROBUF ; LOAD TEST DATA INTO BUFFER
6190 002172 012777 052525 177022 MOV @DROBUF,R0 ; COPY DATA FROM BUFFER TO R0
6191 002200 017700 177016 MOV @DROBUF,R0 ; COPY DATA FROM BUFFER TO R0
6192 002204 022700 052525 CMP #52525,R0 ; COMPARE DATA
6193 002210 001416 BEQ 1$ ; BR IF DATA IS CORRECT
6194 002212 032767 040000 176202 BIT #BIT14,$SWREG ; CHECK FOR LOOP ON ERROR
(1) 002220 001361 BNE TST5 ; GO TO LOOP ERROR
(2) 002222 012767 000010 176152 MOV #10,$FATAL
(1) 002230 012767 000001 176142 MOV #1,$MSGTY ; MOVE ERROR NUM TO MAILBOX
(1) 002236 005767 176160 TST $SWREG ; CHECK FOR HALT ON ERROR
(1) 002242 100401 BMI 1$ ; CONTINUE IF SET
(1) 002244 000000 HALT ; <INCORRECT DATA IN REG>
(1) 002246 1$:
(1) 002246 032767 002000 176146 BIT #BIT10,$SWREG ; CHECK FOR LOOP ON TEST
(1) 002254 001343 BNE TST5 ; GO TO LOOP ON TEST
6195
6196 002256 TST6: MOV #6,$TESTN ; MOVE TEST NUMBER TO MAILBOX
(2) 002256 012767 000006 176120 MOV #125252,@DROBUF ; LOAD TEST DATA INTO BUFFER
6197 002264 012777 125252 176730 MOV @DROBUF,R0 ; COPY DATA FROM BUFFER TO R0
6198 002272 017700 176724 MOV @DROBUF,R0 ; COPY DATA FROM BUFFER TO R0
6199 002276 022700 125252 CMP #125252,R0 ; COMPARE DATA
6200 002302 001416 BEQ 1$ ; BR IF DATA IS CORRECT
6201 002304 032767 040000 176110 BIT #BIT14,$SWREG ; CHECK FOR LOOP ON ERROR
(1) 002312 001361 BNE TST6 ; GO TO LOOP ERROR
(2) 002314 012767 000011 176060 MOV #11,$FATAL
(1) 002322 012767 000001 176050 MOV #1,$MSGTY ; MOVE ERROR NUM TO MAILBOX
(1) 002330 005767 176066 TST $SWREG ; CHECK FOR HALT ON ERROR
(1) 002334 100401 BMI 1$ ; CONTINUE IF SET
(1) 002336 000000 HALT ; <INCORRECT DATA IN REG>
(1) 002340 1$:
(1) 002340 032767 002000 176054 BIT #BIT10,$SWREG ; CHECK FOR LOOP ON TEST
(1) 002346 001343 BNE TST6 ; GO TO LOOP ON TEST
6202
6203 ;TEST RELIABILITY OF DR11-C OUTPUT BUFFER REGISTER
6204 002350 TST7: MOV #7,$TESTN ; MOVE TEST NUMBER TO MAILBOX
(2) 002350 012767 000007 176026 BUFTST: MOV R5,R2 ; GET ADDRESS OF DRCSR
6205 002356 010502 TST (R2)+ ; R2=ADDRESS OF OUTPUT BUFFER REG.
6206 002360 005722 CLR R3 ; INITIALIZE DATA REGISTER
6207 002362 005003 LP7: MOV R3,(R2) ; SEND THE DATA
6208 002364 010312 CMP (R2),R3 ; CHECK THE RECEIVED DATA
6209 002366 021203 BNE 5$ ; ERROR IF NOT THE SAME
6210 002370 001004 INC R3 ; INCREMENT THE DATA
6211 002372 005203 TSTB R3 ; CHECK FOR END OF DATA
6212 002374 105703 BEQ 1$ ; CONTINUE IF END
6213 002376 001417 BR LP7 ; GO TO SEND DATA IF NOT
6214 002400 000771 5$:
6215 002402 BIT #BIT14,$SWREG ; CHECK FOR LOOP ON ERROR
(1) 002402 032767 040000 176012 BNE LP7 ; GO TO LOOP ERROR
(1) 002410 001365 MOV #12,$FATAL
(2) 002412 012767 000012 175762 MOV #1,$MSGTY ; MOVE ERROR NUM TO MAILBOX
(1) 002420 012767 000001 175752 TST $SWREG ; CHECK FOR HALT ON ERROR
(1) 002426 005767 175770

```



```

(1) 002432 100401      BMI      1$      ; CONTINUE IF SET
(1) 002434 000000      HALT      ;<DATA INCORRECT IN REG>
(1) 002436
(1) 002436 032767 002000 175756 1$:      BIT      #BIT10,$SWREG ; CHECK FOR LOOP ON TEST
(1) 002444 001341      BNE      TST7      ; GO TO LOOP ON TEST
6216 ;TEST THAT BYTE REFERENCE TO DROBUF AFFECT PROPER BYTE ONLY
6217
6218 002446      TST10:
(2) 002446 012767 000010 175730      MOV      #10,$TESTN ; MOVE TEST NUMBER TO MAILBOX
6219 002454 012777 177777 176540      TAG:      MOV      #-1,@DROBUF ; SET ALL ONES IN BUFFER
6220 002462 105077 176534      CLRB     @DROBUF ; CLEAR LOW BYTE
6221 002466 017700 176530      MOV      @DROBUF,R0 ; COPY DATA
6222 002472 022700 177400      CMP      #177400,R0 ; VERIFY THAT LOW BYTE IS CLEAR
6223 002476 001416      BEQ      1$
6224 002500 032767 040000 175714      BIT      #BIT14,$SWREG ; CHECK FOR LOOP ON ERROR
(1) 002506 001362      BNE      TAG      ; GO TO LOOP ERROR
(2) 002510 012767 000013 175664      MOV      #13,$FATAL
(1) 002516 012767 000001 175654      MOV      #1,$MSGTY ; MOVE ERROR NUM TO MAILBOX
(1) 002524 005767 175672      TST      $SWREG ; CHECK FOR HALT ON ERROR
(1) 002530 100401      BMI      1$      ; CONTINUE IF SET
(1) 002532 000000      HALT      ;<LOW BYTE FAILED TO CLEAR>
(1) 002534
(1) 002534 032767 002000 175660 1$:      BIT      #BIT10,$SWREG ; CHECK FOR LOOP ON TEST
(1) 002542 001341      BNE      TST10 ; GO TO LOOP ON TEST
6225
6226 002544      TST11:
(2) 002544 012767 000011 175632      MOV      #11,$TESTN ; MOVE TEST NUMBER TO MAILBOX
6227 002552 012777 177777 176442      MOV      #-1,@DROBUF ; SET ALL ONES IN BUFFER
6228 002560 105077 176442      CLRB     @DRBHI0 ; CLEAR HIGH BYTE
6229 002564 017700 176432      MOV      @DROBUF,R0
6230 002570 022700 000377      CMP      #377,R0 ; VERIFY THAT HIGH BYTE IS CLEAR
6231 002574 001416      BEQ      1$
6232 002576 032767 040000 175616      BIT      #BIT14,$SWREG ; CHECK FOR LOOP ON ERROR
(1) 002604 001357      BNE      TST11 ; GO TO LOOP ERROR
(2) 002606 012767 000014 175566      MOV      #14,$FATAL
(1) 002614 012767 000001 175556      MOV      #1,$MSGTY ; MOVE ERROR NUM TO MAILBOX
(1) 002622 005767 175574      TST      $SWREG ; CHECK FOR HALT ON ERROR
(1) 002626 100401      BMI      1$      ; CONTINUE IF SET
(1) 002630 000000      HALT      ;<HIGH BYTE FAILED TO CLEAR>
(1) 002632
(1) 002632 032767 002000 175562 1$:      BIT      #BIT10,$SWREG ; CHECK FOR LOOP ON TEST
(1) 002640 001341      BNE      TST11 ; GO TO LOOP ON TEST
6233 002642      TST12:
(2) 002642 012767 000012 175534      MOV      #12,$TESTN ; MOVE TEST NUMBER TO MAILBOX

```

```

6235
6236 002650 005067 000110      CLR      T12DAT      ;CLEAR DATA LOCATION
6237 002654 012704 002764      MOV      #T12DAT,R4 ;STORE ADDRESS OF DATA LOCATION
6238 002660 005077 176336      CLR      @DROBUF    ;CLEAR OUTPUT BUFFER
6239 002664 105077 176336      T12CON: CLR      @DRBHIO ;CLEAR HIGH BYTE
3$:  INCB  @DRBHIO    ;INCREMENT HIGH BYTE
6240 002670 105277 176332      INCB    1(R4)      ;INCREMENT COMPARISON DATA
6241 002674 105264 000001      INCB    @DROBUF,(R4) ;COMPARE DATA
6242 002700 027714 176316      CMP     6$        ;BRANCH ON ERROR
6243 002704 001004          BNE     4$        ;FINISHED?
6244 002706 105764 000001      TSTB   1(R4)      ;YES
6245 002712 001417          BEQ     1$        ;CONTINUE TESTING
6246 002714 000765          BR     3$
6247 002716          6$:
(1) 002716 032767 040000 175476  BIT     #BIT14,$SWREG ; CHECK FOR LOOP ON ERROR
(1) 002724 001346          BNE     TST12     ; GO TO LOOP ERROR
(2) 002726 012767 000015 175446  MOV     #15,$FATAL
(1) 002734 012767 000001 175436  MOV     #1,$MSGTY  ; MOVE ERROR NUM TO MAILBOX
(1) 002742 005767 175454  TST     $SWREG    ; CHECK FOR HALT ON ERROR
(1) 002746 100401          BMI    1$        ; CONTINUE IF SET
(1) 002750 000000          HALT                    ;<DATA INCORRECT IN REG>
(1) 002752          1$:
(1) 002752 032767 002000 175442  BIT     #BIT10,$SWREG ; CHECK FOR LOOP ON TEST
(1) 002760 001330          BNE     TST12     ; GO TO LOOP ON TEST
6248 002762 000401          BR     TST13
6249 002764 000000      T12DAT: .WORD 0
6250          ;CONTROL STATUS REGISTER (DRCSR) TESTS.
6251 002766      TST13:
(2) 002766 012767 000013 175410  MOV     #13,$TESTN ; MOVE TEST NUMBER TO MAILBOX
6252 002774 005015          CLR     (R5)      ;CLEAR STATUS REGISTER
6253 002776 011500          MOV     (R5),R0   ;COPY DATA
6254 003000 032700 000143          BIT     #143,R0   ;VERIFY THAT IE AND CSR BITS ARE CLEAR
6255 003004 001416          BEQ     T13CON    ;IF YES, CONTINUE
6256 003006 032767 040000 175406  BIT     #BIT14,$SWREG ; CHECK FOR LOOP ON ERROR
(1) 003014 001364          BNE     TST13     ; GO TO LOOP ERROR
(2) 003016 012767 000016 175356  MOV     #16,$FATAL
(1) 003024 012767 000001 175346  MOV     #1,$MSGTY  ; MOVE ERROR NUM TO MAILBOX
(1) 003032 005767 175364  TST     $SWREG    ; CHECK FOR HALT ON ERROR
(1) 003036 100401          BMI    1$        ; CONTINUE IF SET
(1) 003040 000000          HALT                    ;<STATUS REG DID NOT CLEAR>
(1) 003042          1$:
6257 003042 012715 000140      T13CON: MOV     #140,@R5  ;INTERRUPT ENABLE FOR A+B
6258 003046 011500          MOV     @R5,R0   ;INTERRUPT ENABLE BITS SET?
6259 003050 032700 000140          BIT     #140,R0   ;CONTINUE IF YES
6260 003054 001016          BNE     1$        ;CHECK FOR LOOP ON ERROR
6261 003056 032767 040000 175336  BIT     #BIT14,$SWREG ; GO TO LOOP ERROR
(1) 003064 001366          BNE     T13CON
(2) 003066 012767 000017 175306  MOV     #17,$FATAL
(1) 003074 012767 000001 175276  MOV     #1,$MSGTY  ; MOVE ERROR NUM TO MAILBOX
(1) 003102 005767 175314  TST     $SWREG    ; CHECK FOR HALT ON ERROR
(1) 003106 100401          BMI    1$        ; CONTINUE IF SET
(1) 003110 000000          HALT                    ;<ENABLE BITS NOT ON>
(1) 003112          1$:
(1) 003112 032767 002000 175302  BIT     #BIT10,$SWREG ; CHECK FOR LOOP ON TEST
(1) 003120 001322          BNE     TST13     ; GO TO LOOP ON TEST
6262
6263 003122      TST14:

```

ENABLE BITS NOT ON

```

(2) 003122 012767 000014 175254      MOV #14,$TESTN      ; MOVE TEST NUMBER TO MAILBOX
6264 003130 012715 000140              MOV #140,@R5        ; SET INTERRUPT ENABLE FLOPS
6265 003134 000005                      RESET               ; CLEAR THOSE FLOPS
6266 003136 011500                      MOV @R5,R0          ; COPY CONTENTS OF DRCSR TO R0
6267 003140 032700 000140              BIT #140,R0         ; TEST INTERRUPT ENABLE BITS
6268 003144 001416                      BEQ 1$              ; BR IF CLEARED
6269 003146 032767 040000 175246      BIT #BIT14,$SWREG  ; CHECK FOR LOOP ON ERROR
(1) 003154 001362                      BNE TST14           ; GO TO LOOP ERROR
(2) 003156 012767 000020 175216      MOV #20,$FATAL
(1) 003164 012767 000001 175206      MOV #1,$MSGTY       ; MOVE ERROR NUM TO MAILBOX
(1) 003172 005767 175224              TST $SWREG          ; CHECK FOR HALT ON ERROR
(1) 003176 100401                      BMI 1$              ; CONTINUE IF SET
(1) 003200 000000                      HALT                ; <INTERRUPT ENABLE DID NOT CLEAR>
(1) 003202                                1$:
(1) 003202 032767 002000 175212      BIT #BIT10,$SWREG  ; CHECK FOR LOOP ON TEST
(1) 003210 001344                      BNE TST14           ; GO TO LOOP ON TEST
6270
6271 003212                                TST15:
(2) 003212 012767 000015 175164      MOV #15,$TESTN     ; MOVE TEST NUMBER TO MAILBOX
6272 003220 052715 000001              BIS #1,@R5          ; SHOULD SET REQ A ALSO
6273 003224 032715 000201              BIT #201,@R5        ; VERIFY THAT REQ A IS SET
6274 003230 001402                      BEQ 5$              ; FLAG ERROR MESSAGE IF NO
6275 003232 005015                      CLR @R5             ; CLEAR STATUS REGISTER
6276 003234 000416                      BR 1$               ; GO TO NEXT TEST
6277 003236                                5$:
(1) 003236 032767 040000 175156      BIT #BIT14,$SWREG  ; CHECK FOR LOOP ON ERROR
(1) 003244 001362                      BNE TST15           ; GO TO LOOP ERROR
(2) 003246 012767 000021 175126      MOV #21,$FATAL
(1) 003254 012767 000001 175116      MOV #1,$MSGTY       ; MOVE ERROR NUM TO MAILBOX
(1) 003262 005767 175134              TST $SWREG          ; CHECK FOR HALT ON ERROR
(1) 003266 100401                      BMI 1$              ; CONTINUE IF SET
(1) 003270 000000                      HALT                ; <A REQ DID NOT SET>
(1) 003272                                1$:
6278

```

```

6280 003272          TST16:
(2) 003272 012767 000016 175104  MOV #16,$TESTN      ; MOVE TEST NUMBER TO MAILBOX
6281 003300          BIS #2,@R5          ; SHOULD SET REQ B
6282 003304 032715 100002          BIT #100002,@R5    ; VERIFY THAT REQ B IS SET
6283 003310 001402          BEQ 5$             ; FLAG ERROR MESSAGE IF NO
6284 003312 005015          CLR @R5            ; CLEAR STATUS REGISTER
6285 003314 000416          BR 1$              ; GO TO NEXT TEST
6286 003316          5$:
(1) 003316 032767 040000 175076  BIT #BIT14,$SWREG  ; CHECK FOR LOOP ON ERROR
(1) 003324 001362          BNE TST16         ; GO TO LOOP ERROR
(2) 003326 012767 000022 175046  MOV #22,$FATAL
(1) 003334 012767 000001 175036  MOV #1,$MSGTY      ; MOVE ERROR NUM TO MAILBOX
(1) 003342 005767 175054          TST $SWREG         ; CHECK FOR HALT ON ERROR
(1) 003346 100401          BMI 1$            ; CONTINUE IF SET
(1) 003350 000000          HALT              ; <B REQ DID NOT SET>
(1) 003352          1$:
6287
6288 003352          TST17:
(2) 003352 012767 000017 175024  MOV #17,$TESTN    ; MOVE TEST NUMBER TO MAILBOX
6289 003360 106427 000300          MTPS #300         ; LOCK OUT INTERRUPTS
6290 003364 052715 177777          BIS #-1,@R5       ; LOAD ALL ONES IN STATUS REGISTER
6291 003370 022715 100343          CMP #100343,(R5) ; VERIFY THAT ALL WRITE BITS ARE SET IN DRCSR
6292 003374 001416          BEQ T17CON        ; BR IF SET
6293 003376 032767 040000 175016  BIT #BIT14,$SWREG  ; CHECK FOR LOOP ON ERROR
(1) 003404 001362          BNE TST17         ; GO TO LOOP ERROR
(2) 003406 012767 000023 174766  MOV #23,$FATAL
(1) 003414 012767 000001 174756  MOV #1,$MSGTY      ; MOVE ERROR NUM TO MAILBOX
(1) 003422 005767 174774          TST $SWREG         ; CHECK FOR HALT ON ERROR
(1) 003426 100401          BMI 1$            ; CONTINUE IF SET
(1) 003430 000000          HALT              ; <INCORRECT DATA IN REG>
(1) 003432          1$:
6294 003432 042715 000003          T17CON: BIC #3,@R5      ; CLEAR CSR BITS
6295 003436 032715 000140          BIT #140,@R5     ; TEST INTERRUPT ENABLE BITS
6296 003442 001016          BNE 1$            ; CONTINUE IF STILL SET
6297 003444 032767 040000 174750  BIT #BIT14,$SWREG  ; CHECK FOR LOOP ON ERROR
(1) 003452 001367          BNE T17CON        ; GO TO LOOP ERROR
(2) 003454 012767 000024 174720  MOV #24,$FATAL
(1) 003462 012767 000001 174710  MOV #1,$MSGTY      ; MOVE ERROR NUM TO MAILBOX
(1) 003470 005767 174726          TST $SWREG         ; CHECK FOR HALT ON ERROR
(1) 003474 100401          BMI 1$            ; CONTINUE IF SET
(1) 003476 000000          HALT              ; <WRONG BITS SET>
(1) 003500          1$:
(1) 003500 032767 002000 174714  BIT #BIT10,$SWREG  ; CHECK FOR LOOP ON TEST
(1) 003506 001321          BNE TST17         ; GO TO LOOP ON TEST
6298
6299 003510          TST20:
(2) 003510 012767 000020 174666  MOV #20,$TESTN    ; MOVE TEST NUMBER TO MAILBOX
6300 003516 106427 000300          MTPS #300         ; LOCK OUT INTERRUPTS
6301 003522 052715 000003          BIS #3,@R5       ; SET CSR BITS
6302 003526 000005          RESET            ; SHOULD CLEAR INTERRUPT ENABLE FLOPS
6303 003530 032715 000140          BIT #140,@R5     ; VERIFY THAT FLOPS ARE CLEARED
6304 003534 001416          BEQ 1$            ; BR IF YES
6305 003536 032767 040000 174656  BIT #BIT14,$SWREG  ; CHECK FOR LOOP ON ERROR
(1) 003544 001361          BNE TST20         ; GO TO LOOP ERROR
(2) 003546 012767 000025 174626  MOV #25,$FATAL
(1) 003554 012767 000001 174616  MOV #1,$MSGTY      ; MOVE ERROR NUM TO MAILBOX
  
```

RESET DID NOT CLEAR BITS

SEQ 0020

```

(1) 003562 005767 174634      TST  $SWREG      ; CHECK FOR HALT ON ERROR
(1) 003566 100401              BMI  1$         ; CONTINUE IF SET
(1) 003570 000000              HALT                ;<RESET DID NOT CLEAR BITS>
(1) 003572 032767 002000 174622 1$: BIT  #BIT10,$SWREG ; CHECK FOR LOOP ON TEST
(1) 003600 001343              BNE  TST20      ; GO TO LOOP ON TEST
6306
6307      ;NOTE: THE WRAP CABLE MUST BE INSTALLED TO EXECUTE
6308      TESTS 21-27
6309 003602      TST21:
(2) 003602 012767 000021 174574  MOV  #21,$TESTN  ; MOVE TEST NUMBER TO MAILBOX
6310 003610 032767 000400 174604  BIT  #BIT8,$SWREG
6311 003616 001402              BEQ  LP21       ; DO TESTS IF NOT INHIBITED
6312 003620 000167 000710      JMP  TST999     ; IF INHIBITED
6313 003624 005015      LP21: CLR  @R5       ; CLEAR STATUS REGISTER
6314 003626 005215      INC  @R5       ; SET CSRO
6315 003630 105715      TSTB @R5      ; CHECK FOR REQ A FLAG
6316 003632 100116      BMI  1$         ; BR IF SET
6317 003634 032767 040000 174560  BIT  #BIT14,$SWREG ; CHECK FOR LOOP ON ERROR
(1) 003642 001357      BNE  TST21     ; GO TO LOOP ERROR
(2) 003644 012767 000026 174530  MOV  #26,$FATAL
(1) 003652 012767 000001 174520  MOV  #1,$MSGTY  ; MOVE ERROR NUM TO MAILBOX
(1) 003660 005767 174536      TST  $SWREG    ; CHECK FOR HALT ON ERROR
(1) 003664 100401              BMI  1$         ; CONTINUE IF SET
(1) 003666 000000              HALT                ;<BIT0 DID NOT SET BIT7>
(1) 003670 032767 002000 174524 1$: BIT  #BIT10,$SWREG ; CHECK FOR LOOP ON TEST
(1) 003676 001341      BNE  TST21     ; GO TO LOOP ON TEST
6318
  
```

BIT0 DID NOT SET BIT7

SEQ 0021

```

6320 003700          TST22:
(2) 003700 012767 000022 174476  MOV #22,$TESTN ; MOVE TEST NUMBER TO MAILBOX
6321 003706 012715 000002          MOV #2,@R5 ; SET CSR1
6322 003712 005715          TST @R5 ; CHECK FOR REQ B FLAG
6323 003714 100416          BMI 1$ ; BR IF SET
6324 003716 032767 040000 174476  BIT #BIT14,$SWREG ; CHECK FOR LOOP ON ERROR
(1) 003724 001365          BNE TST22 ; GO TO LOOP ERROR
(2) 003726 012767 000027 174446  MOV #27,$FATAL
(1) 003734 012767 000001 174436  MOV #1,$MSGTY ; MOVE ERROR NUM TO MAILBOX
(1) 003742 005767 174454          TST $SWREG ; CHECK FOR HALT ON ERROR
(1) 003746 100401          BMI 1$ ; CONTINUE IF SET
(1) 003750 000000          HALT ; <BIT1 DID NOT SET BIT15>
(1) 003752          1$:
(1) 003752 032767 002000 174442  BIT #BIT10,$SWREG ; CHECK FOR LOOP ON TEST
(1) 003760 001347          BNE TST22 ; GO TO LOOP ON TEST
6325 003762          TST23:
(2) 003762 012767 000023 174414  MOV #23,$TESTN ; MOVE TEST NUMBER TO MAILBOX
6326 003770 012715 000003          MOV #3,@R5 ; CSRO AND CSR1
6327 003774 011500          MOV (R5),R0 ; COPY DATA
6328 003776 022700 100203          CMP #100203,R0 ; COMPARE DATA
6329 004002 001416          BEQ 1$ ; BR IF GOOD DATA IS RECEIVED
6330 004004 032767 040000 174410  BIT #BIT14,$SWREG ; CHECK FOR LOOP ON ERROR
(1) 004012 001363          BNE TST23 ; GO TO LOOP ERROR
(2) 004014 012767 000030 174360  MOV #30,$FATAL
(1) 004022 012767 000001 174350  MOV #1,$MSGTY ; MOVE ERROR NUM TO MAILBOX
(1) 004030 005767 174366          TST $SWREG ; CHECK FOR HALT ON ERROR
(1) 004034 100401          BMI 1$ ; CONTINUE IF SET
(1) 004036 000000          HALT ; <INCORRECT BITS SET IN REG>
(1) 004040          1$:
(1) 004040 032767 002000 174354  BIT #BIT10,$SWREG ; CHECK FOR LOOP ON TEST
(1) 004046 001345          BNE TST23 ; GO TO LOOP ON TEST
6331
6332          :CAN WE RAISE INTERRUPT "A"
6333 004050          TST24:
(2) 004050 012767 000024 174326  MOV #24,$TESTN ; MOVE TEST NUMBER TO MAILBOX
6334 004056 106427 000300          MTPS #300 ; LOCK OUT INTERRUPTS
6335 004062 012706 001200          MOV #STKPTR,SP ; INITIALIZE STACK POINTER
6336 004066 012777 004110 175134  MOV #4,@DRVECA ; INTERRUPT RETURN POINTER
6337 004074 012715 000101          MOV #101,@R5 ; INTERRUPT ENABLE AND CSRO
6338 004100 106427 000000          MTPS #0 ; ALLOW INTERRUPTS
6339 004104 000240          NOP
6340 004106 000402          BR 5$ ; NO INTERRUPT
6341
6342 004110 005015          4$: CLR @R5 ; CLEAR INTERRUPT ENABLE
6343 004112 000416          BR 1$ ; GO TO NEXT TEST
6344 004114          5$:
(1) 004114 032767 040000 174300  BIT #BIT14,$SWREG ; CHECK FOR LOOP ON ERROR
(1) 004122 001352          BNE TST24 ; GO TO LOOP ERROR
(2) 004124 012767 000031 174250  MOV #31,$FATAL
(1) 004132 012767 000001 174240  MOV #1,$MSGTY ; MOVE ERROR NUM TO MAILBOX
(1) 004140 005767 174256          TST $SWREG ; CHECK FOR HALT ON ERROR
(1) 004144 100401          BMI 1$ ; CONTINUE IF SET
(1) 004146 000000          HALT ; <NO A INTERRUPT>
(1) 004150          1$:
(1) 004150 032767 002000 174244  BIT #BIT10,$SWREG ; CHECK FOR LOOP ON TEST
(1) 004156 001334          BNE TST24 ; GO TO LOOP ON TEST
  
```

```

6345
6346      ;RAISE INTERRUPT 'B'
6347      TST25:
(2) 004160 012767 000025 174216      MOV      #25,$TESTN      ; MOVE TEST NUMBER TO MAILBOX
6348 004166 012706 001200              MOV      #STKPTR,SP      ; INITIALIZE STACK POINTER
6349 004172 106427 000300              MTPS     #300            ; LOCK OUT INTERRUPTS
6350 004176 012777 004220 175030      MOV      #4$,@DRVECB     ; INTERRUPT RETURN POINTER
6351 004204 012715 000042              MOV      #42,@R5        ; IE AND CSR1
6352 004210 106427 000000              MTPS     #0              ; ALLOW INTERRUPTS
6353 004214 000240              NOP
6354 004216 000402              BR       5$              ; NO INTERRUPT
6355 004220 005015      4$:      CLR      @R5              ; CLEAR INTERRUPT ENABLE
6356 004222 000416              BR       1$              ; GO TO NEXT TEST
6357 004224      5$:
(1) 004224 032767 040000 174170      BIT      #BIT14,$SWREG   ; CHECK FOR LOOP ON ERROR
(1) 004232 001352              BNE     TST25            ; GO TO LOOP ERROR
(2) 004234 012767 000032 174140      MOV      #32,$FATAL
(1) 004242 012767 000001 174130      MOV      #1,$MSGTY      ; MOVE ERROR NUM TO MAILBOX
(1) 004250 005767 174146      TST     $SWREG           ; CHECK FOR HALT ON ERROR
(1) 004254 100401              BMI     1$              ; CONTINUE IF SET
(1) 004256 000000              HALT
(1) 004260      1$:      ;<NO B INTERRUPT>
(1) 004260 032767 002000 174134      BIT      #BIT10,$SWREG   ; CHECK FOR LOOP ON TEST
(1) 004266 001334              BNE     TST25            ; GO TO LOOP ON TEST
6358
6359      ;TEST FOR INTERRUPT FROM DEVICE
6360      TST26:
(2) 004270 012767 000026 174106      MOV      #26,$TESTN     ; MOVE TEST NUMBER TO MAILBOX
6361 004276 017702 174730              MOV      @DRLVLA,R2     ; SAVE INTERRUPT PSW
6362 004302 016777 174736 174722      LP26:   MOV      PL,@DRLVLA     ; LOCK OUT SUCCESSIVE INTERRUPTS
6363 004310 012706 001200              MOV      #STKPTR,SP     ; INITIALIZE STACK POINTER
6364 004314 012777 004370 174706      MOV      #1$,@DRVECA    ; INTERRUPT RETURN POINTER
6365 004322 012715 000101              MOV      #101,@R5      ; SET INTERRUPT ENABLE-AND CSRO
6366 004326 106427 000000              MTPS     #0              ; ALLOW INTERRUPTS
6367 004332 000240              NOP
6368 004334      5$:
(1) 004334 032767 040000 174060      BIT      #BIT14,$SWREG   ; CHECK FOR LOOP ON ERROR
(1) 004342 001352              BNE     TST26            ; GO TO LOOP ERROR
(2) 004344 012767 000033 174030      MOV      #33,$FATAL
(1) 004352 012767 000001 174020      MOV      #1,$MSGTY      ; MOVE ERROR NUM TO MAILBOX
(1) 004360 005767 174036      TST     $SWREG           ; CHECK FOR HALT ON ERROR
(1) 004364 100401              BMI     1$              ; CONTINUE IF SET
(1) 004366 000000              HALT
(1) 004370      1$:      ;<NO DEVICE INTERRUPT>
(1) 004370 032767 002000 174024      BIT      #BIT10,$SWREG   ; CHECK FOR LOOP ON TEST
(1) 004376 001341              BNE     LP26             ; GO TO LOOP ON TEST
6369 004400 005015      CLR      @R5              ; CLEAR INTERRUPT ENABLE
6370 004402 010277 174624      MOV      R2,@DRLVLA     ; RESTORE INTERRUPT PSW
6371      ; PLU WRAP TEST
6372      TST27:
(2) 004406 012767 000027 173770      MOV      #27,$TESTN     ; MOVE TEST NUMBER TO MAILBOX
6373 004414 005000              CLR      R0              ; SET UP STARTING DATA
6374 004416 010077 174600      WLOOP:  MOV      R0,@DROBUF     ; SEND DATA
6375 004422 027700 174576      CMP     @DRIBUF,R0      ; CHECK THE DATA
6376 004426 001020              PNE     5$              ; ERROR IF NOT RIGHT
6377 004430 005200              INC     R0              ; CHANGE DATA
  
```

6378	004432	001434			BEQ	1\$:NEXT TEST IF END
6379	004434	022700	031460	3\$:	CMP	#31460,R0		: CHECK FOR TEST MODULE CODE
6380	004440	001411			BEQ	4\$		
6381	004442	022700	031461		CMP	#31461,R0		
6382	004446	001406			BEQ	4\$		
6383	004450	022700	031462		CMP	#31462,R0		
6384	004454	001403			BEQ	4\$		
6385	004456	022700	031463		CMP	#31463,R0		
6386	004462	001355			BNE	WLOOP		
6387	004464	005200		4\$:	INC	R0		
6388	004466	000762			BR	3\$: RECHECK DATA CODE
6389	004470			5\$:				
(1)	004470	032767	040000	173724	BIT	#BIT14,\$SWREG		: CHECK FOR LOOP ON ERROR
(1)	004476	001347			BNE	WLOOP		: GO TO LOOP ERROR
(2)	004500	012767	000034	173674	MOV	#34,\$FATAL		
(1)	004506	012767	000001	173664	MOV	#1,\$MSGTY		: MOVE ERROR NUM TO MAILBOX
(1)	004514	005767	173702		TST	\$SWREG		: CHECK FOR HALT ON ERROR
(1)	004520	100401			BMI	1\$: CONTINUE IF SET
(1)	004522	000000			HALT			:<WRAP DATA DID NOT COMPARE>
(1)	004524							
(1)	004524	032767	002000	173670	BIT	#BIT10,\$SWREG		: CHECK FOR LOOP ON TEST
(1)	004532	001325			BNE	TST27		: GO TO LOOP ON TEST


```
6391
6392
6393 004534
6394 004534 005237 000406
6395 004540 132767 000040 173653
6396 004546 001010
6397 004550 012700 004654
6398 004554 105737 177564
6399 004560 100375
6400 004562 112037 177566
6401 004566 001372
6402 004570 013700 000042
6403 004574 001405
6404 004576 000005
6405 004600 004710
6406 004602 000240
6407 004604 000240
6408 004606 000240
6409 004610 000167 174500
6410
6411
6412 004614
6413 177564
6414 177566
6415 004614 177777
6416 004616 047103 040513 026506
        004624 030101 020040 042040
        004632 053122 030461 042040
        004640 040511 047107 051517
        004646 044524 006503 000012
6417 004654 047105 020104 043117
        004662 050040 051501 006523
        004670 000012
6418
```

TST999: INC @#SPASS ; INCREMENT PASS COUNT
BITB #40,\$ENVM ; WILL APT ALLOW PRINTING?
BNE ACT ; NO
MOV #MSG,RO ; GET MESSAGE ADDRESS
WAIT: TSTB @#TPS ; CHECK IF TTY READY
BPL WAIT ; IF NOT
MOVB (RO)+,@#TPB ; PRINT THE CHARACTER
BNE WAIT ; NEXT IF NOT DONE
ACT: MOV @#42,RO ; CHECK ACT
BEQ GOAGIN ; KEEP GOING
RESET
SENDAD: JSR PC,(RO) ; ACT HOOKS
NOP
NOP
NOP
GOAGIN: JMP AROUND ;+ DO ANOTHER PASS
 ;+ ON PASSES >1 THE TITLE PRINTOUT WILL BE SUPPRESSED
.EVEN
STORE:
TPS=177564
TPB=177566
PASSPT: -1
TITLE1: .ASCIZ .CNKAF-A0 DRV11 DIAGNOSTIC. <15><12>
MSG: .ASCIZ .END OF PASS.<15><12>
.EVEN

6420
6421
6422
6423
6424
6425
6426
6427
6428
6429
6430
6431
6432
6433
6434
6435
6436
6437
6438
6439
6440
6441
6442
6443
6444
6445
6446
6447
6448
6449
6450
6451
6452
6453

004672 010046
004674 010146
004676 010246
004700 010346
004702 010446
004704 010546
004706 016746 173112
004712 010637 004726
004716 012737 004730 000024
004724 000000
004726 000000
004730 016706 177772
004734 012667 173064
004740 012605
004742 012604
004744 012603
004746 012602
004750 012601
004752 012600
004754 000137 001256

004760
000100
000102 000002
000140
000140 170000
000142 000300
004760
000001

000024

:ENTER HERE FOR POWER FAIL

```
PFAIL:  MOV    %0,-(6)           ;SAVE REGISTER OR STACK
        MOV    %1,-(6)           ;WHEN POWERING DOWN
        MOV    %2,-(6)
        MOV    %3,-(6)
        MOV    %4,-(6)
        MOV    %5,-(6)
        MOV    24,-(6)
        MOV    %6,@#SAVR6        ;STORE STACK POSITION
        MOV    #RESTAR,@#24
        HALT                       ;HALT ON POWER DOWN NORMAL
SAVR6:  0                           ;STACK IS SAVED HERE
RESTAR:MOV    SAVR6,%6            ;RESTORE REGISTER OFF STACK
        MOV    (6)+,%24           ;WHEN POWERING UP
        MOV    (6)+,%5
        MOV    (6)+,%4
        MOV    (6)+,%3
        MOV    (6)+,%2
        MOV    (6)+,%1
        MOV    (6)+,%0
        JMP    @#START
```

::THE FOLLOWING WAS ADDED TO INIT BRKVEC AND LKVEC TO BE SPECIFIC TO
::SBC 11/21 PROCESSOR.

```
POINT=.                               ;SAVE POINTER
.=100
102,2                                   ;DISMISS BEVNT
.=140
170000                                  ;ODT START ADDRESS
300                                     ;PRIORITY
.=POINT                                 ;RESTORE POINTER
.END
```

ABASE = 000000	6057		
ACDW1 = 000000	6057		
ACDW2 = 000000	6057		
ACPUOP= 000000	6057		
ACT 004570	6396	6402#	
ADDW0 = 000000	6057		
ADDW1 = 000000	6057		
ADDW10= 000000	6057		
ADDW11= 000000	6057		
ADDW12= 000000	6057		
ADDW13= 000000	6057		
ADDW14= 000000	6057		
ADDW15= 000000	6057		
ADDW2 = 000000	6057		
ADDW3 = 000000	6057		
ADDW4 = 000000	6057		
ADDW5 = 000000	6057		
ADDW6 = 000000	6057		
ADDW7 = 000000	6057		
ADDW8 = 000000	6057		
ADDW9 = 000000	6057		
ADEVCT= 000000	6057		
ADEVM = 000000	6057		
AENV = 000000	6057		
AENVM = 000000	6057		
AFATAL= 000000	6057		
AMADR1= 000000	6057		
AMADR2= 000000	6057		
AMADR3= 000000	6057		
AMADR4= 000000	6057		
AMAMS1= 000000	6057		
AMAMS2= 000000	6057		
AMAMS3= 000000	6057		
AMAMS4= 000000	6057		
AMSGAD= 000000	6057		
AMSGLG= 000000	6057		
AMSGTY= 000000	6057		
AMTYP1= 000000	6057		
AMTYP2= 000000	6057		
AMTYP3= 000000	6057		
AMTYP4= 000000	6057		
APASS = 000000	6057		
APRIOR= 000000	6057		
AROUND 001314	6101	6113#	6409
ASWREG= 000000	6057		
ATESTN= 000000	6057		
AUNIT = 000000	6057		
AUSWR = 000000	6057		
AVECT1= 000000	6057		
AVECT2= 000000	6057		
BIT0 = 000001	6032#	6159	
BIT00 = 000001	6032#		
BIT01 = 000002	6032#		
BIT02 = 000004	6032#		
BIT03 = 000010	6032#		
BIT04 = 000020	6032#		

.SPOWE	4229#	
.SRAND	4307#	
.SRDDE	3891#	
.SRDOC	3797#	
.SREAD	3395#	
.SR2AZ	4958#	
.SSAVE	3969#	
.SSB2D	4771#	
.SSB2O	4874#	
.SSCOP	2454#	
.SSIZE	4361#	
.SSUPR	4913#	
.STRAP	4073#	6031#
.STYPB	3287#	
.STYPD	3209#	
.STYPE	2985#	6031#
.STYPO	3112#	
.\$4OCA	972#	

. ABS. 004760 000

ERRORS DETECTED: 0

CNKafa,CNKafa/CRF/NL:TOC=CNMAC2.SML,CNKafa.P11
 RUN-TIME: 9 7 .5 SECONDS
 RUN-TIME RATIO: 49/17=2.7
 CORE USED: 32K (64 PAGES)