

# MINC-11

CHN TERMINATOR  
CVMNGA0

AH-E763A-MC

COPYRIGHT © 1978

FICHE 1 OF 1

DEC 1978

**digital**

MADE IN USA

IDENTIFICATION

B 1

SEQ 0001

PRODUCT CODE: AC-E762A-MC  
PRODUCT NAME: CVMNGAO MINC-11 CHAIN TERMINATOR MESSAGE  
DATE: AUGUST 1978  
MAINTAINER: DIAGNOSTIC ENGINEERING

COPYRIGHT (C) 1978  
DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.

THIS SOFTWARE IS FURNISHED UNDER A LICENSE FOR USE ONLY ON A SINGLE COMPUTER SYSTEM AND MAY BE COPIED ONLY WITH THE INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE, OR ANY OTHER COPIES THEREOF, MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON EXCEPT FOR USE ON SUCH SYSTEM AND TO ONE WHO AGREES TO THESE LICENSE TERMS. TITLE TO AND OWNERSHIP OF THE SOFTWARE SHALL AT ALL TIMES REMAIN IN DEC.

THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION.

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

TABLE OF CONTENTS

- 1.0 ABSTRACT
- 2.0 REQUIREMENTS
- 3.0 STARTING PROCEDURE
- 4.0 PROGRAM START
- 5.0 ERROR REPORTING

1.0 ABSTRACT  
-----

THE PURPOSE OF THE MINC-11 CHAIN TERMINATOR MESSAGE PROGRAM IS TO INFORM THE OPERATOR ABOUT THE COMPLETION OF RUNNING OF A 'DIAGNOSTIC CHAIN'. UNDER THE CURRENT 'CHAIN' OPERATION, THE END OF A 'CHAIN' OF DIAGNOSTIC PROGRAMS IS NOT CLEARLY VISABLE. THE GOAL OF THIS PROGRAM IS TO INFORM THE OPERATOR THAT A CHAIN HAS BEEN COMPLETED IN A MANOR THAT AVOIDS ANY DOUBT OR CONFUSION. BECAUSE THE PROGRAM DOES NO TESTING, THE PROGRAM SHOULD ONLY BE THE LAST PROGRAM IN A 'CHAIN'.

2.0 REQUIREMENT  
-----

1. ANY PDP11 COMPUTER WITH 4K OF MEMORY
2. DLV11 WITH I/O TERMINAL (LA36, VT100, ETC.)

3.0 STARTING PROCEDURE  
-----

1. THE PROGRAM WILL RESPOND BY TYPING THE PROGRAM TITLE.
2. THE PROGRAM WILL NOW TYPE A MESSAGE INFORMING THE OPERATOR THAT THE DIAGNOSTIC CHAIN HAS BEEN COMPLETED.

4.0 PROGRAM START  
-----

LOCATION 200 IS THE STARTING AND RESTART ADDRESS.

5.0 ERROR REPORTING  
-----

NO ERRORS ARE REPORTED.

10	BASIC DEFINITIONS
17	ACT11 HOOKS
19	APT PARAMETER BLOCK
20	COMMON TAGS
(2)	APT MAILBOX-ETABLE
(1)	ERROR POINTER TABLE
26	INITIAL START-UP,HOUSEKEEPING, AND DIALOGUE
28	INITIALIZE THE COMMON TAGS
29	TYPE PROGRAM NAME
32	END OF PASS ROUTINE
44	TYPE ROUTINE
45	APT COMMUNICATIONS ROUTINE
46	TRAP DECODER
(3)	TRAP TABLE

```
14
15
16 000200 000200 001246      .SPTTL  .SPTTL  .SPTTL  .SPTTL  .SPTTL  .SPTTL  .SPTTL  .SPTTL  .SPTTL  .SPTTL
17                                     .SBTTL  ACT11 HOOKS                                     ;START ADDRESS
(1)
(2)                                     ::*****
(1)                                     :HOOKS REQUIRED BY ACT11
(1)                                     $SVPC=.                                     ;SAVE PC
(1) 000046 000046                                     .=46
(1) 000046 001572 $ENDAD                                     ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
(1) 000052 000052                                     .=52
(1) 000052 000000 .WORD 0                                     ;;2)SET LOC.52 TO ZERO
(1) 000052 000204                                     .=$SVPC                                     ;; RESTORE PC
18                                     .=1000
19                                     .SPTTL  APT PARAMETER BLOCK
(1)
(2)                                     ::*****
(1)                                     :SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
(2)                                     ::*****
(1) 001000 001000                                     .$X=.                                     ;;SAVE CURRENT LOCATION
(1) 000024 000024                                     .=24                                     ;;SET POWER FAIL TO POINT TO START OF PROGRAM
(1) 000024 000200 200                                     ;;FOR APT START UP
(1) 000044 000044                                     .=44                                     ;;POINT TO APT INDIRECT ADDRESS PNTR.
(1) 000044 001000 $APTHDR                                     ;;POINT TO APT HEADER BLOCK
(1) 001000 001000 .=$X                                     ;;RESET LOCATION COUNTER
(2)                                     ::*****
(1)                                     :SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
(1)                                     :INTERFACE SPEC.
(1)
(1) $APTHD:
(1) 001000 000000 $HIBTS: .WORD 0                                     ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
(1) 001002 001164 $MBADR: .WORD $MAIL                                     ;;ADDRESS OF APT MAILBOX (BITS 0-15)
(1) 001004 000014 $STMT: .WORD 12.                                     ;;RUN TIM OF LONGEST TEST
(1) 001006 000005 $PASTM: .WORD 5.                                     ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
(1) 001010 000021 $UNITM: .WORD 17.                                     ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
(1) 001012 000031 .WORD $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
```

```

20      .SBTTL COMMON TAGS
(1)
(2)      ::*****
(1)      ::*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
(1)      ::*USED IN THE PROGRAM.
(1)
(1)      001100      .=1100
(1)      001100      000000      $CMTAG:      .;START OF COMMON TAGS
(1)      001102      000      $TSTNM: .WORD 0      .;CONTAINS THE TEST NUMBER
(1)      001103      000      $ERFLG: .BYTE 0      .;CONTAINS ERROR FLAG
(1)      001104      000000      $ICNT: .WORD 0      .;CONTAINS SUBTEST ITERATION COUNT
(1)      001106      000000      $LPADR: .WORD 0      .;CONTAINS SCOPE LOOP ADDRESS
(1)      001110      000000      $LPERR: .WORD 0      .;CONTAINS SCOPE RETURN FOR ERRORS
(1)      001112      000000      $ERTTL: .WORD 0      .;CONTAINS TOTAL ERRORS DETECTED
(1)      001114      000      $ITEMB: .BYTE 0      .;CONTAINS ITEM CONTROL BYTE
(1)      001115      001      $ERMAX: .BYTE 1      .;CONTAINS MAX. ERRORS PER TEST
(1)      001116      000000      $ERRPC: .WORD 0      .;CONTAINS PC OF LAST ERROR INSTRUCTION
(1)      001120      000000      $GDADR: .WORD 0      .;CONTAINS ADDRESS OF 'GOOD' DATA
(1)      001122      000000      $BDADR: .WORD 0      .;CONTAINS ADDRESS OF 'BAD' DATA
(1)      001124      000000      $GDDAT: .WORD 0      .;CONTAINS 'GOOD' DATA
(1)      001126      000000      $BDDAT: .WORD 0      .;CONTAINS 'BAD' DATA
(1)      001130      000000      .WORD 0      .;RESERVED--NOT TO BE USED
(1)      001132      000000      .WORD 0
(1)      001134      000      $AUTOB: .BYTE 0      .;AUTOMATIC MODE INDICATOR
(1)      001135      000      $INTAG: .BYTE 0      .;INTERRUPT MODE INDICATOR
(1)      001136      000000      .WORD 0
(1)      001140      177570      SWR: .WORD DSWR      .;ADDRESS OF SWITCH REGISTER
(1)      001142      177570      DISPLAY: .WORD DDISP      .;ADDRESS OF DISPLAY REGISTER
(1)      001144      177560      $TKS: . 177560      .;TTY KBD STATUS
(1)      001146      177562      $TKB: . 177562      .;TTY KBD BUFFER
(1)      001150      177564      $TPS: . 177564      .;TTY PRINTER STATUS REG. ADDRESS
(1)      001152      177566      $TPB: . 177566      .;TTY PRINTER BUFFER REG. ADDRESS
(1)      001154      000      $NULL: .BYTE 0      .;CONTAINS NULL CHARACTER FOR FILLS
(1)      001155      002      $FILLS: .BYTE 2      .;CONTAINS # OF FILLER CHARACTERS REQUIRED
(1)      001156      012      $FILLC: .BYTE 12      .;INSERT FILL CHARS. AFTER A 'LINE FEED'
(1)      001157      000      $TPFLG: .BYTE 0      .;'TERMINAL AVAILABLE' FLAG (BIT<07>=0=YES)
(1)      001160      077      $QUES: .ASCII /?/      .;QUESTION MARK
(1)      001161      015      $CRLF: .ASCII <15>      .;CARRIAGE RETURN
(1)      001162      000012      $LF: .ASCIZ <12>      .;LINE FEED
(2)      ::*****
(2)      .SBTTL APT MAILBOX-ETABLE
(2)
(3)      ::*****
(2)      .EVEN
(2)      001164      $MAIL:      .;APT MAILBOX
(2)      001164      000000      $MSGTY: .WORD AMSGTY      .;MESSAGE TYPE CODE
(2)      001166      000000      $FATAL: .WORD AFATAL      .;FATAL ERROR NUMBER
(2)      001170      000000      $TESTN: .WORD ATESTN      .;TEST NUMBER
(2)      001172      000000      $PASS: .WORD APASS      .;PASS COUNT
(2)      001174      000000      $DEVCT: .WORD ADEVCT      .;DEVICE COUNT
(2)      001176      000000      $UNIT: .WORD AUNIT      .;I/O UNIT NUMBER
(2)      001200      000000      $MSGAD: .WORD AMSGAD      .;MESSAGE ADDRESS
(2)      001202      000000      $MSGLG: .WORD AMGLG      .;MESSAGE LENGTH

```

(2)	001204		\$ETABLE:		::APT ENVIRONMENT TABLE
(2)	001204	000	\$ENV:	.BYTE AENV	::ENVIRONMENT BYTE
(2)	001205	000	\$ENVM:	.BYTE AENVM	::ENVIRONMENT MODE BITS
(2)	001206	000000	\$SWREG:	.WORD ASWREG	::APT SWITCH REGISTER
(2)	001210	000000	\$USWR:	.WORD AUSWR	::USER SWITCHES
(2)	001212	000000	\$CPUOP:	.WORD ACPUOP	::CPU TYPE,OPTIONS
(2)			:*		BITS 15-11=CPU TYPE
(2)			:*		11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
(2)			:*		11/70=06,PDQ=07,Q=10
(2)			:*		BIT 10=REAL TIME CLOCK
(2)			:*		BIT 9=FLOATING POINT PROCESSOR
(2)			:*		BIT 8=MEMORY MANAGEMENT
(2)	001214	000	\$MAMS1:	.BYTE AMAMS1	::HIGH ADDRESS,M.S. BYTE
(2)	001215	000	\$MTYP1:	.BYTE AMTYP1	::MEM. TYPE,BLK#1
(2)			:*		MEM.TYPE BYTE -- (HIGH BYTE)
(2)			:*		900 NSEC CORE=001
(2)			:*		300 NSEC BIPOLAR=002
(2)			:*		500 NSEC MOS=003
(2)	001216	000000	\$MADR1:	.WORD AMADR1	::HIGH ADDRESS,BLK#1
(2)			:*		MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF 'TYPE' ABOVE
(2)	001220	000	\$MAMS2:	.BYTE AMAMS2	::HIGH ADDRESS,M.S. BYTE
(2)	001221	000	\$MTYP2:	.BYTE AMTYP2	::MEM. TYPE,BLK#2
(2)	001222	000000	\$MADR2:	.WORD AMADR2	::MEM.LAST ADDRESS,BLK#2
(2)	001224	000	\$MAMS3:	.BYTE AMAMS3	::HIGH ADDRESS,M.S.BYTE
(2)	001225	000	\$MTYP3:	.BYTE AMTYP3	::MEM. TYPE,BLK#3
(2)	001226	000000	\$MADR3:	.WORD AMADR3	::MEM.LAST ADDRESS,BLK#3
(2)	001230	000	\$MAMS4:	.BYTE AMAMS4	::HIGH ADDRESS,M.S.BYTE
(2)	001231	000	\$MTYP4:	.BYTE AMTYP4	::MEM. TYPE,BLK#4
(2)	001232	000000	\$MADR4:	.WORD AMADR4	::MEM.LAST ADDRESS,BLK#4
(2)	001234	000000	\$VECT1:	.WORD AVECT1	::INTERRUPT VECTOR#1,BUS PRIORITY#1
(2)	001236	000000	\$VECT2:	.WORD AVECT2	::INTERRUPT VECTOR#2BUS PRIORITY#2
(2)	001240	000000	\$BASE:	.WORD ABASE	::BASE ADDRESS OF EQUIPMENT UNDER TEST
(2)	001242	000000	\$DEV1:	.WORD ADEV1	::DEVICE MAP
(2)	001244	000000	\$CDW1:	.WORD ACDW1	::CONTROLLER DESCRIPTION WORD#1
(2)	001246		\$ETEND:		
(2)			.MEXIT		



(1)  
(1)  
(1)  
(1)  
(1)  
(1)  
(1)  
(1)  
(1)  
(1)  
(1)  
(1)  
(1)  
(1)  
(1)  
21  
22  
23

001246

.SBTTL ERROR POINTER TABLE

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

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

\$ERRTB:

;NO ERRORS ARE REPORTED

```

26      .SBTTL      INITIAL START-UP,HOUSEKEEPING, AND DIALOGUE
27      001246  000005  BEGIN:  RESET
28      .SBTTL      INITIALIZE THE COMMON TAGS
(1)      ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
(1)      001250  012706  001100  MOV      #CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
(1)      001254  005026          CLR      (R6)+          ;;CLEAR MEMORY LOCATION
(1)      001256  022706  001140  CMP      #SWR,R6      ;;DONE?
(1)      001262  001374          BNE      -6            ;;LOOP BACK IF NO
(1)      001264  012706  001100  MOV      #STACK,SP     ;;SETUP THE STACK POINTER
(1)      ;;INITIALIZE A FEW VECTORS
(1)      001270  012737  002470  000034  MOV      #STRAP,@TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
(1)      001276  012737  000340  000036  MOV      #340,@TRAPVEC+2;LEVEL 7
(1)      001304  013737  001552  001544  MOV      $ENDCT,$EOPCT  ;;SETUP END-OF-PROGRAM COUNTER
(2)      ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
(2)      ;;EQUAL TO A '-1', SETUP FOR A SOFTWARE SWITCH REGISTER.
(2)      001312  013746  000004          MOV      @ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
(2)      001316  012737  001352  000004  MOV      #64$,@ERRVEC  ;;SET UP ERROR VECTOR
(2)      001324  012737  177570  001140  MOV      #DSWR,SWR     ;;SETUP FOR A HARDWARE SWICH REGISTER
(2)      001332  012737  177570  001142  MOV      #DDISP,DISPLAY ;;AND A HARDWARE DISPLAY REGISTER
(2)      001340  022777  177777  177572  CMP      #-1,@SWR     ;;TRY TO REFERENCE HARDWARE SWR
(2)      001346  001012          BNE      66$         ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
(2)      ;;AND THE HARDWARE SWR IS NOT = -1
(2)      001350  000403          BR      65$         ;;BRANCH IF NO TIMEOUT
(2)      001352  012716  001360          64$:  MOV      #65$,(SP)   ;;SET UP FOR TRAP RETURN
(2)      001356  000002          RTI
(2)      001360  012737  000176  001140  65$:  MOV      #SWREG,SWR   ;;POINT TO SOFTWARE SWR
(2)      001366  012737  000174  001142  MOV      #DISPREG,DISPLAY
(2)      001374  012637  000004          66$:  MOV      (SP)+,@ERRVEC ;;RESTORE ERROR VECTOR
(1)
(2)      001400  005037  001172          CLR      $PASS        ;;CLEAR PASS COUNT
(2)      001404  132737  000200  001205  BITB    #APTSIZE,$ENVM ;;TEST USER SIZE UNDER APT
(2)      001412  001403          BEQ     67$         ;;YES,USE NON-APT SWITCH
(2)      001414  012737  001206  001140  MOV     #SSWREG,SWR   ;;NO,USE APT SWITCH REGISTER
(2)      001422          67$:
29      .SBTTL      TYPE PROGRAM NAME
(1)      ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
(1)      001422  005227  177777          INC     #-1          ;;FIRST TIME?
(1)      001426  001035          BNE     68$         ;;BRANCH IF NO
(1)      001430  022737  001572  000042  CMP     #SENDAD,@#42  ;;ACT-11?
(1)      001436  001431          BEQ     68$         ;;BRANCH IF YES
(1)      001440  104401  001446          TYPE   ,69$        ;;TYPE ASCIZ STRING
(1)      001444  000426          BR     68$         ;;GET OVER THE ASCIZ
(1)      ;;69$: .ASCIZ <CRLF>#CVMNG-A MINC-11 CHAIN TERMINATOR PROGRAM#<CRLF>
(1)      001522          68$:
30      001522  104401  001612          TYPE   ,SETMNC      ;TELL OPER. ABOUT COMPLETING THE CHAIN

```

```

32      .SBTTL  END OF PASS ROUTINE
(1)
(2)      ;:*****
(1)      ;:*INCREMENT THE PASS NUMBER ($PASS)
(1)      ;:*DING THE TTY BELL ON END OF PROGRAM
(1)      ;:*IF THERES A MONITOR GO TO IT
(1)      ;:*IF THERE ISN'T JUMP TO 200
(1)      ;:*WHEN A TERMINAL DOES NOT HAVE A BELL LOCATION $BELL CAN BE CHANGED
(1)      ;:*TO A PRINTING CHARACTER TO INDICATE AN 'END OF PASS'.
(1)      ;:*IE. 177501 WOULD PRINT AN 'A'
(1)
(1)      $EOP:
(2)      001526 000240      NOP
(1)      001530 005237 001172      INC      $PASS      ;;INCREMENT THE PASS NUMBER
(1)      001534 042737 100000 001172      BIC      #100000,$PASS  ;;DON'T ALLOW A NEG. NUMBER
(1)      001542 005327      DEC      (PC)+      ;;LOOP?
(1)      001544 000001      $EOPCT: .WORD      1
(1)      001546 003015      BGT      $DOAGN      ;;YES
(1)      001550 012737      MOV      (PC)+,@(PC)+  ;;RESTORE COUNTER
(1)      001552 000001      $ENDCT: .WORD      1
(1)      001554 001544      $EOPCT
(1)      001556 104401 001606      TYPE      , $BELL      ;;RING A BELL
(1)      001562 013700 000042      $GET42: MOV      @#42,R0  ;;GET MONITOR ADDRESS
(1)      001566 001405      BEQ      $DOAGN      ;;BRANCH IF NO MONITOR
(1)      001570 000005      RESET
(1)      001572 004710      $ENDAD: JSR      PC,(R0)  ;;GO TO MONITOR
(1)      001574 000240      NOP      ;;SAVE ROOM
(1)      001576 000240      NOP      ;;FOR
(1)      001600 000240      NOP      ;;ACT11
(1)      001602
(1)      001602 000137 000200      $DOAGN:  JMP      @#200      ;;RETURN
(1)      001606      207      $BELL:  .BYTE      207
(1)      001607      377      377      000  $ENULL: .BYTE      -1,-1,0  ;;NULL CHARACTER STRING
  
```



```

(1) 002020 001003          BNE      60$          ;;YES,SKIP TYPE OUT
(1) 002022 112046          2$:   MOVB    (R0)+, -(SP)  ;;PUSH CHARACTER TO BE TYPED ONTO STACK
(1) 002024 001005          BNE      4$          ;;BR IF IT ISN'T THE TERMINATOR
(1) 002026 005726          TST     (SP)+        ;;IF TERMINATOR POP IT OFF THE STACK
(1) 002030 012600          60$:  MOV     (SP)+, R0   ;;RESTORE R0
(1) 002032 062716 000002  3$:   ADD     #2, (SP)    ;;ADJUST RETURN PC
(1) 002036 000002          RTI                    ;;RETURN
(1) 002040 122716 000011  4$:   CMPB   #HT, (SP)    ;;BRANCH IF <HT>
(1) 002044 001430          BEQ     8$          ;;BRANCH IF NOT <CRLF>
(1) 002046 122716 000200  CMPB   #CRLF, (SP)
(1) 002052 001006          BNE     5$          ;;POP <CR><LF> EQUIV
(1) 002054 005726          TST     (SP)+        ;;TYPE A CR AND LF
(1) 002056 104401          TYPE
(1) 002060 001161          $CRLF
(1) 002062 105037 002216  CLRB   $CHARCNT    ;;CLEAR CHARACTER COUNT
(1) 002066 000755          BR      2$          ;;GET NEXT CHARACTER
(1) 002070 004737 002152  5$:   JSR    PC, $TYPEC   ;;GO TYPE THIS CHARACTER
(1) 002074 123726 001156  6$:   CMPB   $FILLC, (SP)+  ;;IS IT TIME FOR FILLER CHARS.?
(1) 002100 001350          BNE     2$          ;;IF NO GO GET NEXT CHAR.
(1) 002102 013746 001154  MOV     $NULL, -(SP)  ;;GET # OF FILLER CHARS. NEEDED
(1)                                ;;AND THE NULL CHAR.
(1) 002106 105366 000001  7$:   DECB   1(SP)        ;;DOES A NULL NEED TO BE TYPED?
(1) 002112 002770          BLT     6$          ;;BR IF NO--GO POP THE NULL OFF OF STACK
(1) 002114 004737 002152  JSR    PC, $TYPEC   ;;GO TYPE A NULL
(1) 002120 105337 002216  DECB   $CHARCNT    ;;DO NOT COUNT AS A COUNT
(1) 002124 000770          BR      7$          ;;LOOP
(1)
(1)                                ;HORIZONTAL TAB PROCESSOR
(1)
(1) 002126 112716 000040  8$:   MOVB   #' , (SP)   ;;REPLACE TAB WITH SPACE
(1) 002132 004737 002152  9$:   JSR    PC, $TYPEC   ;;TYPE A SPACE
(1) 002136 132737 000007 002216  BITB   #7, $CHARCNT  ;;BRANCH IF NOT AT
(1) 002144 001372          BNE     9$          ;;TAB STOP
(1) 002146 005726          TST     (SP)+        ;;POP SPACE OFF STACK
(1) 002150 000724          BR      2$          ;;GET NEXT CHARACTER
(1) 002152 105777 176772  $TYPEC: TSTB   @ $TPS   ;;WAIT UNTIL PRINTER IS READY
(1) 002156 100375          BPL     $TYPEC
(1) 002160 116677 000002 176764  MOVB   2(SP), @ $TPB  ;;LOAD CHAR TO BE TYPED INTO DATA REG.
(1) 002166 122766 000015 000002  CMPB   #CR, 2(SP)    ;;IS CHARACTER A CARRIAGE RETURN?
(1) 002174 001003          BNE     1$          ;;BRANCH IF NO
(1) 002176 105037 002216  CLRB   $CHARCNT    ;;YES--CLEAR CHARACTER COUNT
(1) 002202 000406          BR      $TYPEX     ;;EXIT
(1) 002204 122766 000012 000002  1$:   CMPB   #LF, 2(SP)  ;;IS CHARACTER A LINE FEED?
(1) 002212 001402          BEQ     $TYPEX     ;;BRANCH IF YES
(1) 002214 105227          INCB   (PC)+        ;;COUNT THE CHARACTER
(1) 002216 000000          $CHARCNT: .WORD 0  ;;CHARACTER COUNT STORAGE
(1) 002220 000207          $TYPEX: RTS      PC
(1)
(1) 45
(1)                                .SBTTL  APT COMMUNICATIONS ROUTINE
(2)
(1) 002222 112737 000001 002466  $ATY1: MOVB   #1, $FFLG  ;;TO REPORT FATAL ERROR
(1) 002230 112737 000001 002464  $ATY3: MOVB   #1, $MFLG  ;;TO TYPE A MESSAGE
(1) 002236 000403          BR      $ATYC

```

```
(1) 002240 112737 000001 002466 $ATY4: MOV      #1,$FFLG      ;;TO ONLY REPORT FATAL ERROR
(2) 002246                                $ATYC:
(3) 002246 010046                                MOV      R0,-(SP)      ;;PUSH R0 ON STACK
(3) 002250 010146                                MOV      R1,-(SP)      ;;PUSH R1 ON STACK
(1) 002252 105737 002464                                TSTB    $MFLG          ;;SHOULD TYPE A MESSAGE?
(1) 002256 001450                                BEQ      5$            ;;IF NOT: BR
(1) 002260 122737 000001 001204                                CMPB    #APTENV,$ENV   ;;OPERATING UNDER APT?
(1) 002266 001031                                BNE     3$            ;;IF NOT: BR
(1) 002270 132737 000100 001205                                BITB    #APTPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
(1) 002276 001425                                BEQ     3$            ;;IF NOT: BR
(1) 002300 017600 000004                                MOV     @4(SP),R0      ;;GET MESSAGE ADDR.
(1) 002304 062766 000002 000004                                ADD     #2,4(SP)       ;;BUMP RETURN ADDR.
(1) 002312 005737 001164 1$: TST     $MSGTYPE       ;;SEE IF DONE W/ LAST XMISSION?
(1) 002316 001375                                BNE     1$            ;;IF NOT: WAIT
(1) 002320 010037 001200                                MOV     R0,$MSGAD      ;;PUT ADDR IN MAILBOX
(1) 002324 105720 2$: TSTB    (R0)+          ;;FIND END OF MESSAGE
(1) 002326 001376                                BNE     2$
(1) 002330 163700 001200                                SUB     $MSGAD,R0      ;;SUB START OF MESSAGE
(1) 002334 006200                                ASR     R0              ;;GET MESSAGE LNTH IN WORDS
(1) 002336 010037 001202                                MOV     R0,$MSGGLT     ;;PUT LENGTH IN MAILBOX
(1) 002342 012737 000004 001164                                MOV     #4,$MSGTYPE   ;;TELL APT TO TAKE MSG.
(1) 002350 000413                                BR      5$
(1) 002352 017637 000004 002376 3$: MOV     @4(SP),4$      ;;PUT MSG ADDR IN JSR LINKAGE
(1) 002360 062766 000002 000004                                ADD     #2,4(SP)       ;;BUMP RETURN ADDRESS
(3) 002366 013746 177776                                MOV     177776,-(SP)  ;;PUSH 177776 ON STACK
(1) 002372 004737 001740                                JSR     PC,$TYPE      ;;CALL TYPE MACRO
(1) 002376 000000 4$: .WORD    0
(1) 002400 5$:
(1) 002400 105737 002466 10$: TSTB    $FFLG          ;;SHOULD REPORT FATAL ERROR?
(1) 002404 001416                                BEQ     12$           ;;IF NOT: BR
(1) 002406 005737 001204                                TST     $ENV           ;;RUNNING UNDER APT?
(1) 002412 001413                                BEQ     12$           ;;IF NOT: BR
(1) 002414 005737 001164 11$: TST     $MSGTYPE       ;;FINISHED LAST MESSAGE?
(1) 002420 001375                                BNE     11$           ;;IF NOT: WAIT
(1) 002422 017637 000004 001166                                MOV     @4(SP),$FATAL ;;GET ERROR #
(1) 002430 062766 000002 000004                                ADD     #2,4(SP)       ;;BUMP RETURN ADDR.
(1) 002436 005237 001164                                INC     $MSGTYPE       ;;TELL APT TO TAKE ERROR
(1) 002442 105037 002466 12$: CLRB    $FFLG          ;;CLEAR FATAL FLAG
(1) 002446 105037 002465                                CLRB    $LFLG          ;;CLEAR LOG FLAG
(1) 002452 105037 002464                                CLRB    $MFLG          ;;CLEAR MESSAGE FLAG
(3) 002456 012601                                MOV     (SP)+,R1      ;;POP STACK INTO R1
(3) 002460 012600                                MOV     (SP)+,R0      ;;POP STACK INTO R0
(1) 002462 000207                                RTS     PC             ;;RETURN
(1) 002464 000                                $MFLG: .BYTE    0      ;;MESSG. FLAG
(1) 002465 000                                $LFLG: .BYTE    0      ;;LOG FLAG
(1) 002466 000                                $FFLG: .BYTE    0      ;;FATAL FLAG
(1) 002470 .EVEN
(1) 000200 APTSIZE=200
(1) 000001 APTENV=001
(1) 000100 APTPOOL=100
(1) 000040 APTCSUP=040
46 .SBTTL TRAP DECODER
(1)
(2) ;;*****
```

```

(1) ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE 'TRAP' INSTRUCTION
(1) ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
(1) ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
(1) ;*GO TO THAT ROUTINE.
(1)
(1) 002470 010046 $TRAP: MOV R0,-(SP) ;;SAVE R0
(1) 002472 016600 000002 MOV 2(SP),R0 ;;GET TRAP ADDRESS
(1) 002476 005740 TST -(R0) ;;BACKUP BY 2
(1) 002500 111000 MOVB (R0),R0 ;;GET RIGHT BYTE OF TRAP
(1) 002502 006300 ASL R0 ;;POSITION FOR INDEXING
(1) 002504 016000 002524 MOV $TRPAD(R0),R0 ;;INDEX TO TABLE
(1) 002510 000200 RTS R0 ;;GO TO ROUTINE
(1)
(1)
(1) ;;THIS IS USE TO HANDLE THE 'GETPRI' MACRO
(1)
(1) 002512 011646 $TRAP2: MOV (SP),-(SP) ;;MOVE THE PC DOWN
(1) 002514 016666 000004 000002 MOV 4(SP),2(SP) ;;MOVE THE PSW DOWN
(1) 002522 000002 RTI ;;RESTORE THE PSW
(1)
(3) .SBTTL TRAP TABLE
(3)
(3) ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
(3) ;*BY THE 'TRAP' INSTRUCTION.
(3)
(3) : ROUTINE
(3) : -----
(3) $TRPAD: .WORD $TRAP2
(3) 002524 002512 $TYPE ;;CALL=TYPE TRAP+1(104401) TTY TYPEOUT ROUTINE
(3) 002526 001740
(1)
(1) 47 000001 .END
  
```

ABASE = 000000	20	
ACDW1 = 000000	20	
ACDW2 = 000000	20	
ACPUOP= 000000	20	
ADDW0 = 000000	20	
ADDW1 = 000000	20	
ADDW10= 000000	20	
ADDW11= 000000	20	
ADDW12= 000000	20	
ADDW13= 000000	20	
ADDW14= 000000	20	
ADDW15= 000000	20	
ADDW2 = 000000	20	
ADDW3 = 000000	20	
ADDW4 = 000000	20	
ADDW5 = 000000	20	
ADDW6 = 000000	20	
ADDW7 = 000000	20	
ADDW8 = 000000	20	
ADDW9 = 000000	20	
ADEVCT= 000000	20	
ADEVPM = 000000	20	
AENV = 000000	20	
AENVPM = 000000	20	
AFATAL= 000000	20	
AMADR1= 000000	20	
AMADR2= 000000	20	
AMADR3= 000000	20	
AMADR4= 000000	20	
AMAMS1= 000000	20	
AMAMS2= 000000	20	
AMAMS3= 000000	20	
AMAMS4= 000000	20	
AMSGAD= 000000	20	
AMSGLG= 000000	20	
AMSGTY= 000000	20	
AMTYP1= 000000	20	
AMTYP2= 000000	20	
AMTYP3= 000000	20	
AMTYP4= 000000	20	
APASS = 000000	20	
APRIOR= 000000	20	
APTCU= 000040	44	45#
APTENV= 000001	44	45#
APTSIZ= 000200	28	45#
APTSPO= 000100	44	45#
ASWREG= 000000	20	
ATESTN= 000000	20	
ALUNIT = 000000	20	
AUSWR = 000000	20	
AVECT1= 000000	20	
AVECT2= 000000	20	
BEGIN 001246	16	27#
BITO = 000001	10#	



BIT00 = 000001	10#				
BIT01 = 000002	10#				
BIT02 = 000004	10#				
BIT03 = 000010	10#				
BIT04 = 000020	10#				
BIT05 = 000040	10#				
BIT06 = 000100	10#				
BIT07 = 000200	10#				
BIT08 = 000400	10#				
BIT09 = 001000	10#				
BIT1 = 000002	10#				
BIT10 = 002000	10#				
BIT11 = 004000	10#				
BIT12 = 010000	10#				
BIT13 = 020000	10#				
BIT14 = 040000	10#				
BIT15 = 100000	10#				
BIT2 = 000004	10#				
BIT3 = 000010	10#				
BIT4 = 000020	10#				
BIT5 = 000040	10#				
BIT6 = 000100	10#				
BIT7 = 000200	10#				
BIT8 = 000400	10#				
BIT9 = 001000	10#				
BPTVEC= 000014	10#				
CR = 000015	10#	44			
CRLF = 000200	10#	29	44		
DDISP = 177570	10#	20	28		
DISPLA 001142	20#	28*			
DISPRE= 000174	28	42#			
DSWR = 177570	10#	20	28		
EMTVEC= 000030	10#				
ERRVEC= 000004	10#	28*			
GNS = ***** U	29	46			
HT = 000011	10#	44			
IOTVEC= 000020	10#				
LF = 000012	10#	44			
PC =%000007	10#	32*	44*	45*	
PIRQ = 177772	10#				
PIRQVE= 000240	10#				
PRO = 000000	10#				
PR1 = 000040	10#				
PR2 = 000100	10#				
PR3 = 000140	10#				
PR4 = 000200	10#				
PR5 = 000240	10#				
PR6 = 000300	10#				
PR7 = 000340	10#				
PS = 177776	10#				
PSW = 177776	10#				
PWRVEC= 000024	10#				
RESVEC= 000010	10#				
RO =%000000	10#	32*	44*	45*	46*

R1	=%000001	10#	45*			
R2	=%000002	10#				
R3	=%000003	10#				
R4	=%000004	10#				
R5	=%000005	10#				
R6	=%000006	10#	28*			
R7	=%000007	10#				
SETMNC	001612	30	35#			
SP	=%000006	10#	28*	44*	45*	46*
STACK	= 001100	10#	28			
STKLMT	= 177774	10#				
SWR	001140	20#	28*			
SWREG	= 000176	28	41#			
SW0	= 000001	10#				
SW00	= 000001	10#				
SW01	= 000002	10#				
SW02	= 000004	10#				
SW03	= 000010	10#				
SW04	= 000020	10#				
SW05	= 000040	10#				
SW06	= 000100	10#				
SW07	= 000200	10#				
SW08	= 000400	10#				
SW09	= 001000	10#				
SW1	= 000002	10#				
SW10	= 002000	10#				
SW11	= 004000	10#				
SW12	= 010000	10#				
SW13	= 020000	10#				
SW14	= 040000	10#				
SW15	= 100000	10#				
SW2	= 000004	10#				
SW3	= 000010	10#				
SW4	= 000020	10#				
SW5	= 000040	10#				
SW6	= 000100	10#				
SW7	= 000200	10#				
SW8	= 000400	10#				
SW9	= 001000	10#				
TBITVE	= 000014	10#				
TKVEC	= 000060	10#				
TPVEC	= 000064	10#				
TRAPVE	= 000034	10#	28*			
TRTVEC	= 000014	10#				
TYPE	= 104401	29	30	32	44	46#
\$APTHD	001000	19#				
\$ASTAT	= ***** U	45				
\$ATYC	002246	45#				
\$ATY1	002222	45#				
\$ATY3	002230	44	45#			
\$ATY4	002240	45#				
\$AUTOB	001134	20#				
\$BASE	001240	20#				
\$BDADR	001122	20#				

\$BDDAT	001126	20#			
\$BELL	001606	32#			
\$CDW1	001244	20#			
\$CHARC	002216	44#*			
\$CKSWR=	***** U	46			
\$CMTAG	001100	20#	28		
\$CM3 =	000000	20#			
\$CPUOP	001212	20#			
\$CRLF	001161	20#	44		
\$DEVCT	001174	20#			
\$DEVM	001242	20#			
\$DOAGN	001602	32#			
\$ENDAD	001572	17	29	32#	
\$ENDCT	001552	28	32#		
\$ENULL	001607	32#			
\$ENV	001204	20#	44	45	
\$ENVM	001205	20#	28	44	45
\$EOP	001526	32#			
\$EOPCT	001544	28*	32#		
\$ERFLG	001103	20#			
\$ERMAX	001115	20#			
\$ERRPC	001116	20#			
\$ERRTB	001246	20#			
\$ERTTL	001112	20#			
\$ETABL	001204	20#			
\$ETEND	001246	19	20#		
\$FATAL	001166	20#	45*		
\$FFLG	002466	45#*			
\$FILLC	001156	20#	44		
\$FILLS	001155	20#	44		
\$GDADR	001120	20#			
\$GDDAT	001124	20#			
\$GET42	001562	32#			
\$GTSWR=	***** U	46			
\$HD =	000003	9			
\$HIBTS	001000	19#			
\$ICNT	001104	20#			
\$INTAG	001135	20#			
\$ITEMB	001114	20#			
\$LF	001162	20#	44		
\$LFLG	002465	45#*			
\$LPADR	001106	20#			
\$LPERR	001110	20#			
\$MADR1	001216	20#			
\$MADR2	001222	20#			
\$MADR3	001226	20#			
\$MADR4	001232	20#			
\$MAIL	001164	19	20#	28	44
\$MAMS1	001214	20#			
\$MAMS2	001220	20#			
\$MAMS3	001224	20#			
\$MAMS4	001230	20#			
\$MBADR	001002	19#			
\$MFLG	002464	45#*			





.\$SCOP	6#	
.\$SPAC	7#	
.\$SWDO	7#	
.\$STRAP	7#	46
.\$TYPB	6#	
.\$TYPD	8#	
.\$TYPE	7#	44
.\$TYPO	6#	



.SBTTL	10	17	19	20	26	28	29	32	44	45	46
.TITLE	9										
.WORD	12	17	19	20	32	44	45	46			

ERRORS DETECTED: 0



CVMNG-A MINC-11 CHAIN TERMINATOR PROGRAM  
CVMNGA.P11

MACY11 27(654) <sup>L 2</sup>19-SEP-78 09:05 PAGE 4-13

SEQ 0024

\*CVMNGA,CVMNGA/CRF=CVMNGA  
RUN-TIME: 20 2 0 SECONDS  
CORE USED: 24K