```
.REM %
```

IDENTIFICATION

PRODUCT CODE:	AC-8954E-MC
PRODUCT NAME:	CZM9AE0 BOOTSTRAP/TERMINATOR (M9301, M9400)
PROGRAM DATE:	APRIL, 1979
MAINTAINER:	DIAGNOSTIC ENGINEERING

```
                        ****************************************
                        *                                      *
                        * SUMMARY OF OPERATING INSTRUCTIONS    *
                        *                                      *
                        ****************************************


THE FOLLOWING PROCEDURE CAN BE USED TO RUN THIS DIAGNOSTIC
IN A DEVICE VERIFICATION MODE.  IF THE PROGRAM DOES NOT
RUN SUCCESSFULLY CONSULT THE FOLLOWING DOCUMENT FOR ASSISTANCE.

OPERATING PROCEDURE:

        1.  LOAD THE PROGRAM USING NORMAL PROCEDURES.

        2.  LOAD ADDRESS 200

        3.  SET SWITCH REGISTER TO SELECT THE PROPER
            VERSION OF THE ROM UNDER TEST.
            (SEE INSTRUCTIONS IF A SOFTWARE SWITCH REGISTER
            IS TO BE USED.)

        4.  PRESS START

        5.  THE PROGRAM SHOULD TAKE ABOUT 1 SEC TO
            COMPLETE THE TEST AND PRINT: "END OF TEST".

        6.  IF THE PROGRAM DOES NOT RUN AS DESCRIBED
            ABOVE, CONSULT THE FULL OPERATING INSTRUCTIONS
            WHICH FOLLOW.


                * *  CAUTION  * *

        BECAUSE THE CONTENTS READ FROM LOCATION 773024
        OF THE M9301 OPTION IS CONFIGURATION DEPENDANT(SWITCH
        REGISTER DEPENDANT), THIS LOCATION IS NOT INCLUDED IN
        THE DATA CHECK.
        THIS LOCATION CAN BE VERIFIED BY EXAMINING IT OR BY USING
        THE ALTERNATE STARTING ADDRESS(SEE SECTION 2.1.3) TO
        PRINT OUT THIS LOCATION.
```

```
                        1.0     GENERAL PROGRAM INFORMATION
                                ------- ------- -----------

                        1.1     PROGRAM PURPOSE
                                ------- -------

                                THIS DIAGNOSTIC PROGRAM IS INTENDED TO VERIFY THE
                                ROM CONTENTS OF THE ROM BOOTSTRAP MODULES. THE PROGRAM
                                COMPUTES AND CHECKS A CYCLIC REDUNDANCY CHARACTER
                                AND A LONGITUDINAL PARITY CHARACTER FOR THE CONTENTS
                                OF THE ROM STORAGE AVAILABLE IN AN M9301 OR M9400 MODULE.

                                A SEPARATE ROUTINE INCLUDED ALLOWS THE USER TO TYPE
                                THE CONTENTS OF THE ROM STORAGE ON THE TELETYPE AS
                                AN AID TO DEBUGGING.

                        1.2     SYSTEM REQUIREMENTS
                                ------ ------------

                        1.2.1   HARDWARE
                                --------

                                PDP/11 PROCESSOR
                                TELETYPE OR EQUIVALENT
                                4K OF MEMORY
                                M9301 OR M9400 MODULE

                        1.2.2   SOFTWARE
                                --------

                                THIS PROGRAM IS WRITTEN TO BE RUN AS A STAND-ALONE PROGRAM.
                                HOWEVER, THE PROGRAM IS DESIGNED TO RUN UNDER AUTOMATED
                                PRODUCT TEST SYSTEM (APT) IN ALL THREE MODES.

                                THE PROGRAM CAN ALSO BE RUN UNDER THE ACT 11 MONITOR.

                        1.3     RELATED DOCUMENTS AND STANDARDS
                                ------- --------- --- ---------

                                DIAGNOSTIC ENGINEERING STANDARDS AND CONVENTIONS PROGRAMMING PRACTICES
                                        DOCUMENT NO. 175-003-009-00

                                APT INTERFACE SPECIFICATION, REV. 13

                        1.4     DIAGNOSTIC HIERARCHY PREREQUISITES
                                ---------- --------- -------------

                                NONE, HOWEVER THE CPU IS ASSUMED TO BE FUNCTIONING

                        1.5     FAILURE ASSUMPTIONS
                                ------- -----------

                                THE PROCESSOR IS ASSUMED TO BE FUNCTIONING PROPERLY.
```

2.0 OPERATING INSTRUCTIONS

2.1 LOADING AND STARTING PROCEDURES

2.1.1 LOADING

USE NORMAL PROCEDURES FOR LOADING DIAGNOSTIC PROGRAMS.

2.1.2 NORMAL START

1. LOAD SOFTWARE SWITCH REGISTER (IF USED) TO SELECT THE ROM VERSION UNDER TEST. (SEE 2.3)

2. LOAD ADDRESS 200

3. SET HARDWARE SWITCH REGISTER (IF AVAILABLE) TO SELECT THE ROM VERSION UNDER TEST (SEE 2.3).

4. START

2.1.3 OPTIONAL START

THE OPTIONAL STARTING ADDRESS IS USED TO TYPE OUT THE CONTENTS OF THE ROM FOR USE IN VISUAL VERIFICATION OR AS A DEBUGGING TOOL.

USE THE SAME PROCEDURE AS A NORMAL START EXCEPT USE ADDRESS 210 IN STEP 2.

2.2 SPECIAL ENVIRONMENTS

THIS PROGRAM IS WRITTEN TO COMPLY WITH ALL THE REQUIREMENTS OF THE APT INTERFACE SPECIFICATION. IT WILL RUN UNDER APT IN EITHER QUICK VERIFY, PROGRAM OR RUN-TIME MODES.

THIS PROGRAM IS WRITTEN TO COMPLY WITH THE ACT11/XXDP INTERFACE REQUIREMENTS.

WHEN RUNNING IN ACT11 QUICK VERIFY OR XXDP CHAIN MODE (LOC. 42 NOT 0), OR APT QUICK VERIFY AND PROGRAM MODE THE PROGRAM ATTEMPTS TO RUN WITHOUT OPERATOR INTERVENTION OR SWITCH REGISTER SELECTION. THE COMPUTED CRC IS COMPARED AGAINST THE CRC FOR ALL KNOWN VERSIONS OF THE ROM BOOTSTRAP. WHEN A MATCH IS FOUND THE VERSION OF THE MODULE IS TYPED FOR VISUAL VERIFICATION.

```
198
199                             2.3     PROGRAM OPTIONS
200                                     ------- -------
201
202                                     THE PROGRAM AUTOMATICALLY CHECKS FOR THE PRESENCE OF
203                                     A HARDWARE SWITCH REGISTER.  IF NO RESPONSE IS FOUND WHEN
204                                     ADDRESSING THE HARDWARE SWR (177570), THE ADDRESS
205                                     OF THE SOFTWARE SWR (176) IS SUBSTITUTED.
206
207                                     FOR PROCESSORS WITH NO HARDWARE SWITCH REGISTER, THE OPERATOR
208                                     SHOULD SET THE DESIRED SWITCH VALUE IN LOCATION 176
209
210                                     WARNING... IN ORDER TO ALLOW TESTING OF M7942-YB BOARDS ON THE VT71,
211                                     IF LOCATION 176 IS SET TO 6, THE SOFTWARE SWITCH REGISTER WILL BE
212                                     USED REGARDLESS OF HARDWARE SWITCH REGISTER AVAILABILITY.
213
214                             2.3.1   SWITCH SELECTION
215                                     ------ ---------
216                                     THE SWITCH REGISTER (HARDWARE OR SOFTWARE) IS USED TO SELECT
217                                     THE VERSION OF THE ROM BOOTSTRAP  TO BE TES1ED
218                                     ACCORDING TO THE FOLLOWING TABLE, NOTE: THESE SETTINGS ARE
219                                     OCTAL NUMBERS. THEY ARE NOT PARTICULAR SWITCHES SET TO A
220                                     ONE. FOR EXAMPLE, TO SELECT THE M9301-YH VERSION, SET
221                                     SWITCHES #3 AND #1 IN THE SWITCH REGISTER. THIS
222                                     CORRESPONDS TO AN OCTAL 12.
223
224                                     SWR     MODULE VERSION
225                                     ---     --------------
226
227                                     1       M9301-YA
228                                     2       M9301-YB
229                                     3       M9301-YC
230                                     4       M9400-YA (OR YC)
231                                     5       M9301-YF
232                                     6       M7942-YB
233                                     7       M9301-YD
234                                     10      M9400-YH (OR YK)
235                                     11      M9311    REVISION  *  --- SEE NOTE BELOW ---
236                                     12      M9301-YH
237                                     13      M9301-YE
238                                     14      M9301-YJ
239                                     15      M9400-YN
240                                     16      M9311    REVISION  A  *** SEE NOTE BELOW ***
241
242                             NOTE:   FOR M9311 ONLY.....
243                                             IF ROM IN LOCATION E-5 IS  #23-524A9 , THEN
244                                        YOU HAVE REVISION *.  (I.E.  USE SWR  11 SETTING)
245
246                                             IF ROM IN LOCATION E-5 IS  #23-688A9 , THEN
247                                        YOU HAVE REVISION A .( I.E.  USE SWR  16 SETTING)
248
249
250                                     IF THE CRC AND LPC FOR NEW VERSIONS ARE KNOWN BUT
251                                     NOT IN THE ABOVE TABLE, SET THE SWITCH REGISTER TO ZERO
252                                     AND ANSWER THE TELETYPE DIALOG.
253
```

TO DETERMINE THE CRC AND LPC FOR A NEW VERSION, START
THE DIAGNOSTIC AT 200 WITH SWR=0, ANSWER 0 TO THE
REQUESTS FOR THE LPC AND CRC, THE RESULTING MESSAGES
WILL INDICATE THE CORRECT FUTURE RESPONSES FOR CRC AND
LPC PROVIDED THE TEST IS RUN ON A KNOWN-GOOD MODULE.

2.3.2 TELETYPE DIALOG

SEVERAL QUESTIONS ARE ASKED OF THE OPERATOR IN ORDER
TO OBTAIN SUFFICIENT INFORMATION FOR TESTING A ROM MODULE
NOT PREVIOUSLY SUPPORTED IN THE DIAGNOSTIC, THE DIALOG
IS INITIATED IF THE PROGRAM IS STARTED WITH THE SWR = 0,
ALL RESPONSES ARE IN OCTAL AND TERMINATED BY A
CARRIAGE RETURN.

ALL RESPONSES ARE CHECKED FOR VALID OCTAL NUMBERS,
IF AN ILLEGAL CHARACTER IS TYPED, THE PROGRAM WILL TYPE
A "?", CARRIAGE RETURN-LINE FEED AND
AWAIT THE PROPER INPUT.

IF A MISTAKE IS NOTICED BEFORE THE CARRIAGE RETURN IS USED
TO TERMINATE THE INPUT, A RUBOUT CAN BE
USED TO DELETE MISTYPED INPUT.

1. TYPE CRC VALUE:
THIS REQUESTS THE VALUE OF THE CYCLIC REDUNDANCY CHECK
PREVIOUSLY CALCULATED FOR THIS VERSION OF THE ROM MODULE,
IT IS THE VALUE AGAINST WHICH THE UNIT UNDER TEST'S CRC
WILL BE COMPARED.

2. TYPE LPC VALUE:
THIS REQUESTS THE VAULE OF THE LONGITUDINAL PARITY CHECK
PREVIOUSLY CALCULATED FOR THIS VERSION OF THE ROM MODULE,
IT IS THE VALUE AGAINST WHICH THE UNIT UNDER TEST'S LPC
WILL BE COMPARED.

3. TYPE STARTING ADDR. OF 1ST ROM ADDR. SPACE:
THIS QUESTION REFERS TO THE FACT THAT THE ROM SPACE IN
AROM BOOTSTRAP MODULE IS DIVIDED INTO 2 DISTINCT ADDRESS SPACES. TYPE
THE STARTING ADDRESS OF THE 1ST RANGE OF ADDRESSES. THE
STANDARD M9301 & M9400 BEGIN AT 173000.

4. TYPE LENGTH (BYTES) OF 1ST ROM ADDR. SPACE:
THIS REQUESTS THE LENGTH OF THE 1ST GROUP OF ROM ADDRESSES
IN BYTES. THE STANDARD M9301 & M9400 HAVE AN INITIAL ADDRESS SPACE
OF 1000 BYTES. IF THIS SECTION OF ADDRESSES IS NOT
USED BY THIS VERSION, ANSWER 0 TO THIS QUESTION.

5. TYPE STARTING ADDR. OF 2ND ROM ADDR. SPACE:
THIS REFERS TO THE FIRST ADDRESS IN THE SECOND DISTINCT
GROUP OF ROM ADDRESSES. THE RESPONSE FOR A STANDARD
M9301 & M9400 WOULD BE 165000.
6. TYPE LENGTH (BYTES) OF 2ND ROM ADDR. SPACE:
THIS REQUESTS THE LENGTH OF THE 2ND GROUP OF ROM ADDRESSES
IN BYTES. THE STANDARD M9301 & M9400 HAVE A SECOND ADDRESS SPACE

OF 1000 BYTES. IF THIS SECTION OF ADDRESSES IS NOT USED
BY THIS VERSION, ANSWER 0 TO THIS QUESTION.

2.4 EXECUTION TIMES

THE DIAGNOSTIC COMPLETES 1 PASS IN LESS THAN 1 SEC.
ONCE THE INPUT DIALOG HAS BEEN COMPLETED.
THE PROGRAM WILL HALT UPON COMPLETION; HOWEVER, IF RUNNING
UNDER APT THE PROGRAM WILL CYCLE CONTINUOUSLY.

3.0 ERROR INFORMATION

WHEN THE DIAGNOSTIC DETECTS A DISCREPANCY BETWEEN THE EXPECTED AND COMPUTED CRC OR LPC BOTH ARE PRINTED ON THE TELETYPE.

ANY DISCREPANCY IN THE LPC CAN ASSIST IN ISOLATING THE PROBLEM TO THE ROM IC.

UNDER APT THE ERROR IS INDICATED BY DEPOSITING ERROR INFORMATION IN THE APT MAILBOX BEFORE HALTING.

4.0 PROGRESS REPORTS

AT THE END OF EACH PASS THE PROGRAM INCREMENTS TO THE LOCATION $PASS WHICH IS IN THE APT MAILBOX. THIS LOCATION WILL ALWAYS CONTAIN THE NUMBER OF PASSES COMPLETED. $PASS IS RESET WITH EVERY RESTART.

ADDITIONALLY, THE MESSAGE "END OF TEST" IS PRINTED ON THE CONSOLE TELETYPE AFTER EACH PASS. NORMALLY ONLY ONE PASS NEEDS TO RUN TO VERIFY THE MODULE.

5.0 TROUBLE SHOOTING

THE ALGORITHM FOR COMPUTING THE CRC IS THE SAME AS THAT USED ON 9-TRACK MAGNETIC TAPE WITH ODD PARITY. THE CRC IS CALCULATED ON A BYTE-BY-BYTE BASIS. WHILE THE ALGORITHM IS SUCCESSFUL IN DETECTING MULTIPLE ERRORS, ITS USE AS A DEBUGGING AID IS LIMITED.

THE LPC IS CALCULATED BY ASSEMBLING THE XOR OF EVERY WORD IN THE ROM. WHILE ONLY USEFUL IN CATCHING AN ODD NUMBER OF ERRORS IN EACH BIT POSITION, IT IS VERY USEFUL IN ISOLATING THE PROBLEM TO A CHIP. BY LOCATING WHICH BIT POSITIONS ARE IN DISCREPANCY, THE CORRESPONDING ROM CHIPS CAN BE ISOLATED.

IF NO OTHER CLUES CAN BE OBTAINED, START THE PROGRAM AT 210 AND COMPARE THE PRINTOUT OF THE CODE WITH THE LISTING FOR THE VERSION BEING TESTED.

* * CAUTION * *

BECAUSE THE CONTENTS READ FROM LOCATION 773024 OF THE M9301 OPTION IS CONFIGURATION DEPENDANT(SWITCH REGISTER DEPENDANT), THIS LOCATION IS NOT INCLUDED IN THE DATA CHECK.
THIS LOCATION CAN BE VERIFIED BY EXAMINING IT OR BY USING THE ALTERNATE STARTING ADDRESS(SEE SECTION 2.1.3) TO PRINT OUT THIS LOCATION.

SEQ 0009

```
                                6.0     LISTING
                                        -------
                                %
```


SEQ 0010

```
000000                  $SWR=0
                        .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          ;;HARDWARE 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
000040                  PR1=    40              ;;PRIORITY LEVEL 1
000100                  PR2=    100             ;;PRIORITY LEVEL 2
000140                  PR3=    140             ;;PRIORITY LEVEL 3
000200                  PR4=    200             ;;PRIORITY LEVEL 4
000240                  PR5=    240             ;;PRIORITY LEVEL 5
000300                  PR6=    300             ;;PRIORITY LEVEL 6
000340                  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
                                                .EQUTV  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
                        000000                          .=0
                                                .SBTTL  TRAP CATCHER

                        000000                          .=0
                                                ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ",+2,HALT"
                                                ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
                                                ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
                        000174                          .=174
000174  000000         DISPREG: .WORD  0               ;;SOFTWARE DISPLAY REGISTER
000176  000000         SWREG:   .WORD  0               ;;SOFTWARE SWITCH REGISTER
                        000200                  .=200
000200  005067  000624                  CLR     TYPOUT
000204  000167  000666                  JMP     START
000210  012767  000001  000612          MOV     #1,TYPOUT
000216  000167  000654                  JMP     START

                        177776                  PS=177776
                        000034                  TRAPVEC=34

                        001000                  .=1000
001000  177570         SWR:    177570
001002  177570         DISPLAY:        177570
001004  173000         ROMSA1: 173000
001006  001000         DATLN1: 512.
001010  165000         ROMSA2: 165000
001012  001000         DATLN2: 512.
001014  000000         XORS:   0
001016  000000         EXCRC:  0
001020  000000         EXLPC:  0
001022  000000         ACTCRC: 0
001024  000000         ACTLPC: 0
001026  000000         PARCNT: 0
001030  000000         TYPOUT: 0

                                                .SBTTL  ACT11 HOOKS

                                                ;;**************************************************************
                                                ;HOOKS REQUIRED BY ACT11
                        001032                          $SVPC=.                 ;SAVE PC
                        000046                          .=46
000046  002060                          $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
                        000052                          .=52
000052  000000                          .WORD   0               ;;2)SET LOC.52 TO ZERO
                        001032                          .=$SVPC                 ;; RESTORE PC
                                                .SBTTL  APT PARAMETER BLOCK

                                                ;;**************************************************************
                                                ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
                                                ;;**************************************************************
                        001032                          .$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.
```

```
                        001032                  $APTHDR ;;POINT TO APT HEADER BLOCK
                        001032                  .=.$X   ;;RESET LOCATION COUNTER
                                        ;;**************************************************************
                                        ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
                                        ;INTERFACE SPEC.
001032                                  $APTHD:
001032  000000                          $HIBTS: .WORD   0       ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
001034  001046                          $MBADR: .WORD   $MAIL   ;;ADDRESS OF APT MAILBOX (BITS 0-15)
001036  000002                          $TSTM:  .WORD   2       ;;RUN TIM OF LONGEST TEST
001040  000002                          $PASTM: .WORD   2       ;;RUN TIME IN SECS, OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
001042  000000                          $UNITM: .WORD   0       ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
001044  000014                                  .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
                                        .SBTTL  APT MAILBOX-ETABLE
                                        ;;**************************************************************
                                        .EVEN
001046                                  $MAIL:                  ;;APT MAILBOX
001046  000000                          $MSGTY: .WORD   AMSGTY  ;;MESSAGE TYPE CODE
001050  000000                          $FATAL: .WORD   AFATAL  ;;FATAL ERROR NUMBER
001052  000000                          $TESTN: .WORD   ATESTN  ;;TEST NUMBER
001054  000000                          $PASS:  .WORD   APASS   ;;PASS COUNT
001056  000000                          $DEVCT: .WORD   ADEVCT  ;;DEVICE COUNT
001060  000000                          $UNIT:  .WORD   AUNIT   ;;I/O UNIT NUMBER
001062  000000                          $MSGAD: .WORD   AMSGAD  ;;MESSAGE ADDRESS
001064  000000                          $MSGLG: .WORD   AMSGLG  ;;MESSAGE LENGTH
001066                                  $ETABLE:                ;;APT ENVIRONMENT TABLE
001066     000                          $ENV:   .BYTE   AENV    ;;ENVIRONMENT BYTE
001067     000                          $ENVM:  .BYTE   AENVM   ;;ENVIRONMENT MODE BITS
001070  000000                          $SWREG: .WORD   ASWREG  ;;APT SWITCH REGISTER
001072  000000                          $USWR:  .WORD   AUSWR   ;;USER SWITCHES
001074  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
001076                                  $ETEND:
                                        .MEXIT

001076  005067  177746          START:  CLR     $FATAL                  ;CLEAR ERROR NO.
001102  005067  177740                  CLR     $MSGTYP                 ;CLEAR MESSAGE TYPE (APT)
001106  012767  000001  177736          MOV     #1,$TESTN               ;SET TEST NO.
                                        .SBTTL  INITIALIZE THE COMMON TAGS
001114  012706  000500                  MOV     #500,SP         ;;SETUP THE STACK POINTER
                                        ;;INITIALIZE A FEW VECTORS
001120  012737  004570  000034          MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
001126  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
001134  012737  004412  000024          MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
001142  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;LEVEL 7
                                        ;;SIZE FOR A HARDWARE SWITCH REGISTER, IF NOT FOUND OR IT IS
                                        ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
001150  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
001154  012737  001210  000004          MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
001162  012767  177570  177610          MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
```



```
001170  012767  177570  177604          MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
001176  022777  177777  177574          CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
001204  001012                          BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                                ;;AND  THE HARDWARE SWR IS NOT = -1
001206  000403                          BR      65$             ;;BRANCH IF NO TIMEOUT
001210  012716  001216          64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
001214  000002                          RTI
001216  012767  000176  177554  65$:    MOV     #SWPEG,SWR      ;;POINT TO SOFTWARE SWR
001224  012767  000174  177550          MOV     #DISPREG,DISPLAY
001232  012637  000004          66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR

001236  005067  177612                  CLR     $PASS           ;;CLEAR PASS COUNT
001242  132767  000200  177617          BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
001250  001403                          BEQ     67$             ;;YES,USE NON-APT SWITCH
001252  012767  001070  177520          MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
001260                          67$:
001260  026727  176556  002060          CMP     42,#$ENDAD              ;ACT AUTO MODE?
001266  001402                          BEQ     RESTRT                  ;YIF SO: BR
001270  104401                          TYPE
001272  005276                          TITL

001274  123727  000176  000006  RESTRT: CMPB    @#176,#6                ;IS SOFTWARE SWITCH REGISTER =6?
001302  001003                          BNE     1$                      ;IF NOT, TEST NORMALY
001304  012767  000176  177466          MOV     #SWREG,SWR              ;IF SO USE THE SOFTWARE SWITCH REG
001312  017700  177462          1$:     MOV     @SWR,R0                 ;GET SWR
001316  001424                          BEQ     GETIN                   ;IF ZERO: GET INPUT
001320  006300                  ST2:    ASL     R0
001322  016067  004714  177470          MOV     TXLPC(R0),EXLPC         ;FETCH EXPECT, LPC
001330  016067  004644  177460          MOV     TXCRC(R0),EXCRC         ;FETCH EXPECTED CRC
001336  016067  004766  177442          MOV     TDLN1(R0),DATLN1        ;FETCH 1ST LENGTH
001344  016067  005036  177432          MOV     TRMSA1(R0),ROMSA1       ;FETCH 1ST STARTING ADDR,
001352  016067  005106  177432          MOV     TDLN2(R0),DATLN2                ;FETCH 2ND LENGTH
001360  016067  005156  177422          MOV     TRMSA2(R0),ROMSA2       ;FETCH 2ND STARTING ADDR
001366  000450                          BR      CHECK                   ;GO COMPUTE
001370  005767  176446          GETIN:  TST     42                      ;UNDER ACT AUT ACCEPT?
001374  001045                          BNE     CHECK                   ;IF SO; BR USE DEFAULT PARAMETERS
001376  122767  000001  177462          CMPB    #1,$ENV                 ;UNDER APT?
001404  001441                          BEQ     CHECK                   ;IF SO: BR
001406  005767  177416                  TST     TYPOUT                  ;ROM TYPE OPTION
001412  001012                          BNE     GET2                    ;IF SO:  BR
001414  104401                          TYPE
001416  005311                          GETCRC
001420  104407                          RDOCT
001422  012667  177370                  MOV     (SP)+,EXCRC             ;STORE EXPECT, CRC
001426  104401                          TYPE                            ;TYPE LPC INPUT REQUEST
001430  005335                          GETLPC
001432  104407                          RDOCT
001434  012667  177360                  MOV     (SP)+,EXLPC             ;STORE EXPECTED LPC
001440  104401                  GET2:   TYPE
001442  005515                          SA1                             ;REQUEST 1ST ADDRESS SPACE
001444  104407                          RDOCT                           ;INPUT SA
001446  012667  177332                  MOV     (SP)+,ROMSA1
001452  104401                          TYPE                            ;REQUEST LENGTH OF 1ST ADDR, SPACE
001454  005655                          SIZE1
001456  104407                          RDOCT                           ;INPUT LENGTH
001460  012667  177322                  MOV     (SP)+,DATLN1
```

```
001464  104401                          TYPE                    ;REQUEST START ADDR, FOR 2ND SPACE
001466  005575                          SA2
001470  104407                          RDOCT                   ;INPUT SA
001472  012667  177312                  MOV     (SP)+,ROMSA2
001476  104401                          TYPE                    ;REQUEST LENGTH OF 2ND SPACE
001500  005735                          SIZE2
001502  104407                          RDOCT                   ;INPUT LENGTH
001504  012667  177302                  MOV     (SP)+,DATLN2
001510  005767  177314          CHECK:  TST     TYPOUT          ;IS TYPOUT REQUESTED?
001514  001402                          BEQ     1$              ;BRANCH IF NOT
001516  000167  000660                  JMP     TYPROM          ;GO TYPE OUT POM
001522  005067  177274          1$:     CLR     ACTCRC          ;CLEAR STORAGE FOR ACTUAL CRC
001526  005067  177272                  CLR     ACTLPC          ;CLEAR STORAGE FOR ACTULAL LPC
001532  016700  177250                  MOV     DATLN1,R0       ;SET LENGTH OF 1ST ROM SPACE
001536  001413                          BEQ     CH0             ;IF NO VERSION SELECTED:  BR
001540  016701  177240                  MOV     ROMSA1,R1       ;POINT TO START OF 1ST ROM SPACE
001544  004767  000324                  JSR     PC,CRC          ;COMPUTE FIRST HALF OF CRC
001550  016701  177230                  MOV     ROMSA1,R1       ;POINT TO START OF 1ST ROM ADDRS.
001554  016700  177226                  MOV     DATLN1,R0       ;SET LENGTH OF 1ST ROM ADDRS.
001560  006200                          ASR     R0              ;CONVERT TO WORDS
001562  004767  000556                  JSR     PC,LPC          ;COMPUTE FIRST HALF OF CRC
001566  016701  177216          CH0:    MOV     ROMSA2,R1       ;POINT TO 2ND ROM ADDRS.
001572  016700  177214                  MOV     DATLN2,R0       ;SET LENGTH OF 2ND ROM ADDRS.
001576  001422                          BEQ     CH1             ;BR IF THIS SPACE NOT USED
001600  004767  000270                  JSR     PC,CRC          ;COMPUTE REMAINDER OF CRC
001604  016701  177200                  MOV     ROMSA2,R1       ;POINT TO START OF 2ND ROM ADDRS.
001610  016700  177176                  MOV     DATLN2,R0       ;SET LENGTH OF 2ND ROM ADDRS.
001614  006200                          ASR     R0              ;CONVERT TO WORDS
001616  004767  000522                  JSR     PC,LPC          ;COMPUTE REMAINDER OF LPC
001622  122767  000001  177236          CMPB    #1,$ENV         ;UNDER APT?
001630  001403                          BEQ     1$              ;IF SO:  BR
001632  005767  176204                  TST     42              ;UNDER ACT AUTO ACCEPT?
001636  001402                          BEQ     CH1             ;IF NOT:  BR
001640  000167  000656          1$:     JMP     AUTACT
001644  026767  177146  177150  CH1:    CMP     EXCRC,ACTCRC    ;COMPUTED = EXPECTED ?
001652  001431                          BEQ     CK1             ;IF SO:  BR

001654  104401                          TYPE                    ;TYPE CRC ERROR MESG,
001656  005361                          EXCRMG
001660  016746  177132                  MOV     EXCRC,-(SP)     ;PUT EXPECT CRC ON STACK
001664  104402                          TYPOC                   ;TYPE EXPECTED CRC
001666  104401                          TYPE                    ;TYPE ACTUAL CRC MESSG
001670  005430                          ACCRMG
001672  016746  177124                  MOV     ACTCRC,-(SP)    ;PUT ACTUAL CRC ON STACK
001676  104402                          TYPOC                   ;TYPE ACTUAL CRC
001700  026727  176136  002060          CMP     42,#$ENDAD      ;UNDER ACT AUTO MODE?
001706  001404                          BEQ     1$              ;IF SO:  BR
001710  122767  000001  177150          CMPB    #1,$ENV         ;UNDER APT?
001716  001007                          BNE     CK1             ;IF NOT:  BR
001720  012767  000002  177122  1$:     MOV     #2,$FATAL       ;MOVE TO MAILBOX ERROR NO.  ****  2  ****
001726  012767  000001  177112          MOV     #1,$MSGTYP      ;SET MAILBOX FOR FATAL ERROR
001734  000000                          HALT                    ;CRC ERROR

001736  026767  177062  177054  CK1:    CMP     ACTLPC,EXLPC    ;COMPARE EXPT. LPC=ACTUAL LPC
001744  001431                          BEQ     CK2             ;IF SO:  BR
001746  104401                          TYPE                    ;TYPE LPC ERROR MESSG,
```



```
001750  005404                          EXLPMG
001752  016746  177042                  MOV     EXLPC,-(SP)     ;PUT EXPECTED LPC ON STACK
001756  104402                          TYPOC                   ;TYPE EXPECTED LPC
001760  104401                          TYPE                    ;TYPE ACTUAL LPC MESSG,
001762  005453                          ACLPMG
001764  016746  177034                  MOV     ACTLPC,-(SP)    ;PUT ACTUAL LPC ON STACK
001770  104402                          TYPOC                   ;TYPE ACTUAL LPC
001772  026727  176044  002060          CMP     42,#$ENDAD      ;UNDER ACT AUTO MODE?
002000  001404                          BEQ     1$              ;IF SO:  BR
002002  122767  000001  177056          CMPB    #1,$ENV         ;UNDER APT?
002010  001007                          BNE     CK2             ;IF NOT:  BR
002012  012767  000003  177030  1$:     MOV     #3,$FATAL       ;MOVE TO MAILBOX ERROR NO.  ****  3  ****
002020  012767  000001  177020          MOV     #1,$MSGTYP      ;SET MAILBOX FOR FATAL ERROR
002026  000000                          HALT                    ;LPC ERROR

002030  026727  176006  002060  CK2:    CMP     42,#$ENDAD      ;ACT AUTO ACCEPT?
002036  001402                          BEQ     1$              ;IF SO:  BR
002040  104401                          TYPE                    ;TYPE END OF TEST
002042  005476                          EOTST
002044  005267  177004          1$:     INC     $PASS           ;BUMP PASS COUNT
002050  013700  000042                  MOV     @#42,R0         ;CHECK APT
002054  001405                          BEQ     GOAGIN          ;KEEP GOING
002056  000005                          RESET
002060  004710                  $ENDAD: JSR     PC,(R0)         ;ACT HOOKS
002062  000240                          NOP
002064  000240                          NOP
002066  000240                          NOP
002070  000167  177200          GOAGIN: JMP     RESTRT          ;DO AGAIN

002074  016767  176722  176712  CRC:    MOV     ACTCRC,XORS
002102  111104                  CL0:    MOVB    (R1),R4         ;GET CHAR,
002104  022701  173024                  CMP     #173024,P1      ;LOCATION EFFECTED BY SWITCHES
002110  001004                          BNE     CL3             ;IF NOT:  BR
002112  005300                          DEC     R0              ;FIX COUNTERS
002114  005300                          DEC     R0
002116  005721                          TST     (P1)+           ;FIX POINTER
002120  000770                          BR      CL0             ;CONTINUE
002122  004767  000114          CL3:    JSR     PC,PARITY       ;GO GET PARITY
002126  004767  000166                  JSR     PC,XOR          ;XOR CHAR
002132  000241                          CLC
002134  006004                          ROR     R4              ;ROTATE 1 POS, RIGHT
002136  103014                          BCC     CL2             ;IF NO CARRY: BR
002140  052704  000400                  BIS     #400,R4         ;SET BIT NINE
002144  000241                          CLC
002146  010405                  CL1:    MOV     R4,P5           ;SAVE CHAR
002150  042705  177703                  BIC     #177703,P5
002154  005105                          COM     P5
002156  042705  177703                  BIC     #177703,R5
002162  042704  000074                  BIC     #74,R4
002166  050504                          BIS     P5,R4
002170  010467  176620          CL2:    MOV     R4,XORS
002174  005300                          DEC     R0
002176  001402                          BEQ     CLLAST          ;IF LAST CAR,:  BR
002200  000167  177676                  JMP     CL0             ;GET NEXT CHAR,
002204  016704  176604          CLLAST: MOV     XOR8,R4
002210  005167  176600                  COM     XORS
```

```
777  002214  042767  177050  176572          BIC     #177050,XORS
778  002222  042704  177727                  BIC     #177727,R4              ;COMPLEMENT ALL BUT BITS 3 & 5
779  002226  050467  176562                  BIS     R4,XORS
780  002232  016767  176556  176562          MOV     XORS,ACTCRC
781  002240  000207                          RTS     PC
782  002242  005067  176560          PARITY: CLR     PARCNT                  ;CLEAR BIT COUNTER
783  002246  012703  000010                  MOV     #10,R3                  ;SET NO. OF BITS
784  002252  032704  000001          CLP0:   BIT     #1,R4                   ;SEE IF ONE BIT
785  002256  001402                          BEQ     CLP1                    ;IF NOT:  BR
786  002260  005267  176542                  INC     PARCNT                  ;BUMP COUNTER
787  002264  000241                  CLP1:   CLC
788  002266  006004                          ROR     R4                      ;ROTATE TO NEXT BIT
789  002270  005303                          DEC     R3
790  002272  001367                          BNE     CLP0                    ;CONTINUE FOR ALL BITS
791  002274  112104                          MOVB    (R1)+,R4
792  002276  042704  177400                  BIC     #177400,R4
793  002302  032767  000001  176516          BIT     #1,PARCNT               ;SEE IF ODD # OF ONE BITS
794  002310  001002                          BNE     CLP2                    ;IF SO:  BR
795  002312  052704  000400                  BIS     #400,R4                 ;SET PARITY BIT
796  002316  000207                  CLP2:   RTS     PC                      ;EXIT
797
798  002320  010446                  XOR:    MOV     R4,-(SP)                ;XOR SUBROUTINE: R4 WITH XORS
799  002322  046716  176466                  BIC     XORS,(SP)
800  002326  040467  176462                  BIC     R4,XORS
801  002332  052667  176456                  BIS     (SP)+,XORS
802  002336  016704  176452                  MOV     XORS,R4
803  002342  000207                          RTS     PC
804
805  002344  016767  176454  176442  LPC:    MOV     ACTLPC,XORS
806  002352  012104                  LPC1:   MOV     (R1)+,R4
807  002354  022701  173026                  CMP     #173026,R1              ;LOCATION EFFECTED BY SWITCHES
808  002360  001402                          BEQ     LPC2                    ;IF SO:  SKIP LOC. BY BRANCHING
809  002362  004767  177732                  JSR     PC,XOR
810  002366  005300                  LPC2:   DEC     R0
811  002370  001370                          BNE     LPC1
812  002372  016767  176416  176424          MOV     XORS,ACTLPC
813  002400  000207                          RTS     PC
814
815  002402  104401          TYPROM: TYPE                                    ;TYPE HEADER
816  002404  006023                  TYPHDR
817  002406  016700  176372                  MOV     ROMSA1,R0               ;POINT OT 1ST ROM SPACE
818  002412  016701  176370                  MOV     DATLN1,R1               ;PUT LENGTH IN R1
819  002416  006201                          ASR     R1                      ;CONERT TO WORDS
820  002420  001402                          BEQ     TYPR1                   ;BRANCH IF 1ST ROM SPACE NOT USED
821  002422  004767  000026                  JSR     PC,TYP                  ;GO TYPE 1ST ADDR, SPACE
822  002426  016700  176356          TYPR1:  MOV     ROMSA2,R0               ;POINT TO 2ND ADDR, SPACE
823  002432  016701  176354                  MOV     DATLN2,R1               ;PUT LENGTH IN R1
824  002436  006201                          ASR     R1                      ;CONVERT TO WORDS
825  002440  001402                          BEQ     ENDOT                   ;BR IF 2ND ADDR, SPACE NOT USED
826  002442  004767  000006                  JSR     PC,TYP                  ;GO TYPE 2ND ADDR, SPACE
827  002446  104401          ENDOT:  TYPE
828  002450  005476                  EOTST
829  002452  000000                  HALT
830
831  002454  104401          TYP:    TYPE
832  002456  006015                  CARLF
```



```
833  002460  000403                          BR      TYP3
834  002462  032700  000003          TYP0:   BIT     #3,R0                   ;ADDRESS MULTIPLE OF 4?
835  002466  001006                          BNE     TYP2                    ;IF NOT:  BR
836  002470  104401          TYP3:   TYPE
837  002472  006015                  CARLF
838  002474  010046                  MOV     R0,-(SP)                        ;PUT ADDRESS ON STACK
839  002476  104402                  TYPOC                                   ;TYPE ADDR,
840  002500  104401                  TYPE
841  002502  006060                  COLON
842  002504  012046          TYP2:   MOV     (R0)+,-(SP)             ;PUT DATA ON STACK
843  002506  104402                  TYPOC                           ;TYP DATA
844  002510  104401                  TYPE                            ;TYPE 2 SPACES
845  002512  006020                  SP2
846  002514  005301                  DEC     R1                      ;FINISHED?
847  002516  001361                  BNE     TYP0                    ;IF NOT:  BR
848  002520  000207                  RTS     PC                      ;RETURN
849  002522  005000          AUTACT: CLR     R0
850  002524  062700  000002  AUT1:   ADD     #2,R0                   ;BUMP TABLE INDEX
851  002530  026027  005226  177777          CMP     TMSG(R0),#-1            ;CHECKED ALL KNOWN VERSIONS?
852  002536  001425                          BEQ     AUTERR                  ;IF SO:  BR
853  002540  026067  004644  176254          CMP     TXCRC(R0),ACTCRC        ;DOES THIS CRC AGREE?
854  002546  001366                          BNE     AUT1                    ;IF NOT:  KEEP LOOKING
855  002550  023727  000042  002060          CMP     @#42,#SENDAD            ;UNDER ACT AUTO ACCEPT?
856  002556  001405                          BEQ     AUT3                    ;IF SO;  BR
857  002560  016067  005226  000002          MOV     TMSG(R0),AUT2           ;SET UP VERSION MESSAGE
858  002566  104401                  TYPE
859  002570  000000          AUT2:   0
860  002572  016067  004644  176216  AUT3:   MOV     TXCRC(R0),EXCRC         ;SET EXPECTED CRC
861  002600  016067  004714  176212          MOV     TXLPC(R0),EXLPC         ;SET EXPECTED LPC
862  002606  000167  177124                  JMP     CK1                     ;CHECK LPC
863
864  002612  104401          AUTERR: TYPE
865  002614  006064                  AUTERM
866  002616  012767  000001  176224          MOV     #1,$FATAL               ;MOVE TO MAILBOX ERROR NO,  ****  1  ****
867  002624  012767  000001  176214          MOV     #1,$MSGTYP              ;SET MAILBOX FOR FATAL ERROR
868  002632  000000                          HALT                            ;AUTO ACCEPT FAILED
869
870                                  .SBTTL  TTY INPUT ROUTINE
871
872                                  ;;*************************************************************
873  002634  177560                  $TKS:   .WORD   177560          ;;TTY KBD STATUS
874  002636  177562                  $TKB:   .WORD   177562          ;;TTY KBD BUFFER
875                                  .ENABL  LSB
876
877                                  .DSABL  LSB
878
879
880                                  ;;*************************************************************
881                                  ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
882                                  ;*CALL:
883                                  ;*      RDCHR                   ;;INPUT A SINGLE CHARACTER FROM THE TTY
884                                  ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
885                                  ;*                              ;;WITH PARITY BIT STRIPPED OFF
886                                  ;
887
888  002640  011646                  $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
```

```
 889  002642  016666  000004  000002          MOV     4(SP),2(SP)     ;;SAVE THE PS
 890  002650  105777  177760          1$:     TSTB    @$TKS           ;;WAIT FOR
 891  002654  100375                          BPL     1$              ;;A CHARACTER
 892  002656  117766  177754  000004          MOVB    @$TKB,4(SP)     ;;READ THE TTY
 893  002664  042766  177600  000004          BIC     #^C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
 894  002672  026627  000004  000023          CMP     4(SP),#23       ;;IS IT A CONTROL-S?
 895  002700  001013                          BNE     3$              ;;BRANCH IF NO
 896  002702  105777  177726          2$:     TSTB    @$TKS           ;;WAIT FOR A CHARACTER
 897  002706  100375                          BPL     2$              ;;LOOP UNTIL ITS THERE
 898  002710  117746  177722                  MOVB    @$TKB,-(SP)     ;;GET CHARACTER
 899  002714  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
 900  002720  022627  000021                  CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
 901  002724  001366                          BNE     2$              ;;IF NOT DISCARD IT
 902  002726  000750                          BR      1$              ;;YES, RESUME
 903  002730  026627  000004  000140  3$:     CMP     4(SP),#140      ;;IS IT UPPER CASE?
 904  002736  002407                          BLT     4$              ;;BRANCH IF YES
 905  002740  026627  000004  000175          CMP     4(SP),#175      ;;IS IT A SPECIAL CHAR?
 906  002746  003003                          BGT     4$              ;;BRANCH IF YES
 907  002750  042766  000040  000004          BIC     #40,4(SP)       ;;MAKE IT UPPER CASE
 908  002756  000002                  4$:     RTI                     ;;GO BACK TO USER
 909                                          ;;*************************************************************
 910                                          ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
 911                                          ;*CALL:
 912                                          ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
 913                                          ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
 914                                          ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
 915
 916  002760  010346                  $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
 917  002762  005046                          CLR     -(SP)           ;;CLEAR THE RUBOUT KEY
 918  002764  012703  003214          1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
 919  002770  022703  003224          2$:     CMP     #$TTYIN+8.,R3   ;;BUFFER FULL?
 920  002774  101456                          BLOS    4$              ;;BR IF YES
 921  002776  104405                          RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
 922  003000  112613                          MOVB    (SP)+,(R3)      ;;GET CHARACTER
 923  003002  122713  000177          10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
 924  003006  001022                          BNE     5$              ;;BR IF NO
 925  003010  005716                          TST     (SP)            ;;IS THIS THE FIRST RUBOUT?
 926  003012  001007                          BNE     6$              ;;BR IF NO
 927  003014  112767  000134  000170          MOVB    #'\,9$          ;;TYPE A BACK SLASH
 928  003022  104401  003212                  TYPE    ,9$
 929  003026  012716  177777                  MOV     #-1,(SP)        ;;SET THE RUBOUT KEY
 930  003032  005303                  6$:     DEC     R3              ;;BACKUP BY ONE
 931  003034  020327  003214                  CMP     R3,#$TTYIN      ;;STACK EMPTY?
 932  003040  103434                          BLO     4$              ;;BR IF YES
 933  003042  111367  000144                  MOVB    (R3),9$         ;;SETUP TO TYPEOUT THE DELETED CHAR.
 934  003046  104401  003212                  TYPE    ,9$             ;;GO TYPE
 935  003052  000746                          BR      2$              ;;GO READ ANOTHER CHAR.
 936  003054  005716                  5$:     TST     (SP)            ;;RUBOUT KEY SET?
 937  003056  001406                          BEQ     7$              ;;BR IF NO
 938  003060  112767  000134  000124          MOVB    #'\,9$          ;;TYPE A BACK SLASH
 939  003066  104401  003212                  TYPE    ,9$
 940  003072  005016                          CLR     (SP)            ;;CLEAR THE RUBOUT KEY
 941  003074  122713  000025          7$:     CMPB    #25,(R3)        ;;IS CHARACTER A CTRL U?
 942  003100  001003                          BNE     8$              ;;BR IF NO
 943  003102  104401  003230                  TYPE    ,$CNTLU         ;;TYPE A CONTROL "U"
 944  003106  000726                          BR      1$              ;;GO START OVER
```



```
 945  003110  122713  000022          8$:     CMPB    #22,(R3)        ;;IS CHARACTER A "^R"?
 946  003114  001011                          BNE     3$              ;;BRANCH IF NO
 947  003116  105013                          CLRB    (R3)            ;;CLEAR THE CHARACTER
 948  003120  104401  003225                  TYPE    ,$CRLF          ;;TYPE A "CR" & "LF"
 949  003124  104401  003214                  TYPE    ,$TTYIN         ;;TYPE THE INPUT STRING
 950  003130  000717                          BR      2$              ;;GO PICKUP ANOTHER CHACTER
 951  003132  104401  003224          4$:     TYPE    ,$QUES          ;;TYPE A "?"
 952  003136  000712                          BR      1$              ;;CLEAR THE BUFFER AND LOOP
 953  003140  111367  000046          3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
 954  003144  104401  003212                  TYPE    ,9$
 955  003150  122723  000015                  CMPB    #15,(R3)+       ;;CHECK FOR RETURN
 956  003154  001305                          BNE     2$              ;;LOOP IF NOT RETURN
 957  003156  105063  177777                  CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
 958  003162  104401  003226                  TYPE    ,$LF            ;;TYPE A LINE FEED
 959  003166  005726                          TST     (SP)+           ;;CLEAN RUBOUT KEY FROM THE STACK
 960  003170  012603                          MOV     (SP)+,R3        ;;RESTORE R3
 961  003172  011646                          MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
 962  003174  016666  000004  000002          MOV     4(SF),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
 963  003202  012766  003214  000004          MOV     #$TTYIN,4(SP)
 964  003210  000002                          RTI                     ;;RETURN
 965  003212     000                  9$:     .BYTE   0               ;;STORAGE FOR ASCII CHR, TO TYPE
 966  003213     000                          .BYTE   0               ;;TERMINATOR
 967  003214  000010          $TTYIN: .BLKB   8.              ;;RESERVE 8 BYTES FOR TTY INPUT
 968  003224     077          $QUES:  .ASCII  "?"     ;;QUESTION MARK
 969  003225     015          $CRLF:  .ASCII  <15>    ;CARRIAGE RETURN
 970  003226  000012          $LF:    .ASCIZ  <12>    ;;LINE FEED
 971  003230  052536  005015    000   $CNTLU: .ASCIZ  /^U/<15><12>    ;;CONTROL "U"
 972  003235     136  006507  000012  $CNTLG: .ASCIZ  /^G/<15><12>    ;;CONTROL "G"
 973  003242  005015  053523  020122  $SWR:   .ASCIZ  <15><12>/SWR = /
 974  003250  020075     000
 975  003253     040  047040  053505  $MNEW:  .ASCIZ  /  NEW = /
 976  003260  036440  000040
 977                                  .SBTTL  READ AN OCTAL NUMBER FROM THE TTY
 978
 979                                  ;;*************************************************************
 980                                  ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
 981                                  ;*CHANGE IT TO BINARY.
 982                                  ;*THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL
 983                                  ;*OCTAL DIGITS. IF AN ILLEGAL CHARACTER IS READ A "?" WILL BE TYPED
 984                                  ;*FOLLOWED BY A CARRIAGE RETURN-LINE FEED, THE COMPLETE NUMBER MUST
 985                                  ;*THEN BE RETYPED. THE INPUT IS TERMINATED BY TYPING A CARRIAGE RETURN.
 986                                  ;*CALL:
 987                                  ;*      RDOCT                   ;;READ AN OCTAL NUMBER
 988                                  ;*      RETURN HERE             ;;LOW ORDER BITS ARE ON TOP OF THE STACK
 989                                  ;*                              ;;HIGH ORDER BITS ARE IN $HIOCT
 990
 991  003264  011646          $RDOCT: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR THE
 992  003266  016666  000004  000002          MOV     4(SP),2(SP)     ;;INPUT NUMBER
 993  003274  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
 994  003276  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
 995  003300  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
 996  003302  104406                  1$:     RDLIN                   ;;READ AN ASCIZ LINE
 997  003304  012600                          MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
 998  003306  010067  000100                  MOV     R0,5$           ;;AND SAVE IT
 999  003312  005001                          CLR     R1              ;;CLEAR DATA WORD
1000  003314  005002                          CLR     R2
```

```
 1041  003316  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
 1042  003320  001420                          BEQ     3$              ;;IF ZERO GET OUT
 1043  003322  122716  000060                  CMPB    #'0,(SP)        ;;MAKE SURE THIS CHARACTER
 1044  003326  003026                          BGT     4$              ;;IS AN OCTAL DIGIT
 1045  003330  122716  000067                  CMPB    #'7,(SP)
 1046  003334  002423                          BLT     4$
 1047  003336  006301                          ASL     R1              ;;*2
 1048  003340  006102                          ROL     R2
 1049  003342  006301                          ASL     R1              ;;*4
 1050  003344  006102                          ROL     R2
 1051  003346  006301                          ASL     R1              ;;*8
 1052  003350  006102                          ROL     R2
 1053  003352  042716  177770                  BIC     #^C7,(SP)       ;;STRIP THE ASCII JUNK
 1054  003356  062601                          ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
 1055  003360  000756                          BR      2$              ;;LOOP
 1056  003362  005726                  3$:     TST     (SP)+           ;;CLEAN TERMINATOR FROM STACK
 1057  003364  010166  000012                  MOV     R1,12(SP)       ;;SAVE THE RESULT
 1058  003370  010267  000026                  MOV     R2,$HIOCT
 1059  003374  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
 1060  003376  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
 1061  003400  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
 1062  003402  000002                          RTI                     ;;RETURN
 1063  003404  005726                  4$:     TST     (SP)+           ;;CLEAN PARTIAL FROM STACK
 1064  003406  105010                          CLRB    (R0)            ;;SET A TERMINATOR
 1065  003410  104401                          TYPE                    ;;TYPE UP THRU THE BAD CHAR,
 1066  003412  000000                  5$:     .WORD   0
 1067  003414  104401  003224                  TYPE    ,$QUES          ;;"?" "CR" & "LF"
 1068  003420  000730                          BR      1$              ;;TRY AGAIN
 1069  003422  000000                  $HIOCT: .WORD   0               ;;HIGH ORDER BITS GO HERE
 1070                                  .SBTTL  TYPE ROUTINE
 1071
 1072                                  ;;*************************************************************
 1073                                  ;*ROUTINE TO TYPE ASCIZ MESSAGE, MESSAGE MUST TERMINATE WITH A 0 BYTE,
 1074                                  ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
 1075                                  ;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER,
 1076                                  ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
 1077                                  ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER,
 1078                                  ;*
 1079                                  ;*CALL:
 1080                                  ;*1) USING A TRAP INSTRUCTION
 1081                                  ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
 1082                                  ;*OR
 1083                                  ;*      TYPE
 1084                                  ;*      MESADR
 1085                                  ;*
 1086
 1087  003424  105767  000265          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
 1088  003430  100002                          BPL     1$              ;;BR IF YES
 1089  003432  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
 1090  003434  000430                          BR      3$              ;;LEAVE
 1091  003436  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
 1092  003440  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
 1093  003444  122767  000001  175414          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
 1094  003452  001011                          BNE     6$              ;;NO,GO CHECK FOR APT CONSOLE
 1095  003454  132767  000100  175405          BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
 1096  003462  001405                          BEQ     6$              ;;NO,GO CHECK FOR CONSOLE
```



```
 1097  003464  010067  000004                  MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
 1098  003470  004767  000230                  JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
 1099  003474  000000                  61$:    .WORD   0               ;;MESSAGE ADDRESS
 1100  003476  132767  000040  175363  62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
 1101  003504  001003                          BNE     6C$             ;;YES,SKIP TYPE OUT
 1102  003506  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
 1103  003510  001005                          BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
 1104  003512  005726                          TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
 1105  003514  012600                  6C$:    MOV     (SP)+,R0        ;;RESTORE R0
 1106  003516  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
 1107  003522  000002                          RTI                     ;;RETURN
 1108  003524  122716  000011          4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
 1109  003530  001430                          BEQ     8$
 1110  003532  122716  000200                  CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
 1111  003536  001006                          BNE     5$
 1112  003540  005726                          TST     (SP)+           ;;POP  <CR><LF> EQUIV
 1113  003542  104401                          TYPE                    ;;TYPE A CR AND LF
 1114  003544  003225                          $CRLF
 1115  003546  105067  000130                  CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
 1116  003552  000755                          BR      2$              ;;GET NEXT CHARACTER
 1117  003554  004767  000056          5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
 1118  003560  126726  000130          6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS,?
 1119  003564  001350                          BNE     2$              ;;IF NO GO GET NEXT CHAR,
 1120  003566  016746  000120                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS, NEEDED
 1121                                                                  ;;AND THE NULL CHAR,
 1122  003572  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
 1123  003576  002770                          BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
 1124  003600  004767  000032                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
 1125  003604  105367  000072                  DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
 1126  003610  000770                          BR      7$              ;;LOOP
 1127
 1128                                  ;HORIZONTAL TAB PROCESSOR
 1129
 1130  003612  112716  000040          8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
 1131  003616  004767  000014          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
 1132  003622  132767  000007  000052          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
 1133  003630  001372                          BNE     9$              ;;TAB STOP
 1134  003632  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
 1135  003634  000724                          BR      2$              ;;GET NEXT CHARACTER
 1136  003636  105777  000044          $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
 1137  003642  100375                          BPL     $TYPEC
 1138  003644  116677  000002  000036          MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG,
 1139  003652  122766  000015  000002          CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
 1140  003660  001003                          BNE     1$              ;;BRANCH IF NO
 1141  003662  105067  000014                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
 1142  003666  000406                          BR      $TYPEX          ;;EXIT
 1143  003670  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
 1144  003676  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
 1145  003700  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
 1146  003702  000000                  $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
 1147  003704  000207                  $TYPEX: RTS     PC
 1148
 1149  003706  177564                  $TPS:   .WORD   177564          ;;TTY PRINTER STATUS REG, ADDRESS
 1150  003710  177566                  $TPB:   .WORD   177566          ;;TTY PRINTER BUFFER REG, ADDRESS
 1151  003712     000                  $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER FOR FILLS
 1152  003713     002                  $FILLS: .BYTE   2               ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
```

```
1113  003714  012                     $FILLC: .BYTE   12              ;;INSERT FILL CHARS, AFTER A "LINE FEED"
1114  003715  000                     $TPFLG: .BYTE   0               ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
1115                                  .SBTTL  APT COMMUNICATIONS ROUTINE
1116
1117                                  ;;**************************************************************
1118  003716  112767  000001  000236 $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
1119  003724  112767  000001  000226 $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
1120  003732  000403                        BR      $ATYC
1121  003734  112767  000001  000220 $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
1122  003742                         $ATYC:
1123  003742  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
1124  003744  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
1125  003746  105767  000206                TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
1126  003752  001450                        BEQ     5$              ;;IF NOT:  BR
1127  003754  122767  000001  175104        CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
1128  003762  001031                        BNE     3$              ;;IF NOT:  BR
1129  003764  132767  000100  175075        BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
1130  003772  001425                        BEQ     3$              ;;IF NOT:  BR
1131  003774  017600  000004                MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
1132  004000  062766  000002  000004        ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
1133  004006  005767  175034         1$:    TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
1134  004012  001375                        BNE     1$              ;;IF NOT:  WAIT
1135  004014  010067  175042                MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
1136  004020  105720                 2$:    TSTB    (R0)+           ;;FIND END OF MESSAGE
1137  004022  001376                        BNE     2$
1138  004024  166700  175032                SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
1139  004030  006200                        ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
1140  004032  010067  175026                MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
1141  004036  012767  000004  175002        MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
1142  004044  000413                        BR      5$
1143  004046  017667  000004  000016  3$:   MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
1144  004054  062766  000002  000004        ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
1145  004062  016746  173710                MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
1146  004066  004767  177332                JSR     PC,$TYPE        ;;CALL TYPE MACRO
1147  004072  000000                 4$:    .WORD   0
1148  004074                         5$:
1149  004074  105767  000062         10$:   TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
1150  004100  001416                        BEQ     12$             ;;IF NOT:  BR
1151  004102  005767  174760                TST     $ENV            ;;RUNNING UNDER APT?
1152  004106  001413                        BEQ     12$             ;;IF NOT:  BR
1153  004110  005767  174732         11$:   TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
1154  004114  001375                        BNE     11$             ;;IF NOT:  WAIT
1155  004116  017667  000004  174724        MOV     @4(SP),$FATAL   ;;GET ERROR #
1156  004124  062766  000002  000004        ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
1157  004132  005267  174710                INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
1158  004136  105067  000020         12$:   CLRB    $FFLG           ;;CLEAR FATAL FLAG
1159  004142  105067  000013                CLRB    $LFLG           ;;CLEAR LOG FLAG
1160  004146  105067  000006                CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
1161  004152  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
1162  004154  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
1163  004156  000207                        RTS     PC              ;;RETURN
1164  004160     000          $MFLG:  .BYTE   0               ;;MESSG. FLAG
1165  004161     000          $LFLG:  .BYTE   0               ;;LOG FLAG
1166  004162     000          $FFLG:  .BYTE   0               ;;FATAL FLAG
1167          004164                  .EVEN
1168          000200          APTSIZE=200
```



```
1169          000001          APTENV=001
1170          000100          APTSPOOL=100
1171          000040          APTCSUP=040
1172                          .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
1173
1174                          ;;**************************************************************
1175                          ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
1176                          ;*OCTAL (ASCII) NUMBER AND TYPE IT.
1177                          ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
1178                          ;*CALL:
1179                          ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
1180                          ;*      TYPOS                   ;;CALL FOR TYPEOUT
1181                          ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
1182                          ;*      .BYTE   M               ;;M=1 OR 0
1183                          ;*                                      ;;1=TYPE LEADING ZEROS
1184                          ;*                                      ;;0=SUPPRESS LEADING ZEROS
1185                          ;*
1186                          ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
1187                          ;*$TYPOS OR $TYPOC
1188                          ;*CALL:
1189                          ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
1190                          ;*      TYPON                   ;;CALL FOR TYPEOUT
1191                          ;*
1192                          ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
1193                          ;*CALL:
1194                          ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
1195                          ;*      TYPOC                   ;;CALL FOR TYPEOUT
1196
1197  004164  017646  000000         $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
1198  004170  116667  000001  000211        MOVB    1(SP),$0FILL    ;;LOAD ZERO FILL SWITCH
1199  004176  112667  000207                MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
1200  004202  062716  000002                ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
1201  004206  000406                        BR      $TYPON
1202  004210  112767  000001  000171 $TYPOC: MOVB    #1,$0FILL       ;;SET THE ZERO FILL SWITCH
1203  004216  112767  000006  000165        MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
1204  004224  112767  000005  000154 $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
1205  004232  010346                        MOV     R3,-(SP)        ;;SAVE R3
1206  004234  010446                        MOV     R4,-(SP)        ;;SAVE R4
1207  004236  010546                        MOV     R5,-(SP)        ;;SAVE R5
1208  004240  116704  000145                MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
1209  004244  005404                        NEG     R4
1210  004246  062704  000006                ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
1211  004252  110467  000132                MOVB    R4,$OMODE       ;;SAVE IT FOR USE
1212  004256  116704  000125                MOVB    $0FILL,R4       ;;GET THE ZERO FILL SWITCH
1213  004262  016605  000012                MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
1214  004266  005003                        CLR     R3              ;;CLEAR THE OUTPUT WORD
1215  004270  006105                 1$:    ROL     R5              ;;ROTATE MSB INTO "C"
1216  004272  000404                        BR      3$              ;;GO DO MSB
1217  004274  006105                 2$:    ROL     R5              ;;FORM THIS DIGIT
1218  004276  006105                        ROL     R5
1219  004300  006105                        ROL     R5
1220  004302  010503                        MOV     R5,R3
1221  004304  006103                 3$:    ROL     R3              ;;GET LSB OF THIS DIGIT
1222  004306  105367  000076                DECB    $OMODE          ;;TYPE THIS DIGIT?
1223  004312  100016                        BPL     7$              ;;BR IF NO
1224  004314  042703  177770                BIC     #177770,R3      ;;GET RID OF JUNK
```

```
1225  004320  001002                        BNE     4$              ;;TEST FOR 0
1226  004322  005704                        TST     R4              ;;SUPPRESS THIS 0?
1227  004324  001403                        BEQ     5$              ;;BR IF YES
1228  004326  005204                4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
1229  004330  052703  000060                BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
1230  004334  052703  000040        5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
1231  004340  110367  000040                MOVB    R3,8$           ;;SAVE FOR TYPING
1232  004344  104401  004404                TYPE    ,8$             ;;GO TYPE THIS DIGIT
1233  004350  105367  000032        7$:     DECB    $OCNT           ;;COUNT BY 1
1234  004354  003347                        BGT     2$              ;;BR IF MORE TO DO
1235  004356  002402                        BLT     6$              ;;BR IF DONE
1236  004360  005204                        INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
1237  004362  000744                        BR      2$              ;;GO DO THE LAST DIGIT
1238  004364  012605                6$:     MOV     (SP)+,R5        ;;RESTORE R5
1239  004366  012604                        MOV     (SP)+,R4        ;;RESTORE R4
1240  004370  012603                        MOV     (SP)+,R3        ;;RESTORE R3
1241  004372  016666  000002  000004        MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
1242  004400  012616                        MOV     (SP)+,(SP)
1243  004402  000002                        RTI                     ;;RETURN
1244  004404     000          8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
1245  004405     000                  .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
1246  004406     000          $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
1247  004407     000          $0FILL: .BYTE   0               ;;ZERO FILL SWITCH
1248  004410  000000          $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
1249                                .SBTTL  POWER DOWN AND UP ROUTINES
1250
1251                                ;;*************************************************************
1252                                ;POWER DOWN ROUTINE
1253  004412  012737  004552  000024 $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
1254  004420  012737  000340  000026         MOV     #340,@#PWRVEC+2 ;;PRIO:7
1255  004426  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
1256  004430  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
1257  004432  010246                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
1258  004434  010346                        MOV     R3,-(SP)        ;;PUSH R3 ON STACK
1259  004436  010446                        MOV     R4,-(SP)        ;;PUSH R4 ON STACK
1260  004440  010546                        MOV     R5,-(SP)        ;;PUSH R5 ON STACK
1261  004442  017746  174332                MOV     @SWR,-(SP)      ;;PUSH @SWR ON STACK
1262  004446  010667  000104                MOV     SP,$SAVR6       ;;SAVE SP
1263  004452  012737  004464  000024         MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
1264  004460  000000                        HALT
1265  004462  000776                        BR      .-2             ;;HANG UP
1266
1267                                ;;*************************************************************
1268                                ;POWER UP ROUTINE
1269  004464  012737  004552  000024 $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
1270  004472  016706  0C0060                MOV     $SAVR6,SP       ;;GET SP
1271  004476  005067  000054                CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
1272  004502  005267  000050        1$:     INC     $SAVR6          ;;WAIT FOR THE INC
1273  004506  001375                        BNE     1$              ;;OF  WORD
1274  004510  012677  174264                MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
1275  004514  012605                        MOV     (SP)+,R5        ;;POP STACK INTO R5
1276  004516  012604                        MOV     (SP)+,R4        ;;POP STACK INTO R4
1277  004520  012603                        MOV     (SP)+,R3        ;;POP STACK INTO R3
1278  004522  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
1279  004524  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
1280  004526  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
```



```
1281  004530  012737  004412  000024         MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
1282  004536  012737  000340  000026         MOV     #340,@#PWRVEC+2 ;;PRIO:7
1283  004544  104401                        TYPE                    ;REPORT THE POWER FAILURE
1284  004546  004560          $PWRMG: .WORD   $POWER          ;;POWER FAIL MESSAGE POINTER
1285  004550  000002                        RTI
1286  004552  000000          $ILLUP: HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
1287  004554  000776                  BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
1288  004556  000000          $SAVR6: 0                       ;;PUT THE SP HERE
1289  004560  005015  047520  042527 $POWER: .ASCIZ  <15><12>"POWER"
1290  004566  000122
1291                                .EVEN
1292                                .SBTTL  TRAP DECODER
1293
1294                                ;;*************************************************************
1295                                ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
1296                                ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
1297                                ;*OF THE DESIRED ROUTINE, THEN USING THE ADDRESS OBTAINED IT WILL
1298                                ;*GO TO THAT ROUTINE.
1299
1300  004570  010046                $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
1301  004572  016000  000002                MOV     2(SP),R0        ;;GET TRAP ADDRESS
1302  004576  005740                        TST     -(R0)           ;;BACKUP BY 2
1303  004600  111000                        MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
1304  004602  006300                        ASL     R0              ;;POSITION FOR INDEXING
1305  004604  016000  004621                MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
1306  004610  000200                        RTS     R0              ;;GO TO ROUTINE
1307
1308
1309                                ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
1310
1311  004612  011646                $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
1312  004614  016666  000004  000002         MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
1313  004622  000002                        RTI                     ;;RESTORE THE PSW
1314
1315                                .SBTTL  TRAP TABLE
1316
1317                                ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
1318                                ;*BY THE "TRAP" INSTRUCTION.
1319
1320                                ;       ROUTINE
1321                                ;       -------
1322  004624  004612          $TRPAD: .WORD   $TRAP2
1323  004626  003424                  $TYPE   ;;CALL=TYPE      TRAP+1(104401)  TTY TYPEOUT ROUTINE
1324  004630  004210                  $TYPOC  ;;CALL=TYPOC     TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
1325  004632  004164                  $TYPOS  ;;CALL=TYPOS     TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
1326  004634  004224                  $TYPON  ;;CALL=TYPON     TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
1327
1328
1329  004636  002640                  $RDCHR  ;;CALL=RDCHR     TRAP+5(104405)  TTY TYPEIN CHARACTER ROUTINE
1330  004640  002760                  $RDLIN  ;;CALL=RDLIN     TRAP+6(104406)  TTY TYPEIN STRING ROUTINE
1331  004642  003264                  $RDOCT  ;;CALL=RDOCT     TRAP+7(104407)  READ AN OCTAL NUMBER FROM TTY
1332
1333  004644  177777          TXCRC:  -1
1334  004646  000571                  571             ;M9301 - YA VERSION
1335  004650  000457                  457             ;M9301 - YB VERSION
1336  004652  000243                  243             ;M9301 - YC VERSION
```

```
1337  004654  000635                  635           ;M9400 - YA(OR YC) VERSION
1338  004656  000207                  207           ;M9301 - YF VERSION
1339  004660  000670                  670           ;M7942 - YB VERSION
1340  004662  000132                  132           ;M9301- YD VERSION
1341  004664  000374                  374           ;M9400 - YH (OR YK) VERSION
1342  004666  000630                  630           ;M9311 VERSION -- REVISION * (ORIGINAL REV.)
1343  004670  000536                  536           ;M9301 - YH VERSION
1344  004672  000752                  752           ;M9301 - YE VERSION
1345  004674  000633                  633           ;M9301 - YJ VERSION
1346  004676  000650                  650           ;M9400 - YN VERSION
1347  004700  000710                  710           ;M9311 VERSION -- REVISION A ( SECOND REV.)
1348  004702  177777                  -1
1349  004704  177777                  -1
1350  004706  177777                  -1
1351  004710  177777                  -1
1352  004712  177777                  -1
1353
1354  004714  177777            TXLPC1: -1
1355  004716  133725                  133725        ;M9301 - YA VERSION
1356  004720  017563                  17563         ;M9301 - YB VERSION
1357  004722  141744                  141744        ;M9301 - YC VERSION
1358  004724  047613                  47613         ;M9400 - YA(OR YC) VERSION
1359  004726  114175                  114175        ;M9301 - YF VERSION
1360  004730  146126                  146126        ;M7942 - YB VERSION
1361  004732  132161                  132161        ;M9301 - YD VERSION
1362  004734  143466                  143466        ;M9400 - YH(OR YK) VERSION
1363  004736  036751                  36751         ;M9311 VERSION-- REVISION * (ORIGINAL REV.)
1364  004740  125411                  125411        ;M9301 - YH VERSION
1365  004742  066246                  066246        ;M9301 - YE VERSION
1366  004744  132367                  132367        ;M9301 - YJ VERSION
1367  004746  030210                  30210         ;M9400 - YN VERSION
1368  004750  036743                  36743         ;M9311 VERSION--REVISION A ( SECOND REV.)
1369  004752  177777                  -1
1370  004754  177777                  -1
1371  004756  177777                  -1
1372  004760  177777                  -1
1373  004762  177777                  -1
1374  004764  177777                  -1
1375
1376  004766  177777            TDLN1:  -1
1377  004770  001000                  1000          ;M9301 - YA VERSION
1378  004772  001000                  1000          ;M9301 - YB VERISION
1379  004774  001000                  1000          ;M9301 - YC VERSION
1380  004776  001000                  1000          ;M9400 - YA(OR YC) VERSION
1381  005000  001000                  1000          ;M9301 - YF VERSION
1382  005002  004000                  4000          ;M7942 - YB VERSION
1383  005004  001000                  1000          ;M9301 - YD VERSION
1384  005006  001000                  1000          ;M9400 - YH(OR YK) VERSION
1385  005010  001000                  1000          ;M9311 VERSION REVISION * (ORIGINAL VERSION)
1386  005012  000734                  734           ;M9301 - YH VERSION
1387  005014  001000                  1000          ;M9301 - YE VERSION
1388  005016  001000                  1000          ;M9301 - YJ VERSION
1389  005020  001000                  1000          ;M9400 - YN VERSION
1390  005022  001000                  1000          ;M9311 VERSION- REVISION A (SECOND VERSION)
1391  005024  177777                  -1
1392  005026  177777                  -1
```



```
1393  005030  177777                  -1
1394  005032  177777                  -1
1395  005034  177777                  -1
1396
1397  005036  177777            TRMSA1: -1
1398  005040  173000                  173000        ;M9301 - YA VERSION
1399  005042  173000                  173000        ;M9301 - YB VERSION
1400  005044  173000                  173000        ;M9301 - YC VERSION
1401  005046  173000                  173000        ;M9400 - YA(OR YC) VERSION
1402  005050  173000                  173000        ;M9301 - YF VERSION
1403  005052  170000                  170000        ;M7942 - YB VERSION
1404  005054  173000                  173000        ;M9301 - YD VERSION
1405  005056  173000                  173000        ;M9400 - YH(OR YK) VERSION
1406  005060  163000                  163000        ;M9311 VERSION REVISION * ( ORIGINAL VERSION)
1407  005062  173000                  173000        ;M9301 - YH VERSION
1408  005064  173000                  173000        ;M9301 - YE VERSION
1409  005066  173000                  173000        ;M9301 - YJ VERSION
1410  005070  173000                  173000        ;M9301 - YN VERSION
1411  005072  163000                  163000        ;M9311 VERSION REVISION A (SECOND VERSION)
1412  005074  177777                  -1
1413  005076  177777                  -1
1414  005100  177777                  -1
1415  005102  177777                  -1
1416  005104  177777                  -1
1417
1418  005106  177777            TDLN2:  -1
1419  005110  001000                  1000          ;M9301 - YA VERSION
1420  005112  001000                  1000          ;M9301 - YB VERISION
1421  005114  001000                  1000          ;M9301 - YC VERSION
1422  005116  001000                  1000          ;M9400 -YA(OR YC) VERSION
1423  005120  001000                  1000          ;M9301 -YF VERSION
1424  005122  300000                  0             ;M7942 - YB VERSION
1425  005124  001000                  1000          ;M9301 - YD VERSION
1426  005126  001000                  1000          ;M9400 - YH(OR YK) VERSION
1427  005130  001000                  1000          ;M9311 VERSION REVISION * (ORIGINAL REV.)
1428  005132  000764                  764           ;M9301 - YH VERSION
1429  005134  001000                  1000          ;M9301 - YE VERSION
1430  005136  001000                  1000          ;M9301 - YJ VERSION
1431  005140  001000                  1000          ;M9400 - YN VERSION
1432  005142  001060                  1000          ;M9311 VERSION REVISION A (SECOND REV.)
1433  005144  177777                  -1
1434  005146  177777                  -1
1435  005150  177777                  -1
1436  005152  177777                  -1
1437  005154  177777                  -1
1438
1439  005156  177777            TRMSA2: -1
1440  005160  165000                  165000        ;M9301 - YA VERSION
1441  005162  165000                  165000        ;M9301 - YB VERSION
1442  005164  165000                  165000        ;M9301 - YC VERSION
1443  005166  165000                  165000        ;M9400 - YA(OR YC) VERSION
1444  005170  165000                  165000        ;M9301 - YF VERSION
1445  005172  000000                  0             ;M7942 - YB VERSION
1446
1447  005174  165000                  165000        ;M9301 - YD VERSION
1448  005176  165000                  165000        ;M9400 - YH(OR YK) VERSION
```

```
 1449  005200  166000                          166000          ;M9311 VERSION REVISION * ( ORIGINAL REV.)
 1450  005202  165000                          165000          ;M9301 - YH VERSION
 1451  005204  165000                          165000          ;M9301 - YE VERSION
 1452  005206  165000                          165000          ;M9301 - YJ VERSION
 1453  005210  165000                          165000          ;M9400 - YN VERSION
 1454  005212  166000                          166000          ;M9311 VERSION REVISION A ( SECOND REV.)
 1455  005214  177777                          -1
 1456  005216  177777                          -1
 1457  005220  177777                          -1
 1458  005222  177777                          -1
 1459  005224  177777                          -1
 1460
 1461  005226  177777                  TMSG:   -1
 1462  005230  006106                          MSG1
 1463  005232  006123                          MSG2
 1464  005234  006140                          MSG3
 1465  005236  006155                          MSG4
 1466  005240  006175                          MSG5
 1467  005242  006212                          MSG6            ;M7942 - YB VERSION
 1468  005244  006227                          MSG7
 1469  005246  006244                          MSG10
 1470  005250  006270                          MSG11
 1471  005252  006347                          MSG12           ;M9301 - YH VERSION
 1472  005254  006364                          MSG13           ;M9301 - YE VERSION
 1473  005256  006401                          MSG14           ;M9301 - YJ VERSION
 1474  005260  006416                          MSG15           ;M9400 - YN VERSION
 1475  005262  006433                          MSG16           ; M9311 REVISION A ( SECOND REV.)
 1476  005264  177777                          -1
 1477  005266  177777                          -1
 1478  005270  177777                          -1
 1479  005272  177777                          -1
 1480  005274  177777                          -1
 1481
 1482
 1483  005276  005015  030122  020115  TITL:   .ASCIZ  <15><12>/ROM TEST/
 1484  005304  042524  052123     000
 1485  005311     015  052012  050131  GETCRC: .ASCIZ  <15><12>/TYPE CRC VALUE:  /
 1486  005316  020105  051103  020103
 1487  005324  040526  052514  035105
 1488  005332  020040     000
 1489  005335     015  052012  050131  GETLPC: .ASCIZ  <15><12>/TYPE LPC VALUE:  /
 1490  005342  020105  050114  020103
 1491  005350  040526  052514  035105
 1492  005356  020040     000
 1493  005361     015  042412  050130  EXCRMG: .ASCIZ  <15><12>/EXPECTED CRC =  /
 1494  005366  041505  042524  020104
 1495  005374  051103  020103  020075
 1496  005402  000040
 1497  005404  005015  042412  050130  EXLPMG: .ASCIZ  <15><12><12>/EXPECTED LPC =  /
 1498  005412  041505  042524  020104
 1499  005420  050114  020103  020075
 1500  005426  000040
 1501  005430  005015  047503  050115  ACCRMG: .ASCIZ  <15><12>/COMPUTED CRC =  /
 1502  005436  052125  042105  041440
 1503  005444  041522  036440  020040
 1504  005452     000
```



```
 1505  005453     015  041412  046517  ACLPMG: .ASCIZ  <15><12>/COMPUTED LPC =  /
 1506  005460  052520  042524  020104
 1507  005466  050114  020103  020075
 1508  005474  000040
 1509  005476  005015  042412  042116  EOTST:  .ASCIZ  <15><12><12>/END OF TEST/
 1510  005504  047440  020106  042524
 1511  005512  052123     000
 1512  005515     015  052012  050131  SA1:    .ASCIZ  <15><12>/TYPE STARTING ADDR. OF 1ST ROM ADDR. SPACE:  /
 1513  005522  020105  052123  051101
 1514  005530  044524  043516  040440
 1515  005536  042104  027122  047440
 1516  005544  020106  051461  020124
 1517  005552  047522  020115  042101
 1518  005560  051104  020056  050123
 1519  005566  041501  035105  020040
 1520  005574     000
 1521  005575     015  052012  050131  SA2:    .ASCIZ  <15><12>/TYPE STARTING ADDR. OF 2ND ROM ADDR. SPACE:  /
 1522  005602  020105  052123  051101
 1523  005610  044524  043516  040440
 1524  005616  042104  027122  047440
 1525  005624  020106  047062  020104
 1526  005632  047522  020115  042101
 1527  005640  051104  020056  050123
 1528  005646  041501  035105  020040
 1529  005654     000
 1530  005655     015  052012  050131  SIZE1:  .ASCIZ  <15><12>/TYPE LENGTH (BYTES) OF 1ST ROM ADDR. SPACE:  /
 1531  005662  020105  042514  043516
 1532  005670  044124  024040  054502
 1533  005676  042524  024523  047440
 1534  005704  020106  051461  020124
 1535  005712  047522  020115  042101
 1536  005720  051104  020056  050123
 1537  005726  041501  035105  020040
 1538  005734     000
 1539  005735     015  052012  050131  SIZE2:  .ASCIZ  <15><12>/TYPE LENGTH (BYTES) OF 2ND ROM ADDR. SPACE:  /
 1540  005742  020105  042514  043516
 1541  005750  044124  024040  054502
 1542  005756  042524  024523  047440
 1543  005764  020106  047062  020104
 1544  005772  047522  020115  042101
 1545  006000  051104  020056  050123
 1546  006006  041501  035105  020040
 1547  006014     000
 1548  006015     015  000012          CARLF:  .ASCIZ  <15><12>
 1549  006020  020040     000          SP2:    .ASCIZ  /  /
 1550  006023     015  040412  042104  TYPHDR: .ASCIZ  <15><12>/ADDRESS               DATA/
 1551  006030  042522  051523  020040
 1552  006036  020040  020040  020040
 1553  006044  020040  020040  020040
 1554  006052  042040  052101  000101
 1555  006060  020072  000040          COLON:  .ASCIZ  /:  /
 1556  006064  005015  047125  047113  AUTERM: .ASCIZ  <15><12>/UNKNOWN MODULE /
 1557  006072  053517  020116  047515
 1558  006100  052504  042514  000040
 1559
 1560  006106  005015  034515  030063  MSG1:   .ASCIZ  <15><12>/M9301 - YA/
```

```
  1561  006114  020061  020055  040531
  1562  006122     000
  1563  006123     015  046412  031471  MSG2:   .ASCIZ  <15><12>/M9301 - YB/
  1564  006130  030460  026440  054440
  1565  006136  000102
  1566  006140  005015  034515  030063  MSG3:   .ASCIZ  <15><12>/M9301 - YC/
  1567  006146  020061  020055  041531
  1568  006154     000
  1569  006155     015  046412  032071  MSG4:   .ASCIZ  <15><12>/M9400 - YA,YC/
  1570  006162  030060  026440  054440
  1571  006170  026101  041531     000
  1572  006175     015  046412  031471  MSG5:   .ASCIZ  <15><12>/M9301 - YF/
  1573  006202  030460  026440  054440
  1574  006210  000106
  1575  006212  005015  033515  032071  MSG6:   .ASCIZ  <15><12>/M7942 - YB/
  1576  006220  020062  020055  041131
  1577  006226     000
  1578  006227     015  046412  031471  MSG7:   .ASCIZ  <15><12>/M9301 - YD/
  1579  006234  030460  026440  054440
  1580  006242  000104
  1581  006244  005015  034515  030064  MSG10:  .ASCIZ  <15><12>/M9400 - YH(OR YK)/
  1582  006252  020060  020055  044131
  1583  006260  047450  020122  045531
  1584  006266  000051
  1585  006270  005015  034515  030463  MSG11:  .ASCIZ  <15><12>/M9311 REVISION  * (LOCATION E=5 = 23=524A9) /
  1586  006276  020061  042522  044526
  1587  006304  044523  047117  020040
  1588  006312  020052  046050  041517
  1589  006320  052101  047511  020116
  1590  006326  026505  020065  020075
  1591  006334  031462  032455  032062
  1592  006342  034501  020051     000
  1593  006347     015  046412  031471  MSG12:  .ASCIZ  <15><12>/M9301 - YH/
  1594  006354  030460  026440  054440
  1595  006362  000110
  1596  006364  005015  034515  030063  MSG13:  .ASCIZ  <15><12>/M9301 - YE/
  1597  006372  020061  020055  042531
  1598  006400     000
  1599  006401     015  046412  031471  MSG14:  .ASCIZ  <15><12>/M9301 - YJ/
  1600  006406  030460  026440  054440
  1601  006414  000112
  1602  006416  005015  034515  030064  MSG15:  .ASCIZ  <15><12>/M9400 - YN/
  1603  006424  020060  020055  047131
  1604  006432     000
  1605  006433     015  046412  031471  MSG16:  .ASCIZ <15><12>/M9311 REVISION A (LOCATION E=5 = 23=688A9) /
  1606  006440  030461  051040  053105
  1607  006446  051511  047511  020116
  1608  006454  020101  046050  041517
  1609  006462  052101  047511  020116
  1610  006470  026505  020065  020075
  1611  006476  031462  033055  034070
  1612  006504  034501  020051     000
  1613          000001                  .END
```



```
ABASE = 000000            569
ACCRMG  005430            707    1501#
ACDW1 = 000000            569
ACDW2 = 000000            569
ACLPMG  005453            725    1505#
ACPUOP= 000000            569     584
ACTCRC  001022            529#    676*    699     708     750     780*    853
ACTLPC  001024            530#    677*    718     726     805     812*
ADDW0 = 000000            569
ADDW1 = 000000            569
ADDW10= 000000            569
ADDW11= 000000            569
ADDW12= 000000            569
ADDW13= 000000            569
ADDW14= 000000            569
ADDW15= 000000            569
ADDW2 = 000000            569
ADDW3 = 000000            569
ADDW4 = 000000            569
ADDW5 = 000000            569
ADDW6 = 000000            569
ADDW7 = 000000            569
ADDW8 = 000000            569
ADDW9 = 000000            569
ADEVCT= 000000            569     575
ADEVM = 003000            569
AENV  = 000000            569     580
AENVM = 000000            569     581
AFATAL= 000000            569     572
AMADR1= 000000            569
AMADR2= 000000            569
AMADR3= 000000            569
AMADR4= 000000            569
AMAMS1= 000000            569
AMAMS2= 000000            569
AMAMS3= 000000            569
AMAMS4= 000000            569
AMSGAD= 000000            569     577
AMSGLG= 000000            569     578
AMSGTY= 000000            569     571
AMTYP1= 000000            569
AMTYP2= 000000            569
AMTYP3= 000000            569
AMTYP4= 000000            569
APASS = 000000            569     574
APRIOR= 000000            569
APTCSU= 000040           1060    1171#
APTENV= 000001           1053    1127    1169#
APTSIZ= 000200            621    1168#
APTSPC= 000100           1055    1129    1170#
ASWREG= 000000            569     582
ATESTN= 000000            569     573
AUNIT = 000000            569     576
AUSWR = 000000            569     583
AUTACT  002522            698     849#
AUTERM  006064            865    1556#
```

```
AUTERR  002612            852     864#
AUT1    002524            850#    854
AUT2    002570            857#    859#
AUT3    002572            856     860#
AVECT1= 000000            569
AVECT2= 000000            569
BIT0  = 000001            485#
BIT00 = 000001            475#    485
BIT01 = 000002            474#    484
BIT02 = 000004            473#    483
BIT03 = 000010            472#    482
BIT04 = 000020            471#    481
BIT05 = 000040            470#    480
BIT06 = 000100            469#    479
BIT07 = 000200            468#    478
BIT08 = 000400            467#    477
BIT09 = 001000            466#    476
BIT1  = 000002            484#
BIT10 = 002000            465#
BIT11 = 004000            464#
BIT12 = 010000            463#
BIT13 = 020000            462#
BIT14 = 040000            461#
BIT15 = 100000            460#
BIT2  = 000004            483#
BIT3  = 000010            482#
BIT4  = 000020            481#
BIT5  = 000040            480#
BIT6  = 000100            479#
BIT7  = 000200            478#
BIT8  = 000400            477#
BIT9  = 001000            476#
BPTVEC= 000014            492#
CARLF   006015            832     837     1548#
CHECK   001510            642     644     646     673#
CH0     001566            679     686#
CH1     001644            688     697     699#
CK1     001736            700     713     718#    862
CK2     002030            719     731     736#
CLLAST  002204            773     775#
CLP0    002252            784#    790
CLP1    002264            785     787#
CLP2    002316            794     796#
CL0     002102            751#    757     774
CL1     002146            765#
CL2     002170            762     771#
CL3     002122            753     758#
COLON   006060            841    1555#
CR    = 000015            400#   1099    1109
CRC     002074            681     689     750#
CRLF  = 000200            401#   1070    1109
DATLN1  001006            523#    638*    664*    678     683     818
DATLN2  001012            525#    640*    672*    687     691     823
DDISP = 177570            407#    609
DISPLA  001002            521#    609*    617*
DISPRE  000174            508#    617
```



```
DS%R  = 177570            406#    608
EMTVEC= 000030            495#
ENDOT   002446            825     827#
EOTST   005476            739     928     1509#
ERRVEC= 000004            488#    606     607*    618#
EXCRC   001016            527#    637*    652*    699     704     860*
EXCRMG  005361            703    1493#
EXLPC   001020            528#    636*    656*    718     722     861*
EXLPMG  005404            721    1497#
GETCRC  005311            650    1485#
GETIN   001370            634     643#
GETLPC  005335            654    1489#
GET2    001440            648     657#
GNS   = ****** U          507    1323    1324    1325    1326    1329    1330    1331
GOAGIN  002070            742     748#
HT    = 000011            398#   1068    1109
IOTVEC= 000020            493#
LF    = 000012            399#   1103    1109
LPC     002344            685     693     805#
LPC1    002352            806#    811
LPC2    002366            808     810#
MSG1    006106           1462    1560#
MSG10   006244           1469    1581#
MSG11   006270           1470    1585#
MSG12   006347           1471    1593#
MSG13   006364           1472    1596#
MSG14   006401           1473    1599#
MSG15   006416           1474    1602#
MSG16   006433           1475    1605#
MSG2    006123           1463    1563#
MSG3    006140           1464    1566#
MSG4    006155           1465    1569#
MSG5    006175           1466    1572#
MSG6    006212           1467    1575#
MSG7    006227           1468    1578#
PARCNT  001026            531#    782*    786*    793
PARITY  002242            758     782#
PIPQ  = 177772            405#
PIROVE= 000240            499#
PR0   = 000000            422#
PR1   = 000040            423#
PR2   = 000100            424#
PR3   = 000140            425#
PR4   = 000200            426#
PR5   = 000240            427#
PR6   = 000300            428#
PR7   = 000340            429#
PS    = 177776            402#    403     516#
PSW   = 177776            403#
PWRVEC= 000024            494#    602*    603*   1253*   1254*   1263*   1269*   1281*   1282*
RDCHR = 104405            921    1329#
RDLIN = 104406            996    1330#
RDOCT = 104407            651     655     659     663     667     671    1331#
RESTRT  001274            626     630#    748
RESVEC= 000010            489#
ROMSA1  001004            522#    639*    660*    680     682     817
```

```
ROMSA2  001010            524*    641*    668*    686     690     822
SA1     005515            658    1512#
SA2     005575            666    1521#
SIZE1   005655            662    1530#
SIZE2   005735            670    1539#
SP2     006020            845    1549#
STACK = 001100            393#
START   001076            512     514     594#
STKLMT= 177774            404#
ST2     001320            635#
SWR     001000            520*    608*    610     616*    623*    632*    633     1261    1274*
SWREG   000176            509#    616     632
SW0   = 000001            457#
SW00  = 000001            447#    457
SW01  = 000002            446#    456
SW02  = 000004            445#    455
SW03  = 000010            444#    454
SW04  = 000020            443#    453
SW05  = 000040            442#    452
SW06  = 000100            441#    451
SW07  = 000200            440#    450
SW08  = 000400            439#    449
SW09  = 001000            438#    448
SW1   = 000002            456#
SW10  = 002000            437#
SW11  = 004000            436#
SW12  = 010000            435#
SW13  = 020000            434#
SW14  = 040000            433#
SW15  = 100000            432#
SW2   = 000004            455#
SW3   = 000010            454#
SW4   = 000020            453#
SW5   = 000040            452#
SW6   = 000100            451#
SW7   = 000200            450#
SW8   = 000400            449#
SW9   = 001000            448#
TBITVE= 000014            490#
TDLN1   004766            638    1376#
TDLN2   005106            640    1418#
TITL    005276            628    1483#
TKVEC = 000060            497#
TMSG    005226            851     857    1461#
TPVEC = 000064            498#
TRAPVE= 000034            496#    517#    600*    601*
TRMSA1  005036            639    1397#
TRMSA2  005156            641    1439#
TRTVEC= 000014            491#
TXCRC   004644            637     853     860    1333#
TXLPC   004714            636     861    1354#
TYP     002454            821     826     831#
TYPE  = 104401            627     649     653     657     661     665     669     702     706     720     724     738     815
                          827     831     836     840     844     858     864     928     934     939     943     948     949
                          951     954     958    1025    1027    1073    1232    1283    1323#
TYPHDR  006023            816    1550#
```



```
TYPOC = 104402            705     709     723     727     839     843    1324#
TYPON = 104404           1326#
TYPOS = 104403           1325#
TYPOUT  001030            511*    513*    532#    647     673
TYPROM  002402            675     815#
TYPR1   002426            820     822#
TYP0    002462            834#    847
TYP2    002504            835     842#
TYP3    002470            833     836#
XOR     002320            759     798#    809
XORS    001014            526*    750*    771*    775     776*    777*    779*    780     799     800*    801*    802     805*
                          812
$APTHD  001032            553     559#
$ASTAT= ****** U         1149    1164
$ATYC   003742           1120    1122#
$ATY1   003716           1118#
$ATY3   003724           1058    1119#
$ATY4   003734           1121#
$CHARC  003702           1075*   1085*   1092    1101*   1106#
$CKSWR= ****** U         1329
$CMTAG= ****** U          598
$CNTLG  003235            972#
$CNTLU  003230            943     971#
$CPUOP  001074            584#
$CRLF   003225            948     969#    1030    1074    1115
$DEVCT  001056            575#
$ENDAD  002060            540     625     710     728     736     744#    855
$ENV    001066            580#    645     694     712     730    1053    1127    1151
$ENVM   001067            581#    621    1055    1060    1129
$ETABL  001066            579#
$ETEND  001076            565     591#
$FATAL  001050            572#    594*    714*    732*    866*   1155*
$FFLG   004162           1118*   1121*   1149    1158*   1166#
$FILLC  003714           1078    1113#
$FILLS  003713           1112#
$GTSWR= ****** U         1328
$HIBTS  001032            560#
$HIOCT  003422           1018*   1029#
$ILLUP  004552           1253    1269    1286#
$LF     003226            958     970#    1030    1115
$LFLG   004161           1159*   1165#
$MAIL   001046            561     565     570#    620    1053
$MBADR  001034            561#
$MFLG   004160           1119*   1125    1160*   1164#
$MNEW   003253            975#
$MSGAD  001062            577#   1135*   1138
$MSGLG  001064            578#   1140#
$MSGTY  001046            571#    595*    715*    733*    867*   1133    1141*   1153    1157*
$MSWR   003242            973#
$NULL   003712           1080    1111#
$OCNT   004406           1204*   1233*   1246#
$OMODE  004410           1199*   1203*   1208    1211*   1222*   1248#
$PASS   001054            574#    620*    740*
$PASTM  001040            563#
$POWEP  004560           1284    1289#
$PWRDN  004412            602    1253#   1281
```

```
$PWRMG  004546          1284#
$PWRUP  004464          1263    1269#
$QUES   003224           951     968#   1027    1030    1115
$RDCHR  002640           888#   1329
$RDDEC= ****** U        1332
$RDLIN  002760           916#   1330
$RDOCT  003264           991#   1331
$RDSZ = 000010           909#
$R2A  = ****** U        1312
$SAVRE= ****** U        1332
$SAVR6  004556          1262*   1270    1271*   1272*   1288#
$SETUP= 000014           510#    599     600     602     604     877     977
$STUP = 177777           510#
$SVPC = 001032           538#    543
$SWR  = 000000           389#   1285
$SWREG  001070           592#    623
$TESTN  001052           573#    596*
$TKB    002636           874#    892     898
$TKS    002634           873#    890     896
$TPB    003710          1098*   1110#
$TPFLG  003715          1047    1114#
$TPS    003706          1096    1109#
$TRAP   004570           600    1300#
$TRAP2  004612          1311#   1322
$TRP  = 000010          1315#   1324#   1325#   1326#   1327#   1329    1330#   1331#   1332#
$TRPAD  004624          1305    1322#
$TSTM   001036           562#
$TTYIN  003214           918     919     931     949     963     967#
$TYPBN= ****** U        1327
$TYPDS= ****** U        1327
$TYPE   003424          1047#   1146    1315    1323
$TYPEC  003636          1077    1084    1091    1096#   1097
$TYPEX  003704          1102    1104    1107#
$TYPOC  004210          1202#   1324
$TYPON  004224          1201    1204#   1326
$TYPOS  004164          1197#   1325
$UNIT   001060           576#
$UNITM  001042           564#
$USWR   001072           593#
$0FILL  004407          1198*   1202*   1212    1247#
.     = 006511           500#    503#    507#    510#    519#    538     539#    541#    543#    549     550#    552#    554#
                         873     874     967#    968     969     970     977    1030    1109    1110    1111    1112    1113
                        1114    1115    1167#   1265    1287
.$ASTA= ****** U        1119    1122
.$X   = 001032           549#    554

. ABS.  006511     000

 ERRORS DETECTED:  0

 CZM9AE,BIN,CZM9AE,SEQ/CRF/SOL/NL:TOC=SYSMAC,SML,CZM9AE,P11
 RUN-TIME:  38 33 1 SECONDS
 RUN-TIME RATIO: 356/74=4.7
 CORE USED:  32K  (63 PAGES)
```