M9301YE/YJ

BOOTSTRAP TERMINATOR
MD-11-DZM9A-D

EP-DZM9A-D-DL-A

FICHE 1 OF 1

AUG 1977
digital
MADE IN USA

300000 .REPT 0

IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DZM9A-D-D

PRODUCT NAME: BOOTSTRAP/TERMINATOR (M9301, M9400)

PROGRAM DATE: MAY 27,1977

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.

2.3 PROGRAM OPTIONS

THE PROGRAM AUTOMATICALLY CHECKS FOR THE PRESENCE OF
A HARDWARE SWITCH REGISTER. IF NO RESPONSE IS FOUND WHEN
ADDRESSING THE HARDWARE SWR (177570), THE ADDRESS
OF THE SOFTWARE SWR (176) IS SUBSTITUTED.

FOR PROCESSORS WITH NO HARDWARE SWITCH REGISTER, THE OPERATOR
SHOULD SET THE DESIRED SWITCH VALUE IN LOCATION 176

WARNING... IN ORDER TO ALLOW TESTING OF M7942-YB BOARDS ON THE VT71.
IF LOACTION 176 IS SET TO 6, THE SOFTWARE SWITCH REGISTER WILL BE
USED REGARDLESS OF HARDWARE SWITCH REGISTER AVAILABILITY.

2.3.1 SWITCH SELECTION

THE SWITCH REGISTER (HARDWARE OR SOFTWARE) IS USED TO SELECT
THE VERSION OF THE ROM BOOTSTRAP TO BE TESTED
ACCORDING TO THE FOLLOWING TABLE. NOTE: THESE SETTINGS ARE
OCTAL NUMBERS. THEY ARE NOT PARTICULAR SWITCHES SET TO A
ONE. FOR EXAMPLE, TO SELECT THE M9301-YH VERSION, SET
SWITCHES #3 AND #1 IN THE SWITCH REGISTER. THIS
CORRESPONDS TO AN OCTAL 12.

SWR	MODULE VERSION
1	M9301-YA
2	M9301-YB
3	M9301-YC
4	M9400-YA (OR YC)
5	M9301-YF
6	M7942-YB
7	M9301-YD
10	M9400-YH (OR YK)
11	M9311
12	M9301-YH
13	M9301-YE
14	M9301-YJ

IF THE CRC AND LPC FOR NEW VERSIONS ARE KNOWN BUT
NOT IN THE ABOVE TABLE, SET THE SWITCH REGISTER TO ZERO
AND ANSWER THE TELETYPE DIALOG.

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.

```
366
367
368
369
                                        6.0     LISTING
                                                -------
                                        .ENDR
```

```
                        .ENABLE ABS
                        .LIST   ME
                        .NLIST  MC,MD,CND
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
                                        .EQUIV  SW04,SW4
                                        .EQUIV  SW03,SW3
                                        .EQUIV  SW02,SW2
                                        .EQUIV  SW01,SW1
                                        .EQUIV  SW00,SW0

                                        ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
                        100000          BIT15=  100000
                        040000          BIT14=  40000
                        020000          BIT13=  20000
                        010000          BIT12=  10000
                        004000          BIT11=  4000
                        002000          BIT10=  2000
                        001000          BIT09=  1000
                        000400          BIT08=  400
                        000200          BIT07=  200
                        000100          BIT06=  100
                        000040          BIT05=  40
                        000020          BIT04=  20
                        000010          BIT03=  10
                        000004          BIT02=  4
                        000002          BIT01=  2
                        000001          BIT00=  1
                                        .EQUIV  BIT09,BIT9
                                        .EQUIV  BIT08,BIT8
                                        .EQUIV  BIT07,BIT7
                                        .EQUIV  BIT06,BIT6
                                        .EQUIV  BIT05,BIT5
                                        .EQUIV  BIT04,BIT4
                                        .EQUIV  BIT03,BIT3
                                        .EQUIV  BIT02,BIT2
                                        .EQUIV  BIT01,BIT1
                                        .EQUIV  BIT00,BIT0

                                        ;*BASIC "CPU" TRAP VECTOR ADDRESSES
                        000004          ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
                        000010          RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
                        000014          TBITVEC=14              ;;"T" BIT
                        000014          TRTVEC= 14              ;;TRACE TRAP
                        000014          BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
                        000020          IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
                        000024          PWRVEC= 24              ;;POWER FAIL
                        000030          EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
                        000034          TRAPVEC=34              ;;"TRAP" TRAP
```

```
                        000060                  TKVEC=  60              ;;TTY KEYBOARD VECTOR
                        000064                  TPVEC=  64              ;;TTY PRINTER VECTOR
                        000240                  PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR
                        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 $ENDAD IN .SEOP
                        000052                          .=52
000052  000000                                          .WORD   0               ;;2)SET LOC.52 TO ZERO
                        001032                          .=$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 TC POINT TO START OF PROGRAM
000024  000200                                          200     ;;FOR APT START UP
                        000044                          .=44    ;;POINT TO APT INDIRECT ADDRESS PNTR.
```

```
                        $APTHDR ;;POINT TO APT HEADER BLOCK
                        .=.$X   ;;RESET LOCATION COUNTER
                ;;*****************************************************************
                ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
                ;INTERFACE SPEC.

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
```

```
  599  001170  012767  177570  177604          MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
  600  001176  022777  177777  177574          CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
  601  001204  001012                          BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
  602                                                                  ;;AND  THE HARDWARE SWR IS NOT = -1
  603  001206  000403                          BR      65$             ;;BRANCH IF NO TIMEOUT
  604  001210  012716  001216          64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
  605  001214  000002                          RTI
  606  001216  012767  000176  177554  65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
  607  001224  012767  000174  177550          MOV     #DISPREG,DISPLAY
  608  001232  012637  000004          66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR
  609
  610  001236  005067  177612                  CLR     $PASS           ;;CLEAR PASS COUNT
  611  001242  132767  000200  177617          BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
  612  001250  001403                          BEQ     67$             ;;YES,USE NON-APT SWITCH
  613  001252  012767  001070  177520          MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
  614  001260                          67$:
  615  001260  026727  176556  002060          CMP     42,#$ENDAD              ;ACT AUTO MODE?
  616  001266  001402                          BEQ     RESTRT                  ;IF SO: BR
  617  001270  104401                          TYPE
  618  001272  005260                          TITL
  619
  620  001274  123727  000176  000006  RESTRT: CMPB    @#176,#6                ;IS SOFTWARE SWITCH REGISTER =6?
  621  001302  001003                          BNE     1$                      ;IF NOT, TEST NORMALY
  622  001304  012767  000176  177466          MOV     #SWREG,SWR              ;IF SO USE THE SOFTWARE SWITCH REG
  623  001312  017700  177462          1$:     MOV     @SWR,R0                 ;GET SWR
  624  001316  001424                          BEQ     GETIN                   ;IF ZERO: GET INPUT
  625  001320  006300                  ST2:    ASL     R0
  626  001322  016067  004712  177470          MOV     TXLPC(R0),EXLPC         ;FETCH EXPECT. LPC
  627  001330  016067  004644  177460          MOV     TXCRC(R0),EXCRC         ;FETCH EXPECTED CRC
  628  001336  016067  004762  177442          MOV     TDLN1(R0),DATLN1        ;FETCH 1ST LENGTH
  629  001344  016067  005030  177432          MOV     TRMSA1(R0),ROMSA1       ;FETCH 1ST STARTING ADDR.
  630  001352  016067  005076  177422          MOV     TDLN2(R0),DATLN2                ;FETCH 2ND LENGTH
  631  001360  016067  005144  177412          MOV     TRMSA2(R0),ROMSA2       ;FETCH 2ND STARTING ADDR
  632  001366  000450                          BR      CHECK                   ;GO COMPUTE
  633  001370  005767  176446          GETIN:  TST     42                      ;UNDER ACT AUT ACCEPT?
  634  001374  001045                          BNE     CHECK                   ;IF SO; BR USE DEFAULT PARAMETERS
  635  001376  122767  000001  177462          CMPB    #1,$ENV                 ;UNDER APT?
  636  001404  001441                          BEQ     CHECK                   ;IF SO: BR
  637  001406  005767  177416                  TST     TYPOUT                  ;ROM TYPE OPTION
  638  001412  001012                          BNE     GET2                    ;IF SO:  BR
  639  001414  104401                          TYPE
  640  001416  005273                          GETCRC
  641  001420  104407                          RDOCT
  642  001422  012667  177370                  MOV     (SP)+,EXCRC             ;STORE EXPECT. CRC
  643  001426  104401                          TYPE                            ;TYPE LPC INPUT REQUEST
  644  001430  005317                          GETLPC
  645  001432  104407                          RDOCT
  646  001434  012667  177360                  MOV     (SP)+,EXLPC             ;STORE EXPECTED LPC
  647  001440  104401          GET2:   TYPE
  648  001442  005477                          SA1                             ;REQUEST 1ST ADDRESS SPACE
  649  001444  104407                          RDOCT                           ;INPUT SA
  650  001446  012667  177332                  MOV     (SP)+,ROMSA1
  651  001452  104401                          TYPE                            ;REQUEST LENGTH OF 1ST ADDR. SPACE
  652  001454  005637                          SIZE1
  653  001456  104407                          RDOCT                           ;INPUT LENGTH
  654  001460  012667  177322                  MOV     (SP)+,DATLN1
```

```
                        INITIALIZE THE COMMON TAGS

001464  104401                  TYPE                    ;REQUEST START ADDR. FOR 2ND SPACE
001466  005557                  SA2
001470  104407                  RDOCT                   ;INPUT SA
001472  012667  177312          MOV     (SP)+,RO=SA2
001476  104401                  TYPE                    ;REQUEST LENGTH OF 2ND SPACE
001500  005717                  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$:     CLR     ACTCRC          ;CLEAR STORAGE FOR ACTUAL CRC
001526  005067  177272          CLR     ACTLPC          ;CLEAR STORAGE FOR ACTULAL LPC
001532  016700  177250          MOV     DATLN1,RO       ;SET LENGTH OF 1ST ROM SPACE
001536  001413                  BEQ     CH0             ;IF NO VERSION SELECTED:  BR
001540  016701  177240          MOV     ROMSA1,R1       ;POINT TO START OF 1ST ROM SPACE
001544  004767  000324          JSR     PC,CRC          ;COMPUTE FIRST HALF OF CRC
001550  016701  177230          MOV     ROMSA1,R1       ;POINT TO START OF 1ST ROM ADDRS.
001554  016700  177226          MOV     DATLN1,RO       ;SET LENGTH OF 1ST ROM ADDRS.
001560  006200                  ASR     RO              ;CONVERT TO WORDS
001562  004767  000556          JSR     PC,LPC          ;COMPUTE FIRST HALF OF CRC
001566  016701  177216  CH0:    MOV     ROMSA2,R1       ;POINT TO 2ND ROM ADDRS.
001572  016700  177214          MOV     DATLN2,RO       ;SET LENGTH OF 2ND ROM ADDRS.
001576  001422                  BEQ     CH1             ;BR IF THIS SPACE NOT USED
001600  004767  000270          JSR     PC,CRC          ;COMPUTE REMAINDER OF CRC
001604  016701  177200          MOV     ROMSA2,R1       ;POINT TO START OF 2ND ROM ADDRS.
001610  016700  177176          MOV     DATLN2,RO       ;SET LENGTH OF 2ND ROM ADDRS.
001614  006200                  ASR     RO              ;CONVERT TO WORDS
001616  004767  000522          JSR     PC,LPC          ;COMPUTE REMAINDER OF LPC
001622  122767  000001  177236  CMPB    #1,$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  005343                  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  005412                  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=ACTURAL LPC
001744  001431                  BEQ     CK2             ;IF SO:  BR
001746  104401                  TYPE                    ;TYPE LPC ERROR MESSG.
```

```
001750  005366                          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  005435                          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  005460                          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                  CLO:    MOVB    (R1),R4                 ;GET CHAR.
002104  022701  173024                  CMP     #173024,R1              ;LOCATION EFFECTED BY SWITCHES
002110  001004                          BNE     CL3                     ;IF NOT:  BR
002112  005300                          DEC     R0                      ;FIX COUNTERS
002114  005300                          DEC     R0
002116  005721                          TST     (R1)+                   ;FIX POINTER
002120  000770                          BR      CLO                     ;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,R5                   ;SAVE CHAR
002150  042705  177703                  BIC     #177703,R5
002154  005105                          COM     R5
002156  042705  177703                  BIC     #177703,R5
002162  042704  000074                  BIC     #74,R4
002166  050504                          BIS     R5,R4
002170  010467  176620          CL2:    MOV     R4,XORS
002174  005300                          DEC     R0
002176  001402                          BEQ     CLLAST                  ;IF LAST CAR.:  BR
002200  000167  177676                  JMP     CLO                     ;GET NEXT CHAR.
002204  016704  176604          CLLAST: MOV     XORS,R4
002210  005167  176600                  COM     XORS
```

```
  767  002214  042767  177050  176572          BIC     #177050,XORS
  768  002222  042704  177727                  BIC     #177727,R4              ;COMPLEMENT ALL BUT BITS 3 & 5
  769  002226  050467  176562                  BIS     R4,XORS
  770  002232  016767  176556  176562          MOV     XORS,ACTCRC
  771  002240  000207                          RTS     PC
  772  002242  005067  176560          PARITY: CLR     PARCNT                  ;CLEAR BIT COUNTER
  773  002246  012703  000010                  MOV     #10,R3                  ;SET NO. OF BITS
  774  002252  032704  000001          CLP0:   BIT     #1,R4                   ;SEE IF ONE BIT
  775  002256  001402                          BEQ     CLP1                    ;IF NOT:  BR
  776  002260  005267  176542                  INC     PARCNT                  ;BUMP COUNTER
  777  002264  000241                  CLP1:   CLC
  778  002266  006004                          ROR     R4                      ;ROTATE TO NEXT BIT
  779  002270  005303                          DEC     R3
  780  002272  001367                          BNE     CLP0                    ;CONTINUE FOR ALL BITS
  781  002274  112104                          MOVB    (R1)+,R4
  782  002276  042704  177400                  BIC     #177400,R4
  783  002302  032767  000001  176516          BIT     #1,PARCNT               ;SEE IF ODD # OF ONE BITS
  784  002310  001002                          BNE     CLP2                    ;IF SO:  BR
  785  002312  052704  000400                  BIS     #400,R4                 ;SET PARITY BIT
  786  002316  000207                  CLP2:   RTS     PC                      ;EXIT
  787
  788  002320  010446                  XOR:    MOV     R4,-(SP)                ;XOR SUBROUTINE: R4 WITH XORS
  789  002322  046716  176466                  BIC     XORS,(SP)
  790  002326  040467  176462                  BIC     R4,XORS
  791  002332  052667  176456                  BIS     (SP)+,XORS
  792  002336  016704  176452                  MOV     XORS,R4
  793  002342  000207                          RTS     PC
  794
  795  002344  016767  176454  176442  LPC:    MOV     ACTLPC,XORS
  796  002352  012104                  LPC1:   MOV     (R1)+,R4
  797  002354  022701  173026                  CMP     #173026,R1              ;LOCATION EFFECTED BY SWITCHES
  798  002360  001402                          BEQ     LPC2                    ;IF SO:  SKIP LOC. BY BRANCHING
  799  002362  004767  177732                  JSR     PC,XOR
  800  002366  005300                  LPC2:   DEC     R0
  801  002370  001370                          BNE     LPC1
  802  002372  016767  176416  176424          MOV     XORS,ACTLPC
  803  002400  000207                          RTS     PC
  804
  805  002402  104401                  TYPROM: TYPE                            ;TYPE HEADER
  806  002404  006005                          TYPHDR
  807  002406  016700  176372                  MOV     ROMSA1,R0               ;POINT OT 1ST ROM SPACE
  808  002412  016701  176370                  MOV     DATLN1,R1               ;PUT LENGTH IN R1
  809  002416  006201                          ASR     R1                      ;CONTERT TO WORDS
  810  002420  001402                          BEQ     TYPR1                   ;BRANCH IF 1ST ROM SPACE NOT USED
  811  002422  004767  000026                  JSR     PC,TYP                  ;GO TYPE 1ST ADDR. SPACE
  812  002426  016700  176356          TYPR1:  MOV     ROMSA2,R0               ;POINT TO 2ND ADDR. SPACE
  813  002432  016701  176354                  MOV     DATLN2,R1               ;PUT LENGTH IN R1
  814  002436  006201                          ASR     R1                      ;CONVERT TO WORDS
  815  002440  001402                          BEQ     ENDOT                   ;BR IF 2ND ADDR. SPACE NOT USED
  816  002442  004767  000006                  JSR     PC,TYP                  ;GO TYPE 2ND ADDR. SPACE
  817  002446  104401                  ENDOT:  TYPE
  818  002450  005460                          EOTST
  819  002452  000000                          HALT
  820
  821  002454  104401                  TYP:    TYPE
  822  002456  005777                          CARLF
```

```
                        000403                          BR      TYP3
002462  032700  000003          TYP0:   BIT     #3,R0           ;ADDRESS MULTIPLE OF 4?
002466  001006                          BNE     TYP2            ;IF NOT:  BR
002470  104401                  TYP3:   TYPE
002472  005777                          CARLF
002474  010046                          MOV     R0,-(SP)                ;PUT ADDRESS ON STACK
002476  104402                          TYPOC                   ;TYPE ADDR.
002500  104401                          TYPE
002502  006042                          COLON
002504  012046                  TYP2:   MOV     (R0)+,-(SP)     ;PUT DATA ON STACK
002506  104402                          TYPOC                   ;TYP DATA
002510  104401                          TYPE                    ;TYPE 2 SPACES
002512  006002                          SP2
002514  005301                          DEC     R1              ;FINISHED?
002516  001361                          BNE     TYP0            ;IF NOT:  BR
002520  000207                          RTS     PC              ;RETURN
002522  005000                  AUTACT: CLR     R0
002524  062700  000002          AUT1:   ADD      #2,R0          ;BUMP TALBE INDEX
002530  026027  005212  177777          CMP     TMSG(R0),#-1    ;CHECKED ALL KNOWN VERSIONS?
002536  001425                          BEQ     AUTERR          ;IF SO:  BR
002540  026067  004644  176254          CMP     TXCRC(R0),ACTCRC ;DOES THIS CRC AGREE?
002546  001366                          BNE     AUT1            ;IF NOT:  KEEP LOOKING
002550  023727  000042  002060          CMP     @#42,#$ENDAD    ;UNDER ACT AUTO ACCEPT?
002556  001405                          BEQ     AUT3            ;IF SO:  BR
002560  016067  005212  000002          MOV     TMSG(R0),AUT2   ;SET UP VERSION MESSAGE
002566  104401                          TYPE
002570  000000                  AUT2:   0
002572  016067  004644  176216  AUT3:   MOV     TXCRC(R0),EXCRC ;SET EXPECTED CRC
002600  016067  004712  176212          MOV     TXLPC(R0),EXLPC ;SET EXPECTED LPC
002606  000167  177124                  JMP     CK1             ;CHECK LPC

002612  104401                  AUTERR: TYPE
002614  006046                          AUTERM
002616  012767  000001  176224          MOV     #1,$FATAL               ;MOVE TO MAILBOX ERROR NO.  ****  1  ****
002624  012767  000001  176214          MOV     #1,$MSGTYP              ;SET MAILBOX FOR FATAL ERRCR
002632  000000                          HALT                            ;AUTO ACCEPT FAILED

                                .SBTTL  TTY INPUT ROUTINE

                                ;;**************************************************************
002634  177560                  $TKS:   .WORD   177560          ;;TTY KBD STATUS
002636  177562                  $TKB:   .WORD   177562          ;;TTY KBD BUFFER
                                .ENABL  LSB

                                .DSABL  LSB


                                ;;**************************************************************
                                ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
                                ;*CALL:
                                ;*      RDCHR                   ;;INPUT A SINGLE CHARACTER FROM THE TTY
                                ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
                                ;*                              ;;WITH PARITY BIT STRIPPED OFF
                                ;

002640  011646                  $RDCHR: MOV     (SP),-(SP       ;;PUSH DOWN THE PC
```

```
 879  002642  016666  000004  000002          MOV     4(SP),2(SP)     ;;SAVE THE PS
 880  002650  105777  177760          1$:     TSTB    @$TKS           ;;WAIT FOR
 881  002654  100375                  BPL     1$              ;;A CHARACTER
 882  002656  117766  177754  000004          MOVB    @$TKB,4(SP)     ;;READ THE TTY
 883  002664  042766  177600  000004          BIC     #^C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
 884  002672  026627  000004  000023          CMP     4(SP),#23       ;;IS IT A CONTROL-S?
 885  002700  001013                  BNE     3$              ;;BRANCH IF NO
 886  002702  105777  177726          2$:     TSTB    @$TKS           ;;WAIT FOR A CHARACTER
 887  002706  100375                  BPL     2$              ;;LOOP UNTIL ITS THERE
 888  002710  117746  177722                  MOVB    @$TKB,-(SP)     ;;GET CHARACTER
 889  002714  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
 890  002720  022627  000021                  CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
 891  002724  001366                  BNE     2$              ;;IF NOT DISCARD IT
 892  002726  000750                  BR      1$              ;;YES, RESUME
 893  002730  026627  000004  000140  3$:     CMP     4(SP),#140      ;;IS IT UPPER CASE?
 894  002736  002407                  BLT     4$              ;;BRANCH IF YES
 895  002740  026627  000004  000175          CMP     4(SP),#175      ;;IS IT A SPECIAL CHAR?
 896  002746  003003                  BGT     4$              ;;BRANCH IF YES
 897  002750  042766  000040  000004          BIC     #40,4(SP)       ;;MAKE IT UPPER CASE
 898  002756  000002                  4$:     RTI                     ;;GO BACK TO USER
 899                                  ;;**************************************************************
 900                                  ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
 901                                  ;*CALL:
 902                                  ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
 903                                  ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
 904                                  ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
 905
 906  002760  010346          $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
 907  002762  005046                  CLR     -(SP)           ;;CLEAR THE RUBOUT KEY
 908  002764  012703  003214          1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
 909  002770  022703  003224          2$:     CMP     #$TTYIN+8.,R3   ;;BUFFER FULL?
 910  002774  101456                  BLOS    4$              ;;BR IF YES
 911  002776  104405                  RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
 912  003000  112613                  MOVB    (SP)+,(R3)      ;;GET CHARACTER
 913  003002  122713  000177          10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
 914  003006  001022                  BNE     5$              ;;BR IF NO
 915  003010  005716                  TST     (SP)            ;;IS THIS THE FIRST RUBOUT?
 916  003012  001007                  BNE     6$              ;;BR IF NO
 917  003014  112767  000134  000170          MOVB    #'\,9$          ;;TYPE A BACK SLASH
 918  003022  104401  003212                  TYPE    ,9$
 919  003026  012716  177777                  MOV     #-1,(SP)        ;;SET THE RUBOUT KEY
 920  003032  005303                  6$:     DEC     R3              ;;BACKUP BY ONE
 921  003034  020327  003214                  CMP     R3,#$TTYIN      ;;STACK EMPTY?
 922  003040  103434                  BLO     4$              ;;BR IF YES
 923  003042  111367  000144                  MOVB    (R3),9$         ;;SETUP TO TYPEOUT THE DELETED CHAR.
 924  003046  104401  003212                  TYPE    ,9$             ;;GO TYPE
 925  003052  000746                  BR      2$              ;;GO READ ANOTHER CHAR.
 926  003054  005716                  5$:     TST     (SP)            ;;RUBOUT KEY SET?
 927  003056  001406                  BEQ     7$              ;;BR IF NO
 928  003060  112767  000134  000124          MOVB    #'\,9$          ;;TYPE A BACK SLASH
 929  003066  104401  003212                  TYPE    ,9$
 930  003072  005016                  CLR     (SP)            ;;CLEAR THE RUBOUT KEY
 931  003074  122713  000025          7$:     CMPB    #25,(R3)        ;;IS CHARACTER A CTRL U?
 932  003100  001003                  BNE     8$              ;;BR IF NO
 933  003102  104401  003230                  TYPE    ,$CNTLU         ;;TYPE A CONTROL "U"
 934  003106  000726                  BR      1$              ;;GO START OVER
```

```
                                        8$:     CMPB    #22,(R3)        ;;IS CHARACTER A "↑R"?
003110  122713  000022                        BNE     3$              ;;BRANCH IF NO
003114  001011                                CLRB    (R3)            ;;CLEAR THE CHARACTER
003116  105013                                TYPE    ,$CRLF          ;;TYPE A "CR" & "LF"
003120  104401  003225                        TYPE    ,$TTYIN         ;;TYPE THE INPUT STRING
003124  104401  003214                        BR      2$              ;;GO PICKUP ANOTHER CHACTER
003130  000717                        4$:     TYPE    ,$QUES          ;;TYPE A '?'
003132  104401  003224                        BR      1$              ;;CLEAR THE BUFFER AND LOOP
003136  000712                        3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
003140  111367  000046                        TYPE    ,9$
003144  104401  003212                        CMPB    #15,(R3)+       ;;CHECK FOR RETURN
003150  122723  000015                        BNE     2$              ;;LOOP IF NOT RETURN
003154  001305                                CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
003156  105063  177777                        TYPE    ,$LF            ;;TYPE A LINE FEED
003162  104401  003226                        TST     (SP)+           ;;CLEAN RUBOUT KEY FROM THE STACK
003166  005726                                MOV     (SP)+,R3        ;;RESTORE R3
003170  012603                                MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
003172  011646                                MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
003174  016666  000004  000002                MOV     #$TTYIN,4(SP)
003202  012766  003214  000004                RTI                     ;;RETURN
003210  000002                        9$:     .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
003212     000                              .BYTE   0               ;;TERMINATOR
003213     000                      $TTYIN: .BLKB   8.              ;;RESERVE 8 BYTES FOR TTY INPUT
003214  000010                      $QUES:  .ASCII  "?"     ;;QUESTION MARK
003224     077                      $CRLF:  .ASCII  <15>    ;CARRIAGE RETURN
003225     015                      $LF:    .ASCII  <12>    ;;LINE FEED
003226  000012                      $CNTLU: .ASCIZ  /↑U/<15><12>    ;;CONTROL "U"
003230  052536  005015     000     $CNTLG: .ASCIZ  /↑G/<15><12>    ;;CONTROL "G"
003235     136  006507  000012     $MSWR:  .ASCIZ  <15><12>/SWR = /
003242  005015  053523  020122     $MNEW:  .ASCIZ  /  NEW = /
003250  020075     000
003253     040  047040  053505
003260  036440  000040

                                        .SBTTL  READ AN OCTAL NUMBER FROM THE TTY

                                        ;;**************************************************************
                                        ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
                                        ;*CHANGE IT TO BINARY.
                                        ;*THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL
                                        ;*OCTAL DIGITS. IF AN ILLEGAL CHARACTER IS READ A "?" WILL BE TYPED
                                        ;*FOLLOWED BY A CARRIAGE RETURN-LINE FEED. THE COMPLETE NUMBER MUST
                                        ;*THEN BE RETYPED. THE INPUT IS TERMINATED BY TYPING A CARRIAGE RETURN.
                                        ;*CALL:
                                        ;*      RDOCT                   ;;READ AN OCTAL NUMBER
                                        ;*      RETURN HERE             ;;LOW ORDER BITS ARE ON TOP OF THE STACK
                                        ;*                              ;;HIGH ORDER BITS ARE IN $HIOCT

003264  011646                  $RDOCT: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR THE
003266  016666  000004  000002          MOV     4(SP),2(SP)     ;;INPUT NUMBER
003274  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
003276  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
003300  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
003302  104406                  1$:     RDLIN                   ;;READ AN ASCIZ LINE
003304  012600                          MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
003306  010067  000100                  MOV     R0,5$           ;;AND SAVE IT
003312  005001                          CLR     R1              ;;CLEAR DATA WORD
003314  005002                          CLR     R2
```

```
 991  003316  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
 992  003320  001420                          BEQ     3$              ;;IF ZERO GET OUT
 993  003322  122716  000060                   CMPB    #'0,(SP)        ;;MAKE SURE THIS CHARACTER
 994  003326  003026                          BGT     4$              ;;IS AN OCTAL DIGIT
 995  003330  122716  000067                   CMPB    #'7,(SP)
 996  003334  002423                          BLT     4$
 997  003336  006301                          ASL     R1              ;;*2
 998  003340  006102                          ROL     R2
 999  003342  006301                          ASL     R1              ;;*4
1000  003344  006102                          ROL     R2
1001  003346  006301                          ASL     R1              ;;*8
1002  003350  006102                          ROL     R2
1003  003352  042716  177770                  BIC     #^C7,(SP)       ;;STRIP THE ASCII JUNK
1004  003356  062601                          ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
1005  003360  000756                          BR      2$              ;;LOOP
1006  003362  005726                  3$:     TST     (SP)+           ;;CLEAN TERMINATOR FROM STACK
1007  003364  010166  000012                  MOV     R1,12(SP)       ;;SAVE THE RESULT
1008  003370  010267  000026                  MOV     R2,$HIOCT
1009  003374  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
1010  003376  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
1011  003400  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
1012  003402  000002                          RTI                     ;;RETURN
1013  003404  005726                  4$:     TST     (SP)+           ;;CLEAN PARTIAL FROM STACK
1014  003406  105010                          CLRB    (R0)            ;;SET A TERMINATOR
1015  003410  104401                          TYPE                    ;;TYPE UP THRU THE BAD CHAR.
1016  003412  000000                  5$:     .WORD   0
1017  003414  104401  003224                  TYPE    ,$QUES          ;;"?" "CR" & "LF"
1018  003420  000730                          BR      1$              ;;TRY AGAIN
1019  003422  000000          $HIOCT: .WORD   0               ;;HIGH ORDER BITS GO HERE
1020                          .SBTTL  TYPE ROUTINE
1021
1022                          ;;**************************************************************
1023                          ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
1024                          ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
1025                          ;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
1026                          ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
1027                          ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
1028                          ;*
1029                          ;*CALL:
1030                          ;*1) USING A TRAP INSTRUCTION
1031                          ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
1032                          ;*OR
1033                          ;*      TYPE
1034                          ;*      MESADR
1035                          ;*
1036
1037  003424  105767  000265  175414  $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
1038  003430  100002                          BPL     1$              ;;BR IF YES
1039  003432  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
1040  003434  000430                          BR      3$              ;;LEAVE
1041  003436  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
1042  003440  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
1043  003444  122767  000001  175414          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
1044  003452  001011                          BNE     6$              ;;NO,GO CHECK FOR APT CONSOLE
1045  003454  132767  000100  175405          BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
1046  003462  001405                          BEQ     6$              ;;NO,GO CHECK FOR CONSOLE
```

```
1047  003464  010067  000004                 MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
1048  003470  004767  000230                 JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
1049  003474  000000                 61$:    .WORD   0               ;;MESSAGE ADDRESS
1050  003476  132767  000040  175363 62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
1051  003504  001003                         BNE     60$             ;;YES,SKIP TYPE OUT
1052  003506  112046                 2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
1053  003510  001005                         BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
1054  003512  005726                         TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
1055  003514  012600                 60$:    MOV     (SP)+,R0        ;;RESTORE R0
1056  003516  062716  000002         3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
1057  003522  000002                         RTI                     ;;RETURN
1058  003524  122716  000011         4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
1059  003530  001430                         BEQ     8$
1060  003532  122716  000200                 CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
1061  003536  001006                         BNE     5$
1062  003540  005726                         TST     (SP)+           ;;POP  <CR><LF> EQUIV
1063  003542  104401                         TYPE                    ;;TYPE A CR AND LF
1064  003544  003225                         $CRLF
1065  003546  105067  000130                 CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
1066  003552  000755                         BR      2$              ;;GET NEXT CHARACTER
1067  003554  004767  000056         5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
1068  003560  126726  000130         6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
1069  003564  001350                         BNE     2$              ;;IF NO GO GET NEXT CHAR.
1070  003566  016746  000120                 MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
1071                                                                 ;;AND THE NULL CHAR.
1072  003572  105366  000001         7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
1073  003576  002770                         BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
1074  003600  004767  000032                 JSR     PC,$TYPEC       ;;GO TYPE A NULL
1075  003604  105367  000072                 DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
1076  003610  000770                         BR      7$              ;;LOOP
1077
1078                                 ;HORIZONTAL TAB PROCESSOR
1079
1080  003612  112716  000040         8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
1081  003616  004767  000014         9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
1082  003622  132767  000007  000052         BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
1083  003630  001372                         BNE     9$              ;;TAB STOP
1084  003632  005726                         TST     (SP)+           ;;POP SPACE OFF STACK
1085  003634  000724                         BR      2$              ;;GET NEXT CHARACTER
1086  003636  105777  000044         $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
1087  003642  100375                         BPL     $TYPEC
1088  003644  116677  000002  000036         MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
1089  003652  122766  000015  000002         CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
1090  003660  001003                         BNE     1$              ;;BRANCH IF NO
1091  003662  105067  000014                 CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
1092  003666  000406                         BR      $TYPEX          ;;EXIT
1093  003670  122766  000012  000002 1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
1094  003676  001402                         BEQ     $TYPEX          ;;BRANCH IF YES
1095  003700  105227                         INCB    (PC)+           ;;COUNT THE CHARACTER
1096  003702  000000         $CHARCNT:.WORD  0                       ;;CHARACTER COUNT STORAGE
1097  003704  000207         $TYPEX: RTS     PC
1098
1099  003706  177564         $TPS:   .WORD   177564          ;;TTY PRINTER STATUS REG. ADDRESS
1100  003710  177566         $TPB:   .WORD   177566          ;;TTY PRINTER BUFFER REG. ADDRESS
1101  003712     000         $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER FOR FILLS
1102  003713     002         $FILLS: .BYTE   2               ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
```

```
1103  003714  012                   $FILLC: .BYTE   12              ;;INSERT FILL CHARS. AFTER A "LINE FEED"
1104  003715  000                   $TPFLG: .BYTE   0               ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
1105                                .SBTTL  APT COMMUNICATIONS ROUTINE
1106
1107                                ;;**********************************************************************
1108  003716  112767  000001  000236  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
1109  003724  112767  000001  000226  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
1110  003732  000403                        BR      $ATYC
1111  003734  112767  000001  000220  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
1112  003742                        $ATYC:
1113  003742  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
1114  003744  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
1115  003746  105767  000206                TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
1116  003752  001450                        BEQ     5$              ;;IF NOT:  BR
1117  003754  122767  000001  175104        CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
1118  003762  001031                        BNE     3$              ;;IF NOT:  BR
1119  003764  132767  000100  175075        BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
1120  003772  001425                        BEQ     3$              ;;IF NOT:  BR
1121  003774  017600  000004                MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
1122  004000  062766  000002  000004        ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
1123  004006  005767  175034          1$:   TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
1124  004012  001375                        BNE     1$              ;;IF NOT:  WAIT
1125  004014  010067  175042                MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
1126  004020  105720                  2$:   TSTB    (R0)+           ;;FIND END OF MESSAGE
1127  004022  001376                        BNE     2$
1128  004024  166700  175032                SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
1129  004030  006200                        ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
1130  004032  010067  175026                MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
1131  004036  012767  000004  175002        MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
1132  004044  000413                        BR      5$
1 33  004046  017667  000004  000016  3$:   MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
1134  004054  062766  000002  000004        ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
1135  004062  016746  173710                MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
1136  004066  004767  177332                JSR     PC,$TYPE        ;;CALL TYPE MACRO
1137  004072  000000                  4$:   .WORD   0
1138  004074                           5$:
1139  004074  105767  000062          10$:  TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
1140  004100  001416                        BEQ     12$             ;;IF NOT:  BR
1141  004102  005767  174760                TST     $ENV            ;;RUNNING UNDER APT?
1142  004106  001413                        BEQ     12$             ;;IF NOT:  BR
1143  004110  005767  174732          11$:  TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
1144  004114  001375                        BNE     11$             ;;IF NOT:  WAIT
1145  004116  017667  000004  174724        MOV     @4(SP),$FATAL   ;;GET ERROR #
1146  004124  062766  000002  000004        ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
1147  004132  005267  174710                INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
1148  004136  105067  000020          12$:  CLRB    $FFLG           ;;CLEAR FATAL FLAG
1149  004142  105067  000013                CLRB    $LFLG           ;;CLEAR LOG FLAG
1150  004146  105067  000006                CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
1151  004152  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
1152  004154  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
1153  004156  000207                        RTS     PC              ;;RETURN
1154  004160     000          $MFLG:  .BYTE   0               ;;MESSG. FLAG
1155  004161     000          $LFLG:  .BYTE   0               ;;LOG FLAG
1156  004162     000          $FFLG:  .BYTE   0               ;;FATAL FLAG
1157        004164                        .EVEN
1158        000200                  APTSIZE=200
```

```
                        000001          APTENV=001
                        000100          APTSPOOL=100
                        000040          APTCSUP=040
                                        .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE

                                        ;;*****************************************************************
                                        ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
                                        ;*OCTAL (ASCII) NUMBER AND TYPE IT.
                                        ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
                                        ;*CALL:
                                        ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                                        ;*      TYPOS                   ;;CALL FOR TYPEOUT
                                        ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
                                        ;*      .BYTE   M               ;;M=1 OR 0
                                        ;*                                      ;;1=TYPE LEADING ZEROS
                                        ;*                                      ;;0=SUPPRESS LEADING ZEROS
                                        ;*
                                        ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
                                        ;*$TYPOS OR $TYPOC
                                        ;*CALL:
                                        ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                                        ;*      TYPON                   ;;CALL FOR TYPEOUT
                                        ;*
                                        ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
                                        ;*CALL:
                                        ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                                        ;*      TYPOC                   ;;CALL FOR TYPEOUT
                                        ;*
004164  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
004170  116667  000001  000211          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
004176  112667  000207                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
004202  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
004206  000406                          BR      $TYPON
004210  112767  000001  000171  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
004216  112767  000006  000165          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
004224  112767  000005  000154  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
004232  010346                          MOV     R3,-(SP)        ;;SAVE R3
004234  010446                          MOV     R4,-(SP)        ;;SAVE R4
004236  010546                          MOV     R5,-(SP)        ;;SAVE R5
004240  116704  000145                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
004244  005404                          NEG     R4
004246  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
004252  110467  000132                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
004256  116704  000125                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
004262  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
004266  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
004270  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
004272  000404                          BR      3$              ;;GO DO MSB
004274  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
004276  006105                          ROL     R5
004300  006105                          ROL     R5
004302  010503                          MOV     R5,R3
004304  006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
004306  105367  000076                  DECB    $OMODE          ;;TYPE THIS DIGIT?
004312  100016                          BPL     7$              ;;BR IF NO
004314  042703  177770                  BIC     #177770,R3      ;;GET RID OF JUNK
```

```
1215  004320  001002                        BNE     4$              ;;TEST FOR 0
1216  004322  005704                        TST     R4              ;;SUPPRESS THIS 0?
1217  004324  001403                        BEQ     5$              ;;BR IF YES
1218  004326  005204                4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
1219  004330  052703  000060                BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
1220  004334  052703  000040        5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
1221  004340  110367  000040                MOVB    R3,8$           ;;SAVE FOR TYPING
1222  004344  104401  004404                TYPE    ,8$             ;;GO TYPE THIS DIGIT
1223  004350  105367  000032        7$:     DECB    $OCNT           ;;COUNT BY 1
1224  004354  003347                        BGT     2$              ;;BR IF MORE TO DO
1225  004356  002402                        BLT     6$              ;;BR IF DONE
1226  004360  005204                        INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
1227  004362  000744                        BR      2$              ;;GO DO THE LAST DIGIT
1228  004364  012605                6$:     MOV     (SP)+,R5        ;;RESTORE R5
1229  004366  012604                        MOV     (SP)+,R4        ;;RESTORE R4
1230  004370  012603                        MOV     (SP)+,R3        ;;RESTORE R3
1231  004372  016666  000002  000004        MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
1232  004400  012616                        MOV     (SP)+,(SP)
1233  004402  000002                        RTI                     ;;RETURN
1234  004404     000                8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
1235  004405     000                        .BYTE   0               ;;TERMINATOR FOR T PE ROUTINE
1236  004406     000                $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
1237  004407     000                $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
1238  004410  000000                $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
1239                                .SBTTL  POWER DOWN AND UP ROUTINES
1240
1241                                ;;**************************************************************
1242                                ;POWER DOWN ROUTINE
1243  004412  012737  004552  000024 $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
1244  004420  012737  000340  000026         MOV     #340,@#PWRVEC+2 ;;PRIO:7
1245  004426  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
1246  004430  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
1247  004432  010246                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
1248  004434  010346                        MOV     R3,-(SP)        ;;PUSH R3 ON STACK
1249  004436  010446                        MOV     R4,-(SP)        ;;PUSH R4 ON STACK
1250  004440  010546                        MOV     R5,-(SP)        ;;PUSH R5 ON STACK
1251  004442  017746  174332                MOV     @SWR,-(SP)      ;;PUSH @SWR ON STACK
1252  004446  010667  000104                MOV     SP,$SAVR6       ;;SAVE SP
1253  004452  012737  004464  000024        MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
1254  004460  000000                        HALT
1255  004462  000776                        BR      .-2             ;;HANG UP
1256
1257                                ;;**************************************************************
1258                                ;POWER UP ROUTINE
1259  004464  012737  004552  000024 $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
1260  004472  016706  000060                MOV     $SAVR6,SP       ;;GET SP
1261  004476  005067  000054                CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
1262  004502  005267  000050        1$:     INC     $SAVR6          ;;WAIT FOR THE INC
1263  004506  001375                        BNE     1$              ;;OF  WORD
1264  004510  012677  174264                MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
1255  004514  012605                        MOV     (SP)+,R5        ;;POP STACK INTO R5
1266  004516  012604                        MOV     (SP)+,R4        ;;POP STACK INTO R4
1267  004520  012603                        MOV     (SP)+,R3        ;;POP STACK INTO R3
1268  004522  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
1269  004524  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
1270  004526  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
```

```
1271  004530  012737  004412  000024          MOV     #$PWRUN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
1272  004536  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
1273  004544  104401                          TYPE                    ;REPORT THE POWER FAILURE
1274  004546  004560                $PWRMG:   .WORD   $POWER          ;;POWER FAIL MESSAGE POINTER
1275  004550  000002                          RTI
1276  004552  000000                $ILLUP:   HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
1277  004554  000776                          BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
1278  004556  000000                $SAVR6:   0                       ;;PUT THE SP HERE
1279  004560  005015  047520  042527  $POWER:   .ASCIZ  <15><12>"POWER"
1280  004566  000122
1281                                          .EVEN
1282                                .SBTTL    TRAP DECODER
1283
1284                                ;;**************************************************************
1285                                ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
1286                                ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
1287                                ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
1288                                ;*GO TO THAT ROUTINE.
1289
1290  004570  010046                $TRAP:    MOV     R0,-(SP)        ;;SAVE R0
1291  004572  016600  000002                   MOV     2(SP),R0        ;;GET TRAP ADDRESS
1292  004576  005740                           TST     -(R0)           ;;BACKUP BY 2
1293  004600  111000                           MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
1294  004602  006300                           ASL     R0              ;;POSITION FOR INDEXING
1295  004604  016000  004624                   MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
1296  004610  000200                           RTS     R0              ;;GO TO ROUTINE
1297
1298
1299                                ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
1300
1301  004612  011646                $TRAP2:   MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
1302  004614  016666  000004  000002          MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
1303  004622  000002                          RTI                     ;;RESTORE THE PSW
1304
1305                                .SBTTL    TRAP TABLE
1306
1307                                ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
1308                                ;*BY THE "TRAP" INSTRUCTION.
1309
1310                                ;         ROUTINE
1311                                ;         -------
1312  004624  004612                $TRPAD:   .WORD   $TRAP2
1313  004626  003424                          $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
1314  004630  004210                          $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
1315  004632  004164                          $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
1316  004634  004224                          $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
1317
1318
1319  004636  002640                          $RDCHR  ;;CALL=RDCHR    TRAP+5(104405)  TTY TYPEIN CHARACTER ROUTINE
1320  004640  002760                          $RDLIN  ;;CALL=RDLIN    TRAP+6(104406)  TTY TYPEIN STRING ROUTINE
1321  004642  003264                          $RDOCT  ;;CALL=RDOCT    TRAP+7(104407)  READ AN OCTAL NUMBER FROM TTY
1322
1323  004644  177777                TXCRC:    -1
1324  004646  000571                          571             ;M9301 - YA VERSION
1325  004650  000457                          457             ;M9301 - YB VERSION
1326  004652  000243                          243             ;M9301 - YC VERSION
```

```
1327  004654  000635                    635             ;M9400 - YA(OR YC) VERSION
1328  004656  000207                    207             ;M9301 - YF VERSION
1329  004660  000670                    670             ;M7942 - YB VERSION
1330  004662  000132                    132             ;M9301- YD VERSION
1331  004664  000374                    374             ;M9400 - YH (OR YK) VERSION
1332  004666  000533                    533             ;M9311 VERSION
1333  004670  000536                    536             ;M9301 - YH VERSION
1334  004672  000752                    752             ;M9301 - YE VERSION
1335  004674  000633                    633             ;M9301 - YJ VERSION
1336  004676  177777                    -1
1337  004700  177777                    -1
1338  004702  177777                    -1
1339  004704  177777                    -1
1340  004706  177777                    -1
1341  004710  177777                    -1
1342
1343  004712  17777                 TXLPC:  -1
1344  004714  133725                    133725          ;M9301 - YA VERSION
1345  004716  017563                    17563           ;M9301 - YB VERSION
1346  004720  141744                    141744          ;M9301 - YC VERSION
1347  004722  047613                    47613           ;M9400 - YA(OR YC) VERSION
1348  004724  114175                    114175          ;M9301 - YF VERSION
1349  004726  146126                    146126          ;M7942 - YB VERSION
1350  004730  132161                    132161          ;M9301 - YD VERSION
1351  004732  143466                    143466          ;M9400 - YH(OR YK) VERSION
1352  004734  036104                    036104          ;M9311 VERSION
1353  004736  125411                    125411          ;M9301 - YH VERSION
1354  004740  066246                    066246          ;M9301 - YE VERSION
1355  004742  132367                    132367          ;M9301 - YJ VERSION
1356  004744  177777                    -1
1357  004746  177777                    -1
1358  004750  177777                    -1
1359  004752  177777                    -1
1360  004754  177777                    -1
1361  004756  177777                    -1
1362  004760  177777                    -1
1363
1364  004762  177777                TDLN1:  -1
1365  004764  001000                    1000            ;M9301 - YA VERSION
1366  004766  001000                    1000            ;M9301 - YB VERISION
1367  004770  001000                    1000            ;M9301 - YC VERSION
1368  004772  001000                    1000            ;M9400 - YA(OR YC) VERSION
1369  004774  001000                    1000            ;M9301 - YF VERSION
1370  004776  004000                    4000            ;M7942 - YB VERSION
1371  005000  001000                    1000            ;M9301 - YD VERSION
1372  005002  001000                    1000            ;M9400 - YH(OR YK) VERSION
1373  005004  001000                    1000            ;M9311 VERSION
1374  005006  000734                    734             ;M9301 - YH VERSION
1375  005010  001000                    1000            ;M9301 - YE VERSION
1376  005012  001000                    1000            ;M9301 - YJ VERSION
1377  005014  177777                    -1
1378  005016  177777                    -1
1379  005020  177777                    -1
1380  005022  177777                    -1
1381  005024  177777                    -1
1382  005026  177777                    -1
```

```

005030 177777            TRMSA1: -1
005032 173000                    173000          ;M9301 - YA VERSION
005034 173000                    173000          ;M9301 - YB VERSION
005036 173000                    173000          ;M9301 - YC VERSION
005040 173000                    173000          ;M9400 - YA(OR YC) VERSION
005042 173000                    173000          ;M9301 - YF VERSION
005044 170000                    170000          ;M7942 - YB VERSION
005046 173000                    173000          ;M9301 - YD VERSION
005050 173000                    173000          ;M9400 - YH(OR YK) VERSION
005052 163000                    163000          ;M9311 VERSION
005054 173000                    173000          ;M9301 - YH VERSION
005056 173000                    173000          ;M9301 - YE VERSION
005060 173000                    173000          ;M9301 - YJ VERSION
005062 177777                    -1
005064 177777                    -1
005066 177777                    -1
005070 177777                    -1
005072 177777                    -1
005074 177777                    -1

005076 177777            TDLN2:  -1
005100 001000                    1000            ;M9301 - YA VERSION
005102 001000                    1000            ;M9301 - YB VERSION
005104 001000                    1000            ;M9301 - YC VERSION
005106 001000                    1000            ;M9400 -YA(OR YC) VERSION
005110 001000                    1000            ;M9301 -YF VERSION
005112 000000                    0               ;M7942 - YB VERSION
005114 001000                    1000            ;M9301 - YD VERSION
005116 001000                    1000            ;M9400 - YH(OR YK) VERSION
005120 001000                    1000            ;M9311 VERSION
005122 000764                    764             ;M9301 - YH VERSION
005124 001000                    1000            ;M9301 - YE VERSION
005126 001000                    1000            ;M9301 - YJ VERSION
005130 177777                    -1
005132 177777                    -1
005134 177777                    -1
005136 177777                    -1
005140 177777                    -1
005142 177777                    -1

005144 177777            TRMSA2: -1
005146 165000                    165000          ;M9301 - YA VERSION
005150 165000                    165000          ;M9301 - YB VERSION
005152 165000                    165000          ;M9301 - YC VERSION
005154 165000                    165000          ;M9400 - YA(OR YC) VERSION
005156 165000                    165000          ;M9301 - YF VERSION
005160 000000                    0               ;M7942 - YB VERSION

005162 165000                    165000          ;M9301 - YD VERSION
005164 165000                    165000          ;M9400 - YH(OR YK) VERSION
005166 166000                    166000          ;M9311 VERSION
005170 165000                    165000          ;M9301 - YH VERSION
005172 165000                    165000          ;M9301 - YE VERSION
005174 165000                    165000          ;M9301 - YJ VERSION
005176 177777                    -1
```

```
 1439  005200  177777                          -1
 1440  005202  177777                          -1
 1441  005204  177777                          -1
 1442  005206  177777                          -1
 1443  005210  177777                          -1
 1444
 1445  005212  177777                  TMSG:   -1
 1446  005214  006070                          MSG1
 1447  005216  006105                          MSG2
 1448  005220  006122                          MSG3
 1449  005222  006137                          MSG4
 1450  005224  006157                          MSG5
 1451  005226  006174                          MSG6            ;M7942 - YB VERSION
 1452  005230  006211                          MSG7
 1453  005232  006226                          MSG10
 1454  005234  006252                          MSG11
 1455  005236  006262                          MSG12           ;M9301 - YH VERSION
 1456  005240  006277                          MSG13           ;M9301 - YE VERSION
 1457  005242  006314                          MSG14           ;M9301 - YJ VERSION
 1458  005244  177777                          -1
 1459  005246  177777                          -1
 1460  005250  177777                          -1
 1461  005252  177777                          -1
 1462  005254  177777                          -1
 1463  005256  177777                          -1
 1464
 1465
 1466  005260  005015  030122  020115  TITL:   .ASCIZ  <15><12>/ROM TEST/
 1467  005266  042524  052123     000
 1468  005273     015  052012  050131  GETCRC: .ASCIZ  <15><12>/TYPE CRC VALUE:  /
 1469  005300  020105  051103  020103
 1470  005306  040526  052514  035105
 1471  005314  020040     000
 1472  005317     015  052012  050131  GETLPC: .ASCIZ  <15><12>/TYPE LPC VALUE:  /
 1473  005324  020105  050114  020103
 1474  005332  040526  052514  035105
 1475  005340  020040     000
 1476  005343     015  042412  050130  EXCRMG: .ASCIZ  <15><12>/EXPECTED CRC =  /
 1477  005350  041505  042524  020104
 1478  005356  051103  020103  020075
 1479  005364  000040
 1480  005366  005015  042412  050130  EXLPMG: .ASCIZ  <15><12><12>/EXPECTED LPC =  /
 1481  005374  041505  042524  020104
 1482  005402  050114  020103  020075
 1483  005410  000040
 1484  005412  005015  047503  050115  ACCRMG: .ASCIZ  <15><12>/COMPUTED CRC =  /
 1485  005420  052125  042105  041440
 1486  005426  041522  036440  020040
 1487  005434     000
 1488  005435     015  041412  046517  ACLPMG: .ASCIZ  <15><12>/COMPUTED LPC =  /
 1489  005442  052520  042524  020104
 1490  005450  050114  020103  020075
 1491  005456  000040
 1492  005460  005015  042412  042116  ECTST:  .ASCIZ  <15><12><12>/END OF TEST/
 1493  005466  047440  020106  042524
 1494  005474  052123     000
```

```
 1495  005477     015  052012  050131  SA1:    .ASCIZ  <15><12>/TYPE STARTING ADDR. OF 1ST ROM ADDR. SPACE:  /
 1496  005504  020105  052123  051101
 1497  005512  044524  043516  040440
 1498  005520  042104  027122  047440
 1499  005526  020106  051461  020124
 1500  005534  047522  020115  042101
 1501  005542  051104  020056  050123
 1502  005550  041501  035105  020040
 1503  005556     000
 1504  005557     015  052012  050131  SA2:    .ASCIZ  <15><12>/TYPE STARTING ADDR. OF 2ND ROM ADDR. SPACE:  /
 1505  005564  020105  052123  051101
 1506  005572  044524  043516  040440
 1507  005600  042104  027122  047440
 1508  005606  020106  047062  020104
 1509  005614  047522  020115  042101
 1510  005622  051104  020056  050123
 1511  005630  041501  035105  020040
 1512  005636     000
 1513  005637     015  052012  050131  SIZE1:  .ASCIZ  <15><12>/TYPE LENGTH (BYTES) OF 1ST ROM ADDR. SPACE:  /
 1514  005644  020105  042514  043516
 1515  005652  044124  024040  054502
 1516  005660  042524  024523  047440
 1517  005666  020106  051461  020124
 1518  005674  047522  020115  042101
 1519  005702  051104  020056  050123
 1520  005710  041501  035105  020040
 1521  005716     000
 1522  005717     015  052012  050131  SIZE2:  .ASCIZ  <15><12>/TYPE LENGTH (BYTES) OF 2ND ROM ADDR. SPACE:  /
 1523  005724  020105  042514  043516
 1524  005732  044124  024040  054502
 1525  005740  042524  024523  047440
 1526  005746  020106  047062  020104
 1527  005754  047522  020115  042101
 1528  005762  051104  020056  050123
 1529  005770  041501  035105  020040
 1530  005776     000
 1531  005777     015  000012          CARLF:  .ASCIZ  <15><12>
 1532  006002  020040     000          SP2:    .ASCIZ  /  /
 1533  006005     015  040412  042104  TYPHDR: .ASCIZ  <15><12>/ADDRESS                 DATA
 1534  006012  042522  051523  020040
 1535  006020  020040  020040  020040
 1536  006026  020040  020040  020040
 1537  006034  042040  052101  000101
 1538  006042  020072  000040          COLON:  .ASCIZ  /:  /
 1539  006046  005015  047125  047113  AUTERM: .ASCIZ  <15><12>/UNKNOWN MODULE /
 1540  006054  053517  020116  047515
 1541  006062  052504  042514  000040
 1542
 1543  006070  005015  034515  030063  MSG1:   .ASCIZ  <15><12>'M9301 - YA/
 1544  006076  020061  020055  040531
 1545  006104     000
 1546  006105     015  046412  031471  MSG2:   .ASCIZ  <15><12>/M9301 - YB/
 1547  006112  030460  026440  054440
 1548  006120  020102
 1549  006122  005015  034515  030063  MSG3:   .ASCIZ  <15><12>/M9301 - YC/
 1550  006130  020061  020055  041531
```

```
1551  006136     000
1552  006137     015  046413  032071  MSG4:   .ASCIZ  <15><12>/M9400 - YA,YC/
1553  006144  030060  026440  054440
1554  006152  026101  041531     000
1555  006157     015  046412  031471  MSG5:   .ASCIZ  <15><12>/M9301 - YF/
1556  006164  030460  026440  054440
1557  006172  000106
1558  006174  005015  033515  032071  MSG6:   .ASCIZ  <15><12>/M7942 - YB/
1559  006202  020062  020055  041131
1560  006210     000
1561  006211     015  046412  031471  MSG7:   .ASCIZ  <15><12>/M9301 - YD/
1562  006216  030460  026440  054440
1563  006224  000104
1564  006226  005015  034515  030064  MSG10:  .ASCIZ  <15><12>/M9400 - YH(OR YK)/
1565  006234  020060  020055  044131
1566  006242  047450  020122  045531
1567  006250  000051
1568  006252  005015  034515  030463  MSG11:  .ASCIZ  <15><12>/M9311/
1569  006260  000061
1570  006262  005015  034515  030063  MSG12:  .ASCIZ  <15><12>/M9301 - YH/
1571  006270  020061  020055  044131
1572  006276     000
1573  006277     015  046412  031471  MSG13:  .ASCIZ  <15><12>/M9301 - YE/
1574  006304  030460  026440  054440
1575  006312  000105
1576  006314  005015  034515  030063  MSG14:  .ASCIZ  <15><12>/M9301 - YJ/
1577  006322  020061  020055  045131
1578  006330     000
1579          000001                  .END
```

```
ABASE = 000000            559
ACCRMG  005412            697    1484#
ACDW1 = 000000            559
ACDW2 = 000000            559
ACLPMG  005435            715    1488#
ACPUOP= 000000            559     574
ACTCRC  001022            519#    666*    689     698     740     770*    843
ACTLPC  001024            520#    667*    708     716     795     802*
ADDW0 = 000000            559
ADDW1 = 000000            559
ADDW10= 000000            559
ADDW11= 000000            559
ADDW12= 000000            559
ADDW13= 000000            559
ADDW14= 000000            559
ADDW15= 000000            559
ADDW2 = 000000            559
ADDW3 = 000000            559
ADDW4 = 000000            559
ADDW5 = 000000            559
ADDW6 = 000000            559
ADDW7 = 000000            559
ADDW8 = 000000            559
ADDW9 = 000000            559
ADEVCT= 000000            559     565
ADEVM = 000000            559
AENV  = 000000            559     570
AENVM = 000000            559     571
AFATAL= 000000            559     562
AMADR1= 000000            559
AMADR2= 000000            559
AMADR3= 000000            559
AMADR4= 000000            559
AMAMS1= 000000            559
AMAMS2= 000000            559
AMAMS3= 000000            559
AMAMS4= 000000            559
AMSGAD= 000000            559     567
AMSGLG= 000000            559     568
AMSGTY= 000000            559     561
AMTYP1= 000000            559
AMTYP2= 000000            559
AMTYP3= 000000            559
AMTYP4= 000000            559
APASS = 000000            559     564
APRIOR= 000000            559
APTCSU= 000040           1050    1161#
APTENV= 000001           1043    1117    1159#
APTSIZ= 000200            611    1158#
APTSPO= 000100           1045    1119    1160#
ASWREG= 000000            559     572
ATESTN= 000000            559     563
AUNIT = 000000            559     566
AUSWR = 000000            559     573
AXTACT  002522            688     839#
AXTERM  006046            855    1539#
```

```
AUTERR  002612            842     854#
AUT1    002524            840#    844
AUT2    002570            847*    849#
AUT3    002572            846     850#
AVECT1= 000000            559
AVECT2= 000000            559
BIT0  = 000001            475#
BIT00 = 000001            465#    475
BIT01 = 000002            464#    474
BIT02 = 000004            463#    473
BIT03 = 000010            462#    472
BIT04 = 000020            461#    471
BIT05 = 000040            460#    470
BIT06 = 000100            459#    469
BIT07 = 000200            458#    468
BIT08 = 000400            457#    467
BIT09 = 001000            456#    466
BIT1  = 000002            474*
BIT10 = 002000            455#
BIT11 = 004000            454#
BIT12 = 010000            453#
BIT13 = 020000            452#
BIT14 = 040000            451#
BIT15 = 100000            450#
BIT2  = 000004            473#
BIT3  = 000010            472#
BIT4  = 000020            471#
BIT5  = 000040            470#
BIT6  = 000100            469#
BIT7  = 000200            468#
BIT8  = 000400            467#
BIT9  = 001000            466#
BPTVEC= 000014            482#
CARLF   005777            822     827     1531#
CHECK   001510            632     634      636     663#
CHO     001566            669     676#
CH1     001644            678     687      689#
CK1     001736            690     703      708#    852
CK2     002030            709     721      726#
CLLAST  002204            763     765#
CLPO    002252            774#    780
CLP1    002264            775     777#
CLP2    002316            784     786#
CLO     002102            741#    747      764
CL1     002146            755#
CL2     002170            752     761#
CL3     002122            743     748#
COLON   006042            831    1538#
CR    = 000015            390#   1089     1099
CRC     002074            671     679      740#
CRLF  = 000200            391#   1060     1099
DATLN1  001006            513#    628*     654*    668     673     808
DATLN2  001012            515#    630*     662*    677     681     813
DDISP = 177570            397#    599
DISPLA  001002            511#    599*     607*
DISPRE  000174            498#    607
```

```
DSWR   = 177570            396#    598
EMTVEC= 000030             485#
ENDOT   002446             815     817#
EOTST   005460             729     818    1492#
ERRVEC= 000004             478#    596     597*    608*
EXCRC   001016             517#    627*    642*    689     694     850*
EXCRMG  005343             693    1476#
EXLPC   001020             518#    626*    646*    708     712     851*
EXLPMG  005366             711    1480#
GETCRC  005273             640    1468#
GETIN   001370             624     633#
GETLPC  005317             644    1472#
GET2    001440             638     647#
GNS    = ****** U          497    1313    1314    1315    1316    1319    1320    1321
GOAGIN  002070             732     738#
HT     = 000011            388#   1058    1099
IOTVEC= 000020             483#
LF     = 000012            389#   1093    1099
LPC     002344             675     683     795#
LPC1    002352             796#    801
LPC2    002366             798     800#
MSG1    006070            1446    1543#
MSG10   006226            1453    1564#
MSG11   006252            1454    1568#
MSG12   006262            1455    1570#
MSG13   006277            1456    1573#
MSG14   006314            1457    1576#
MSG2    006105            1447    1546#
MSG3    006122            1448    1549#
MSG4    006137            1449    1552#
MSG5    006157            1450    1555#
MSG6    006174            1451    1558#
MSG7    006211            1452    1561#
PARCNT  001026             521#    772*    776*    783
PARITY  002242             748     772#
PIRQ   = 177772            395#
PIRQVE= 000240             489#
PR0    = 000000            412#
PR1    = 000040            413#
PR2    = 000100            414#
PR3    = 000140            415#
PR4    = 000200            416#
PR5    = 000240            417#
PR6    = 000300            418#
PR7    = 000340            419#
PS     = 177776            392#    393     506#
PSW    = 177776            393#
PWRVEC= 000024             484#    592*    593*   1243*   1244*   1253*   1259*   1271*   1272*
RDCHR  = 104405            911    1319#
RDLIN  = 104406            986    1320#
RDOCT  = 104407            641     645     649     653     657     661    1321#
RESTRT  001274             616     620#    738
RESVEC= 000010             479#
ROMSA1  001004             512#    629*    650*    670     672     907
ROMSA2  001010             514#    631*    658*    676     680     812
SA1     005477             648    1495#
```

```
SA2     005557            656    1504#
SIZE1   005637            652    1513#
SIZE2   005717            660    1522#
SP2     006002            835    1532#
STACK = 001100            383#
START   001076            502     504     584#
STKLMT= 177774            394#
ST2     001320            625#
SWR     001000            510#    598*    600     606*    613*    622*    623    1251    1264#
SWREG   000176            499#    606     622
SW0   = 000001            447#
SW00  = 000001            437#    447
SW01  = 000002            436#    446
SW02  = 000004            435#    445
SW03  = 000010            434#    444
SW04  = 000020            433#    443
SW05  = 000040            432#    442
SW06  = 000100            431#    441
SW07  = 000200            430#    440
SW08  = 000400            429#    439
SW09  = 001000            428#    438
SW1   = 000002            446#
SW10  = 002000            427#
SW11  = 004000            426#
SW12  = 010000            425#
SW13  = 020000            424#
SW14  = 040000            423#
SW15  = 100000            422#
SW2   = 000004            445#
SW3   = 000010            444#
SW4   = 000020            443#
SW5   = 000040            442#
SW6   = 000100            441#
SW7   = 000200            440#
SW8   = 000400            439#
SW9   = 001000            438#
TBITVE= 000014            480#
TDLN1   004762            628    1364#
TDLN2   005076            630    1404#
TITL    005260            618    1466#
TKVEC = 000060            487#
TMSG    005212            841     847    1445#
TPVEC = 000064            488#
TRAPVE= 000034            486#    507#    590#    591#
TRMSA1  005030            629    1384#
TRMSA2  005144            631    1424#
TRTVEC= 000014            481#
TXCRC   004644            627     843     850    1323#
TXLPC   004712            626     851    1343#
TYP     002454            811     816     821#
TYPE  = 104401            617     639     643     647     651     655     659     692     696     710     714     728     805
                          817     821     826     830     834     848     854     918     924     929     933     938     939
                          941     944     948    1015    1017    1063    1222    1273    1313#
TYPHDR  006005            806    1533#
TYPOC = 104402            695     699     713     717     829     833    1314#
TYPON = 104404           1316#
```

```
TYPOS = 104403          1315#
TYPOUT  001030           501*   503*   522#   637    663
TYPROM  002402           665    805#
TYPR1   002426           810    812#
TYPO    002462           824#   837
TYP2    002504           825    832#
TYP3    002470           823    826#
XOR     002320           749    788#   799
XORS    001014           516#   740*   761*   765    766*   767*   769*   770    789    790*   791*   792    795*
                         802
$APTHD  001032           543    549#
$ASTAT= ****** U        1139   1154
$ATYC   003742          1110   1112#
$ATY1   003716          1108#
$ATY3   003724          1048   1109#
$ATY4   003734          1111#
$CHARC  003702          1065*  1075*  1082   1091*  1096#
$CKSWR= ****** U        1319
$CMTAG= ****** U         588
$CNTLG  003235           962#
$CNTLU  003230           933    961#
$CPUOP  001074           574#
$CRLF   003225           938    959*  1020   1064   1105
$DEVCT  001056           565#
$ENDAD  002060           530    615    700    718    726    734#   845
$ENV    001066           570#   635    684    702    720   1043   1117   1141
$ENVM   001067           571#   611   1045   1050   1119
$ETABL  001066           569#
$ETEND  001076           555    581#
$FATAL  001050           562#   584*   704*   722*   856*  1145*
$FFLG   004162          1108*  1111*  1139   1148*  1156#
$FILLC  003714          1068   1103#
$FILLS  003713          1102#
$GTSWR= ****** U        1318
$HIBTS  001032           550#
$HIOCT  003422          1008*  1019#
$ILLUP  004552          1243   1259   1276#
$LF     003226           948    960#  1020   1105
$LFLG   004161          1149*  1155#
$MAIL   001046           551    555    560#   610   1043
$MBADR  001034           551#
$MFLG   004160          1109*  1115   1150*  1154#
$MNEW   003253           965#
$MSGAD  001062           567#  1125*  1128
$MSGLG  001064           568#  1130*
$MSGTY  001046           561#   585*   705*   723*   857*  1123   1131*  1143   1147*
$MSWR   003242           963#
$NULL   003712          1070   1101#
$OCNT   004406          1194*  1223*  1236#
$OMODE  004410          1189*  1193*  1198   1201*  1212*  1238#
$PASS   001054           564#   610*   730*
$PASTM  001040           553#
$POWER  004560          1274   1279#
$PWRDN  004412           592   1243#  1271
$PWRMG  004546          1274#
$PWRUP  004464          1253   1259#
```

```
$QUES   003224            941     958*   1017    1020    1105
$RDCHR  002640            878*   1319
$RDDEC= ****** U         1322
$RDLIN  002760            906*   1320
$RDOCT  003264            981*   1321
$RDSZ = 000010            899*
$R2A  = ****** U         1322
$SAVRE= ****** U         1322
$SAVR6  004556           1252*   1260    1261*   1262*   1278*
$SETUP= 000014            500*    589     590     592     594     867     967
$STUP = 177777            500*
$SVPC = 001032            528*    533
$SWR  = 000000            379*   1275
$SWREG  001070            572*    613
$TESTN  001052            563*    586*
$TKB    002636            864*    882     888
$TKS    002634            863*    880     886
$TPB    003710           1088*   1100*
$TPFLG  003715           1037    1104*
$TPS    003706           1086    1099*
$TRAP   004570            590    1290*
$TRAP2  004612           1301*   1312
$TRP  = 000010           1305*   1314*   1315*   1316*   1317*   1319    1320*   1321*   1322*
$TRPAD  004624           1295    1312*
$TSTM   001036            552*
$TTYIN  003214            908     909     921     939     953     957*
$TYPBN= ****** U         1317
$TYPDS= ****** U         1317
$TYPE   003424           1037*   1136    1305    1313
$TYPEC  003636           1067    1074    1081    1086*   1087
$TYPEX  003704           1092    1094    1097*
$TYPOC  004210           1192*   1314
$TYPON  004224           1191    1194*   1316
$TYPOS  004164           1187*   1315
$UNIT   001060            566*
$UNITM  001042            554*
$USWR   001072            573*
$OFILL  004407           1163*   1192*   1202    1237*
.     = 006331            490*    493*    497*    500*    509*    528     529*    531*    533*    539     540*    542*    544*
                          863     864     957*    958     959     960     967    1020    1099    1100    1101    1102    1103
                         1104    1105    1157*   1255    1277
.$ASTA= ****** U         1109    1112
.$X   = 001032            539*    544
```

```
COMMEN    490#
ENDCOM    490#
ERPOR     384#
ESCAPE    490#
GETPRI    490#
GETSWR    490#
MULT      490#
NEWTST    490#
POP       379#   490#  1009  1151  1152  1264  1265
PUSH      379#   490#   983  1112  1114  1135  1245  1251
REPORT    490#
SCOPE     385#
SETPRI    490#
SETTRA   1305#  1314  1315  1316  1319  1320  1321
SETUP     379#   490#   587
SKIP      490#
SLASH     490#
SPACE     490#
STARS     379#   490#   526   536   538   545   558   862   870   899   969  1022  1107  1164  1241
         1257  1284
SWRSU     490#   594#
TRMTRP   1305#
TYPBIN    490#
TYPDEC    490#
TYPNAM    490#
TYPNUM    490#
TYPOCS    490#
TYPOCT    490#
TYPTXT    490#
$$ESCA    490#
$$NEWT    490#
$$SET    1305#  1314  1315  1316  1319  1320  1321
$$SETM    610#
$$SKIP    490#
.EQUAT    379#   380
.SETUP    379#   500
.$ACT1    379#   524
.$APTB    379#   556
.$APTH    379#   534
.$APTY    379#  1105
.$CATC    379#   491
.$POWE    379#  1239
.$RDOC    379#   967
.$READ    379#   860
.$TRAP    379#  1282
.$TYPE    379#  1020
.$TYPO    379#  1162

. ABS.   006331     000

 ERRORS DETECTED:  0
 DEFAULT GLOBALS GENERATED:  0

 DZM9AD.BIN,DZM9AD.LST/CRF/SOL/NL:TOC=DZM9AD.P11
```

RUN-TIME: 11 4 .5 SECONDS
RUN-TIME RATIO: 632/16=38.6
CORE USED: 20K (40 PAGES)

NO3