MS11

DIAGNOSTIC

MD-11-DCMSB-B

EP-DCMSB-B-DL-A

FICHE 1 OF 1

NOV 1976

digital

MADE IN USA

```
                                .REM %
```

IDENTIFICATION

PRODUCT CODE:	MAINDEC-11-DCMSB-B-D
PRODUCT NAME:	MOS SEMICONDUCTOR MEMORY DIAGNOSTIC
DATE CREATED:	23-APR-1973
MAINTAINER:	DIAGNOSTIC GROUP
AUTHOR:	SUB MALLICK

1.0 ABSTRACT

THIS PROGRAM IS TO BE USED TO TEST MOS SEMICONDUCTOR MEMORY ONLY AFTER ALL OTHER MEMORY DIAGNOSTICS HAVE BEEN RUN. THIS PROGRAM ONLY CHECKS REFRESH CYCLE PROBLEMS OF MOS MEMORY ON PDP-11/45.

2.0 REQUIREMENTS

2.1 EQUIPMENT

PDP-11/45 STANDARD COMPUTER WITH MOS SEMICONDUCTOR MEMORY.

2.2 STORAGE

THE ROUTINE USES MEMORY 0 TO 11510

3.0 LOADING PROCEDURE

PROCEDURE FOR NORMAL ABSOLUTE TAPES SHOULD BE FOLLOWED.

3.0.1 RELOCATION PROCEDURE

THIS PROGRAM IS RELOCATABLE.
FOLLOW NORMAL RELOCATION PROCEDURE.
THAT IS SET SWITCH "0". TO INDICATE THAT RELOCATION IS REQUIRED. THEN SET THE SWITCHES WHERE THE PROGRAM IS TO START LOADING. THEN HIT "START" TO LOAD TAPE.
ALL PROGRAM ADDRESSES BECOME INCREASED BY THIS RELOCATION FACTOR

FOR EXAMPLE IF SWITCH 12 AND 0 WAS SET BEFORE LOADING THEN 40000 IS THE RELOCATION FACTOR AND HENCE 40204 IS THE STARTING ADDRESS

4.0 STARTING PROCEDURE

4.1 CONTROL SWITCH SETTINGS

BASIC SWITCH REGISTER SETTINGS ARE:

SWITCH	USE
15	HALT ON ERROR
14	0 - DISABLE PARITY CHECKING 1 - ENABLE PARITY CHECKING
13	INHIBIT ERROR TYPEOUTS

```
                                        12          0 - SET PROGRAM TO CATCH BOTH BIT
                                                        DROPS AND PICKUPS
                                                    1 - SET PROGRAM AS INDICATED BY SWITCH 11
                                        11          0 - SET PROGRAM TO CATCH BIT DROPS
                                                    1 - SET PROGRAM TO CATCH BIT PICKUPS
                                        10          0 - PRINT END PASS
                                                    1 - BELL ON ERROR
                                         9          0 - PRINT FIRST ERROR PER PRECHARGE
                                                    1 - PRINT ALL ERRORS PER PRECHARGE
                                         8          INHIBIT PRINTING DATA HEADER
                                         7          0=TEST READ AND WRITE FROM OUTSIDE OF THE CHIP
                                                    1=TEST INSTRUCTIONS IN THE CHIP
                                         1          LOCK ON PRESENT MOD
                                         0          LOCK ON PRESENT PRECHARGE


                    4.2     STARTING ADDRESSES

                            200 = STARTING ADDRESS IF ALL OF MEMORY IS TO BE TESTED
                                  AND NO LIMITS CHOSEN
                            204 = STARTING ADDRESS IF LIMITS ARE TO BE CHOSEN
                            210 = RESTART ADDRESS WITH LAST SPECIFIED LIMITS


                    4.3     PROGRAM AND/OR OPERATOR ACTION


4.3.1   OPERATOR ACTION

        THERE ARE TWO MODES OF OPERATION FOR THIS PROGRAM
        A) NO LIMITS CHOSEN
        B) LIMITS CHOSEN



A)  NO LIMITS CHOSEN

        FOR THIS MODE OF OPERATION RELOCATION CANNOT BE USED
        THAT IS BEFORE LOADING TAPE, SWITCH 0 MUST BE 0.
        STARTING ADDRESS IS 200.
        THE MEMORY SIZE IS PRINTED IN OCTAL
         THE PROGRAM TYPES OUT "SET
        SWITCHES FOR OPERATING MODE.  REFER SECTION 4.1

        IE. PICKUPS OR DROPS OR BOTH THEN HIT CONTINUE "
        ON SUBSEQUENT STARTS THE TYPEOUT IS SHORTER.
        NOW CHOOSE ALL SWITCH SETTINGS.  SWITCH 11 AND 12 MUST BE
        LEFT ON AS DESIRED AND IF CHANGED DURING OPERATION NO
        CHANGE WILL TAKE PLACE TILL THE "END PASS" PRINT OUT. THEN ALL OF
        MEMORY IS CHECKED STARTING FROM THE SECOND 4K TILL THE LAST
        ADDRESS IS REACHED.  THEN THE PROGRAM IS MOVED INTO THE
        SECOND 4K AND THE FIRST 4K IS CHECKED.
        THIS MODE OF OPERATION DESTROYES THE LOADER AND IF
        THE PROGRAM HAS MOVED INTO THE SECOND 4K AND AN OPERATOR
        HALT IS PERFORMED THEN RESTART ADDRESS IS NOT 200 BUT
```

```
                                40200.

                                B)  LIMITS CHOSEN

                                IN THIS MODE OF OPERATION RELOCATION CAN BE USED.  TO
                                RELOCATE PROGRAM REFER TO SECTION 3.0.1.
                                STARTING ADDRESS IS 204 OR RELOCATION FACTOR PLUS 204.
                                THE MEMORY SIZE IS PRINTED IN OCTAL THEN
                                THE PROGRAM TYPES "TYPE "STARTING BANK" NUMBER AND
                                CARRIAGE RETURN.  REFER TO SECTION 4.3.1 BEFORE CHOOSING"
                                OR SHORTER TYPE OUTS ON SUBSEQUENT STARTS THIS IS THE SECTION
                                REFERED TO

                                TYPE THE 4K BANK WHERE YOU WISH TO BEGIN TESTING
                                      TO                                  TO
                                TYPE  START                       TYPE  START
                                      AT                                  AT
                                0     000000 (0K)
                                1     020000 (4K)                 20    400000 (64K)
                                2     040000 (8K)                 21    420000 (68K)
                                3     060000 (12K)                22    440000 (72K)
                                4     100000 (16K)                23    460000 (76K)
                                5     120000 (20K)                24    500000 (80K)
                                6     140000 (24K)                25    520000 (84K)
                                7     160000 (28K)                26    540000 (88K)
                                10    200000 (32K)                27    560000 (92K)
                                11    220000 (36K)                30    600000 (96K)
                                12    240000 (40K)                31    620000 (100K)
                                13    260000 (44K)                32    640000 (104K)
                                14    300000 (48K)                33    660000 (108K)
                                15    320000 (52K)                34    700000 (112K)
                                16    340000 (56K)                35    720000 (116K)
                                17    360000 (60K)                36    740000 (120K)

                                TYPE ONLY NUMBERS SHOWN !!!

                                MEMORY CAN ONLY BE TESTED IN 4K GROUPS STARTING AND
                                FINISHING AT 4K BOUNDARY
                                FOR EXAMPLE IF YOU WISH TO TEST THE THIRD 4K BANK
                                (STARTING ADDRESS 40000) TYPE "2".

                                THEN THE PROGRAM TYPES "TYPE NUMBER OF 4K BANKS TO BE TESTED AND
                                CARRIAGE RETURN".  ON SUBSEQUENT STARTS TYPEOUT IS SHORTER.
                                AFTER TYPING THE APPROPIATE NUMBER AND CARRIAGE RETURN
                                 THE PROGRAM TYPES " SET SWITCHES FOR OPERATING MODE.  REFER
                                SECTION 4.1  I.E. PICKUPS OR DROPS OR BOTH THEN HIT
                                CONTINUE"  ON SUBSEQUENT STARTS TYPEOUTS ARE SHORTER.
                                THIS MEANS NOW IS THE TIME TO CHOOSE ALL SWITCH
                                SETTINGS.  SWITCH 11 AND 12 MUST BE LEFT ON AS DESIRED
                                AND IF CHANGED DURING OPERATION NO CHANGE WILL TAKE
                                PLACE TILL THE "END PASS" PRINT OUT.
                                CHECKING IS NOW STARTED

                                IF STARTING BANK AND NUMBER OF BANKS ARE INCORRECTLY
                                CHOSEN THEN RUBOUT MAY BE HIT TO DELETE THE LAST CHARACTER
                                IF CARRIAGE RETURN HAS NOT ALREADY BEEN HIT.  IF CARRIAGE RETURN
```

HAS BEEN HIT THEN RESTARTING IS NECESSARY. IF NO CORRECTION
IS MADE THEN THE PROGRAM MAY DESTROY ITSELF OR IT MAY TIME
OUT ON A NON EXISTANT MEMORY.

IN ORDER TO TEST THE FIRST 4K OF MEMORY WITH 204
STARTING THAT IS BY CHOOSING LIMITS RELOCATE THE
PROGRAM BY LOADING IT INTO A HIGHER AREA OF MEMORY SAY AT 40000
AND TEST THE FIRST FOUR K BY MAKING THE STARTING BANK "0"
AND NUMBER OF BANKS "1"

4.3.2 PROGRAM ACTION

THE PROGRAM WILL FIRST FILL A 4K BLOCK WITH WHAT EVER
IS IN "BCKGRD" .THEN THE PROGRAM WILL WRITE "TSTPAT" INTO ONE LOCATION
CALLED TSTTPAT . IN ANOTHER LOCATION CALLED TEST-LOCATION
IT WILL WRITE A CLR RO (OR ALL "0" OR ALL"1"
DEPENDING ON SWITCHES 7 AND 12. LOC TSTPAT AND TEST-
LOCATION WILL HAVE BIT 8 OF OPPOSITE VALUE. THE TEST-
LOCATION WILL BE OPERATED ON 1000 TIMES THEN THE 4K WILL BE
CHECKED. THEN ONE WILL BE ADDED TO LOCATION TSTPAT AND
THE PROCESS REPEATED TILL ONE 4K IS DONE THEN THE NEXT
4K WILL BE STARTED.

5.0 OPERATING PROCEDURE

5.1 OPERATIONAL SWITCH SETTINGS

SEE 4.1

6.0 ERRORS

6.1 ERROR PRINTOUT

THERE ARE THREE TYPES OF ERROR PRINTOUTS

A) BACK GROUND CHANGED
B) INSTRUCTION CHANGED
C) TEST PATTERN

EXAMPLE

BACK GROUND CHANGED

ERROR PC	BCKGRD ADDRESS	EXPECT DATA	RCV'D DATA	COL	ROW	PRECHARGE
103166	006500	177777	177577	6	23	0

```
                        THE COL=COLIMN AND ROW PRINTOUTS ARE FOR ENGINEERING
                        USE WHICH TELLS WHAT BIT WITHIN A CHIP IS FAULTY.
                        IF INFORMATION WITHIN A CHIP IS NOT NECESSARY DISREGARD
                        THESE TWO (COL, ROW)


                6.2     ERROR RECOVERY

                        IN GENERAL, TEST FAILURES WILL PRINTOUT AN ERROR MESSAGE
                        AND CONTINUE.  IF THE "HALT ON ERROR" SWITCH IS SET,
                        HITTING CONTINUE WILL RECOVER.  IF THE PROGRAM HANGS
                        UP IN A LOOP, THE ERROR IS LIKELY TO BE A SIGNAL WHICH
                        WAS NEVER RECEIVED.  IF A HALT OCCURS IN THE TRAP AND
                        VECTER AREA THE PROGRAM MUST BE RESTARTED.  IF THE
                        PROGRAM HALTS IN THE MAIN FLOW, CONSULT THE LISTING
                        IF NO MESSAGE IS TYPED OUT.


                7.0     RESTRICTIONS
                        RELOCATION CANNOT BE DONE INTO 160000
                        IF PARITY CHECKING IS ENABLED THEN ON AN ERROR IT
                        IS NOT GUARANTEED WHICH WILL SHOW UP: THE PARITY ERROR
                        OR THE BIT ERROR.
                        IT IS RECOMMENDED THAT PARITY IS ENABLED ONLY WHEN
                        LOOKING FOR PARITY ERRORS.



                8.0     MSCELLANEOUS


                8.1     EXECUTION TIME

                        APPOXIMATELY 10 MIN PER 4K TESTED.  (BOTH PICKUPS & DROPS.)
                %
                .ENABL  ABS
                .TITLE  TEST MOS MEMORY
                ;COPYRIGHT 1973 DIGITAL EQUIPMENT CORP., MAYNARD, MASS. 01754
                ;PROGRAM BY JIM LACEY, SUB MALLICK
                ;*****************************************************************************

                ;OPERATIONAL SWITCH SETTINGS
                ;
                ;       SWITCH          USE
                ;       ------          --------------------
                ;         15            HALT ON ERROR
                ;         14            0 - DISABLE PARITY CHECKING
                ;                       1 - ENABLE PARITY CHECKING
                ;         13            INHIBIT ERROR TYPEOUTS
                ;         12            0 - SET PROGRAM TO CATCH BOTH BIT
                ;                           DROPS AND PICKUPS
                ;                       1 - SET PROGRAM AS INDICATED BY SWITCH 11
                ;         11            0 - SET PROGRAM TO CATCH BIT DROPS
                ;                       1 - SET PROGRAM TO CATCH BIT PICKUPS
                ;         10            0 - PRINT END PASS
```

```
                                ;                       1 - BELL ON ERROR
                                ;               9       0 - PRINT FIRST ERROR PER PRECHARGE
                                ;                       1 - PRINT ALL ERRORS PER PRECHARGE
                                ;               8       INHIBIT PRINTING DATA HEADER
                                ;               7       0=TEST READ AND WRITE FROM OUTSIDE OF THE CHIP
                                ;                       1=TEST INSTRUCTIONS IN THE CHIP
                                ;               1       LOCK ON PRESENT MOD
                                ;               0       LOCK ON PRESENT PRECHARGE
                                ;*******************************************************************************


                                ;BASIC DEFINITIONS
                                .EQUIV  EMT,HLT                 ;BASIC DEFINITION OF ERROR CALL
                                .EQUIV  IOT,SCOPE               ;BASIC DEFINITION OF SCOPE CALL
177776                          PS=     177776                  ;PROCESSOR STATUS WORD
                                .EQUIV  PS,PSW
177570                          SWR=    177570                  ;SWITCH REGISTER
177570                          DISPLAY=SWR

                                ;REGISTER DEFINITION
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
                                .EQUIV  R6,SP                   ;STACK POINTER
                                .EQUIV  R7,PC                   ;PROGRAM COUNTER

                                ;SWITCH DEFINITION
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

                                ;MISCELLANEOUS BIT ASSIGNMENT
            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

                                ;VECTOR ADDRESSES
            000004              ERRVEC= 4
            000010              RESVEC= 10
            000014              TBITVEC=14
            000014              TRTVEC= 14
            000014              BPTVEC= 14
            000020              IOTVEC= 20
            000024              PWRVEC= 24
            000030              EMTVEC= 30
            000034              TRAPVEC=34
            000114              PARVEC=114
            000250              MMVEC=250
```

```
                    000100                          .=100
000100  000000      FIRST:  .WORD   0

                            ;KT11-D STATUS REGISTER ADDRESSES

000102  177572      SR0:    177572
000104  177574      SR1:    177574
000106  177576      SR2:    177576
000110  172516      SR3:    172516

                            ;KERNAL PAGE DESCRIPTOR REGISTERS

000112  172300      KIPDR0: 172300
000114  172302      KIPDR1: 172302
000116  172304      KIPDR2: 172304
000120  172306      KIPDR3: 172306
000122  172310      KIPDR4: 172310
000124  172312      KIPDR5: 172312
000126  172314      KIPDR6: 172314
000130  172316      KIPDR7: 172316

                            ;KERNAL PAGE ADDRESS REGISTERS

000132  172340      KIPAR0: 172340
000134  172342      KIPAR1: 172342
000136  172344      KIPAR2: 172344
000140  172346      KIPAR3: 172346
000142  172350      KIPAR4: 172350
000144  172352      KIPAR5: 172352
000146  172354      KIPAR6: 172354
000150  172356      KIPAR7: 172356

                            ;KT11 VECTOR ADDRESS

000152  000250  000252  SEGVEC: 250,252
```

```
                    000200                          .=200
000200  000167  003250                      JMP     BEGN11          ;JUMP TO START TO TEST ALL OF
                                                                    ;MEMORY WITHOUT CHOOSING LIMITS
000204  000167  002536                      JMP     BEGIN1          ;JUMP TO START CHOOSE LIMITS
000210  000167  003666                      JMP     BEGIN3          ;JUMP TO RESTART USING LAST SPECIFIED LIMITS


000214  000000                  IN2AD:  0
000216  000000                  FRSTAD: 0
```

```
                        000300                          .=300
                                                        
                                                        
000300  016701  002364          RESTOR: MOV     TO1,R1
000304  016702  005022                  MOV     FROM,R2
000310  016703  005014                  MOV     LENGTH,R3
000314  012122                  1$:     MOV     (R1)+,(R2)+
000316  077302                          SOB     R3,1$
000320  000137  000204                  JMP     @#204
                                                        
                                                        
                                ;***************************************************************************
                                                        
                                                        
                                                        
                                                        
000324  010704                  LOADER: MOV     PC,R4
000326  062704  011164                  ADD     #LAST+2-.,R4
000332  016702  002334                  MOV     LODER,R2
000336  012701  003132                  MOV     #17774-LAST/2,R1
000342  012442                  1$:     MOV     (R4)+,-(R2)
000344  077102                          SOB     R1,1$
000346  004767  006666                  JSR     PC,TYPE
000352  011466                          FIN
000354  000000                          HALT
                                                        
                                                        
                                                        
```

```
                        ;*****************************************************************************
001100                          .=1100

                        ;*****************************************************************************

                        ;ROUTINE TO TYPE ASCII MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.


                        ;THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
                        ;NOTE1: $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
                        ;NOTE2: $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.

                        ;CALL:
                        ;       JSR     PC,$TYPE                ;CALL TYPE ROUTINE
                        ;       MESADDR                         ;FIRST ADRESS OF MESSAGE

001100  177564          $TPS:   177564                          ;TTY PRINTER STATUS REG. ADDRESS
001102  177566          $TPB:   177566                          ;TTY PRINTER BUFFER REG. ADDRESS
001104  000             $NULL:  .BYTE   0                       ;CONTAINS NULL CHARACTER FOR FILLS
001105  002             $FILLS: .BYTE   2                       ;CONTAINS # OF FILLER CHARACTERS REQUIRED
001106  000             $TPFLG: .BYTE   0                       ;"TERMINAL AVAILABLE" FLAG (0=YES)
001107  000                     .BYTE   0                       ;RESERVED

001110  105767  177772  $TYPE:  TSTB    $TPFLG                  ;IS THERE A TERMINAL?
001114  001402                  BEQ     6$                      ;BR IF YES
001116  000000                  HALT                            ;HALT HERE IF NO TERMINAL
001120  000407                  BR      7$                      ;LEAVE
001122  010046          6$:     MOV     R0,-(SP)                ;SAVE R0
001124  017600  000002          MOV     @2(SP),R0               ;GET ADDRESS OF ASCIZ STRING
001130  112046          1$:     MOVB    (R0)+,-(SP)             ;PUSH CHARACTER TO BE TYPED ONTO STACK
001132  001005                  BNE     2$                      ;BR IF IT ISN'T THE TERMINATOR
001134  005726                  TST     (SP)+                   ;IF TERMINATOR POP IT OFF THE STACK
001136  012600                  MOV     (SP)+,R0                ;RESTORE R0
001140  062716  000002  7$:     ADD     #2,(SP)                 ;ADJUST RETURN PC
001144  000207                  RTS     PC                      ;RETURN
001146  004767  000026  2$:     JSR     PC,5$                   ;GO TYPE THIS CHARACTER
001152  122726  000012  3$:     CMPB    #12,(SP)+               ;CHECK IF THE CHAR. TYPED WAS A  LINE FEED
001156  001364                  BNE     1$                      ;GO GET NEXT CHAR. IF NOT LINE FEED
001160  016746  177720          MOV     $NULL,-(SP)             ;GET # OF FILLER CHARS. NEEDED
                                                                ;AND THE NULL CHAR.
001164  105366  000001  4$:     DECB    1(SP)                   ;DOES A NULL NEED TO BE TYPED?
001170  002770                  BLT     3$                      ;BR IF NO--GO POP THE NULL OFF OF STACK
001172  004767  000002          JSR     PC,5$                   ;GO TYPE A NULL
001176  000772                  BR      4$                      ;LOOP
001200  105777  177674  5$:     TSTB    @$TPS                   ;WAIT UNTIL PRINTER IS READY
001204  100375                  BPL     5$
001206  116677  000002  177666  MOVB    2(SP),@$TPB             ;LOAD CHAR TO BE TYPED INTO DATA REG.
001214  000207                  RTS     PC
001216  000062                  .BLKB   62                      ;RESERVE SOME MORE CORE FOR OVERLAY CAPIBILITIES


                        ;*****************************************************************************

                        ;ROUTINE TO SIZE MEMORY
```

```
                                        ;CALL:
                                        ;       JSR     PC,$SIZE
                                        ;       RETURN
                                        ;$LSTAD WILL CONTAIN THE LAST VIRTUAL ADDRESS OF THE LAST BANK
                                        ;$LSTBK WILL CONTAIN THE LAST BANK AS A SAF
                                        ;$KT11 IS THE MEMORY MANAGEMENT KEY
                                        ;BIT07 = 0 DON'T USE MEMORY MANAGEMENT
                                        ;BIT15 = 0 DON'T HAVE MEMORY MANAGEMENT OPTION

001300  010546                  $SIZE:  MOV     R5,-(SP)                ;SAVE R5 ON THE STACK
001302  010446                          MOV     R4,-(SP)
001304  010346                          MOV     R3,-(SP)
001306  010146                          MOV     R1,-(SP)                ;SAVE R1 ON THE STACK
001310  010046                          MOV     R0,-(SP)                ;SAVE R0 ON THE STACK
001312  010605                          MOV     SP,R5                   ;SAVE THE STACK POINTER
001314  012701  003776                  MOV     #3776,R1                ;
001320  105727                          TSTB    (PC)+                   ;USE MEMORY MANAGEMENT?
001322  000000                  $KT11:  0
001324  100152                          BPL     $CORE                   ;BR IF NO
001326  010746                  $SEG:   MOV     PC,-(SP)
001330  062716  000322                  ADD     #$CORE-.,(SP)
001334  012637  000004                  MOV     (SP)+,@#ERRVEC          ;SET FOR TIMEOUT
001340  005777  176536                  TST     @$R0                    ;KT11 ARE YOU THERE?
001344  052767  100000  177750          BIS     #BIT15,$KT11            ;YES--SET KT11 KEY
001352  012703  077406                  MOV     #77406,R3
001356  010377  176530                  MOV     R3,@KIPDR0
001362  010377  176526                  MOV     R3,@KIPDR1
001366  010377  176524                  MOV     R3,@KIPDR2
001372  010377  176522                  MOV     R3,@KIPDR3
001376  010377  176520                  MOV     R3,@KIPDR4
001402  010377  176516                  MOV     R3,@KIPDR5
001406  010377  176514                  MOV     R3,@KIPDR6
001412  010377  176512                  MOV     R3,@KIPDR7

001416  005077  176510                  CLR     @KIPAR0
001422  012777  000200  176504          MOV     #200,@KIPAR1
001430  012777  000400  176500          MOV     #400,@KIPAR2
001436  012777  000600  176474          MOV     #600,@KIPAR3
001444  012777  001000  176470          MOV     #1000,@KIPAR4
001452  012777  001200  176464          MOV     #1200,@KIPAR5
001460  012777  001400  176460          MOV     #1400,@KIPAR6
001466  012777  007600  176454          MOV     #7600,@KIPAR7

001474  010703                          MOV     PC,R3
001476  000241                          CLC
001500  006103                          ROL     R3
001502  006103                          ROL     R3
001504  006103                          ROL     R3
001506  006103                          ROL     R3
001510  042703  177770                  BIC     #177770,R3
001514  022703  000006                  CMP     #6,R3
001520  001403                          BEQ     2$
```

```
                                                MOV     KIPAR6,R4
001526  000425                                  BR      3$


                                        2$:     MOV     KIPAR5,R4
001534  012767  123776  000070                MOV     #123776,1$+2
001542  012703  120000                        MOV     #120000,R3
001546  010367  003354                        MOV     R3,SEG0+2
001552  010367  003356                        MOV     R3,SEG0+10
001556  010367  003376                        MOV     R3,SEG10+2
001562  010367  003400                        MOV     R3,SEG10+10
001566  012767  137776  003346                MOV     #137776,SEG0+16
001574  012767  137776  003372                MOV     #137776,SEG10+16


001602  010467  001060                3$:     MOV     R4,PAGE
001606  005014                                CLR     (R4)


001610  012777  000001  176264                MOV     #1,@SR0                 ;TURN ON MEMORY MANAGEMENT
001616  010746                                MOV     PC,-(SP)
001620  062716  000026                        ADD     #$KTOUT-.,(SP)
001624  012637  000004                        MOV     (SP)+,@#ERRVEC  ;SET FOR TIME OUT
001630  005737  143776                1$:     TST     @#143776                ;TRAP ON NON-EX-MEM
001634  062714  000040                        ADD     #40,(R4)                ;MAKE A 1K STEP
001640  027714  176304                        CMP     @KIPAR7,(R4)            ;LAST ONE?
001644  003371                                BGT     1$                      ;NO--TRY IT
001646  011400                        $KTOUT: MOV     (R4),R0         ;GET LAST BANK+1
001650  000420                                BR      $SIZEX
001652  010746                        $CORE:  MOV     PC,-(SP)
001654  062716  000032                        ADD     #$COROUT-.,(SP)
001660  012637  000004                        MOV     (SP)+,@#ERRVEC  ;SET FOR TIMEOUT
001664  005000                                CLR     R0                      ;SET UP BANK
001666  062701  004000                1$:     ADD     #4000,R1                ;INCREMENT BY 1K
001672  062700  000040                        ADD     #40,R0                  ;1K STEP
001676  005711                                TST     (R1)                    ;TRAP ON TIME OUT
001700  022701  177776                        CMP     #177776,R1              ;LAST ONE
001704  001370                                BNE     1$                      ;NO--TRY AGAIN
001706  162701  004000                $COROUT:        SUB     #4000,R1        ;
001712  162700  000040                $SIZEX: SUB     #40,R0                  ;DROP BACK
001716  012737  000006  000004                MOV     #6,@#ERRVEC             ;SET FOR ERRORS
001724  010506                                MOV     R5,SP                   ;RESTORE THE STACK
001726  010167  000032                        MOV     R1,$LSTAD               ;
001732  010067  000030                        MOV     R0,$LSTBK               ;
001736  005767  177360                        TST     $KT11                   ;IS MEMORY MANAGEMENT THERE
001742  100002                                BPL     1$                      ;IF NOT BRANCH
001744  005077  176132                        CLR     @SR0                    ;DISABLE MEMORY MANAGEMENT
001750  012600                        1$:     MOV     (SP)+,R0                ;RESTORE R0
001752  012601                                MOV     (SP)+,R1                ;RESTORE R1
001754  012603                                MOV     (SP)+,R3
001756  012604                                MOV     (SP)+,R4
001760  012605                                MOV     (SP)+,R5                ;RESTORE R5
```

```
 659 001762 000207                        RTS     PC
 660 001764 000000              $LSTAD:  .WORD    0                  ;CONTAINS THE LAST ADDRESS
 661 001766 000000              $LSTBK:  .WORD    0                  ;CONTAINS THE LAST BANK
```

```
                                ;ROUTINE TO GET TYPED ASCII DATA AND CONVERT TO OCTAL.
                                ;LEAVES THE OCTAL DATA IN THE ADDRESS FOLLOWING THE CALL.
                                ;CALL:  RECD            ;(A TRAP TYPE INST.)
                                ;       .WORD   0               ;CONTAINS RETURNED DATA
                                ;       CONTINUE                ;RETURNS HERE

001770  177560                 TKS:    177560
001772  177562                 TKB:    177562

001774  000240                 .RECD:  NOP
001776  005046                         CLR     -(SP)
002000  105777  177764         1$:     TSTB    @TKS            ;WAIT FOR USER RESPONSE
002004  100375                         BPL     1$
002006  117746  177760                 MOVB    @TKB,-(SP)      ;GET ASCII CHARACTER
002012  042716  177600                 BIC     #177600,(SP)    ;
002016  122716  000177                 CMPB    #177,(SP)       ;CHECK IF RUBOUT
002022  001013                         BNE     2$              ;BRANCH IF NOT RUBOUT
002024  012767  000134  000142         MOV     #'\,    4$
002032  004767  005202                 JSR     PC,'TYPE                ;ECHO BACK SLASH
002036  002174                         4$
002040  005726                 6$:     TST     (SP)+           ;POP LAST TYPED CHAR OFF THE STACK
002042  006016                         ROR     (SP)            ;SHIFT LAST TYPD CHARACTER
002044  006216                         ASR     (SP)
002046  006216                         ASR     (SP)
002050  000753                         BR      1$              ;AND WAIT FOR NEXT CHAR.


002052  122716  000015         2$:     CMPB    #15,(SP)        ;CHECK IF CARRIAGE RETURN
002056  001011                         BNE     3$              ;BRANCH IF NOT A CARRIAGE RETURN
002060  004767  005154                 JSR     PC,TYPE
002064  002647                         $CRLF
002066  005726                         TST     (SP)+           ;POP CARRIAGE RETURN OFF THE STACK
002070  012676  000000                 MOV     (SP)+,@(SP)     ;PUT DATA IN ADDREESS FOLLOWING CALL
002074  062716  000002                 ADD     #2,(SP)
002100  000207                         RTS     PC
002102  111667  000066         3$:     MOVB    (SP),4$

002106  004767  005126                 JSR     PC,TYPE         ;GO TO PRINT ROUTINE
002112  002174                         4$                      ;ECHO CHARACTER
002114  042767  177707  000052         BIC     #177707,4$
002122  022767  000060  000044         CMP     #60,4$
002130  001407                         BEQ     5$
002132  004767  005102                 JSR     PC,TYPE
002136  002647                         $CRLF
002140  004767  005074                 JSR     PC,TYPE
002144  011217                         SORRY
002146  000734                         BR      6$
002150  042716  177770         5$:     BIC     #177770,(SP)    ;SAVE ONLY 3 MSBS
002154  006366  000002                 ASL     2(SP)           ;SHIFT LAST CHARACTER OVER
002160  006366  000002                 ASL     2(SP)           ;THREE PLACES
002164  006366  000002                 ASL     2(SP)
002170  052616                         BIS     (SP)+,(SP)      ;INSET OCTAL CHAR
002172  000702                         BR      1$              ;GO WAIT FOR NEXT CHAR.
002174  000000         4$:     .WORD   0               ;CONTAINS CHARACTER TO BE TYPED
```

```
                                        ;ROUTINE TO SET PARITY ENABLE ON MA/MF PARITY MEMORIES
                 172100                         PARCSR= 172100  ;ADDRESS OF FIRST POSSIBLE PARITY REG.


002176  012737  002254  000114  MAMF:   MOV     #PARSRV,@#PARVEC        ;LOAD VECTOR
002204  012737  000340  000116          MOV     #340,@#PARVEC+2         ;AND PRIORITY LEVEL
002212  012737  000006  000004          MOV     #ERRVEC+2,@#ERRVEC      ;SET TIME OUT TRAP VECTOR
002220  012737  000002  000006          MOV     #RTI,@#ERRVEC+2         ;DO RTI ON TIMEOUT TRAP
002226  012700  172100                  MOV     #PARCSR,R0              ;GET FIRST POSSIBLE ADDRESS
002232  012702  000001                  MOV     #1,R2                   ;SET REG. COUNTER

002236  012720  000001          1$:     MOV     #1,(R0)+                ;SET ACTION ENABLE IF THERE
002242  006302                          ASL     R2                      ;CHECK IF 16 REG. HAVE BEEN DONE
002244  103374                          BCC     1$                      ;REPEAT IF 16 NOT ENABLED
002246  005037  000006                  CLR     @#ERRVEC+2              ;RESTORE HALT ON TIMEOUT
002252  000207                          RTS     PC                      ;RETURN
```

```
                                        ;PARITY ERROR SERVICE ROUTINE
002254  004767  004760          PARSRV: JSR     PC,     TYPE    ;TYPE PARITY ERROR SO PARITY
002260  011131                          PARERR                  ;ERROR TEST STARTED
002262  010746                          MOV     PC,     -(SP)   ;MAKE 2$ POSITION INDEPENDENT
002264  062716  000120                  ADD     #2$-.,  (SP)    ;
002270  012637  000114                  MOV     (SP)+,@#PARVEC  ;SET PARITY ERROR TRAP
002274  010746                          MOV     PC,     -(SP)   ;MAKE 4$ POSITION INDEPENDENT
002276  062716  000216                  ADD     #4$-.,  (SP)    ;
002302  012637  000004                  MOV     (SP)+,  @#ERRVEC
002306  010746                          MOV     PC,     -(SP)   ;MAKE 5$ POSITION INDEPENDENT
002310  062716  000214                  ADD     #5$-.,  (SP)    ;
002314  012637  000250                  MOV     (SP)+,@#MMVEC   ;SET MEMORY MANAGEMENT ABORT TRAP
002320  005000                          CLR     R0              ;
002322  005767  176774                  TST     $KT11           ;IS MEM MANG. THERE?
002326  100024                          BPL     1$              ;IF NOT BRANCH
002330  012702  077406                  MOV     #77406,R2       ;SET UP MEM MANGT.
002334  005077  175572                  CLR     @KIPAR0
002340  010277  175546                  MOV     R2,@KIPDR0
002344  005077  175564                  CLR     @KIPAR1
002350  005077  175542                  CLR     @KIPDR2
002354  010277  175534                  MOV     R2,@KIPDR1
002360  012777  007600  175562          MOV     #7600,@KIPAR7
002366  010277  175536                  MOV     R2,@KIPDR7
002372  012777  000001  175502          MOV     #1,@SR0
                                                                ;ENABLE MEM MANGT.,


002400  005720                  1$:     TST     (R0)+           ;SCAN MEMORY FOR PARITY
002402  000776                          BR      1$


002404  004767  004630          2$:     JSR     PC,TYPE         ;TYPE LOCATION GIVING PARITY ERROR
002410  011021                          PAERR
002412  017777  175516  175522          MOV     @KIPAR1,@KIPAR4
002420  014046                          MOV     -(R0),-(SP)
002422  004067  004672                  JSR     R0,$B2OCT
002426     006                          .BYTE   6
002427     003                          .BYTE   3
002430  005767  176666          6$:     TST     $KT11
002434  100005                          BPL     3$
002436  005077  175440                  CLR     @SR0
002442  004767  004572                  JSR     PC,TYPE
002446  002647                          $CRLF
002450  000005                  3$:     RESET
002452  010746                          MOV     PC,-(SP)
002454  062716  177600                  ADD     #PARSRV-.,(SP)
002460  012637  000114                  MOV     (SP)+,@#PARVEC
002464  010746                          MOV     PC,-(SP)
002466  062716  175320                  ADD     #ERRVEC+2-.,(SP)
002472  012637  000004                  MOV     (SP)+,@#ERRVEC  ;RESET PARITY ERROR TRAP AND ERROR VECTOR
002476  012737  000252  000250          MOV     #MMVEC+2,@#MMVEC        ;RESET MEM MANGT. ABORT TRAP
002504  010746                          MOV     PC,-(SP)
002506  062716  001374                  ADD     #BEGIN3-.,(SP)
002512  000136                          JMP     @(SP)+          ;JUMP RESTART
```

```
                                                ;MEMORY MANAGEMENT ABORT ROUTINE
002524  062777  000200  175402  5$:   ADD     #200,@KIPAR1    ;ADD NEXT 4K TO ADDRESS
002532  012700  020000                MOV     #20000,R0       ;RESET VIRTUAL ADDRESS
002536  000002                        RTI
```

002514 004767 004520 4$: JSR PC, TYPE
002520 011061   NOPARE
002522 000742   BR   6$

```
                                ;*******************************************************************************

                                ;COMMON TAGS


002540  000000          $PASS:  .WORD   0               ;CONTAINS PASS COUNT
002542  000000          $TSTNM: .WORD   0               ;CONTAINS THE TEST NUMBER
002544  000000          $ICNT:  .WORD   0               ;CONTAINS SUBTEST ITERATION COUNT
002546  000000          $LPADR: .WORD   0               ;CONTAINS SCOPE LOOP ADDRESS
002550  000000          $LPERR: .WORD   0               ;CONTAINS SCOPE RETURN FOR ERRORS
002552  000000          $ERTTL: .WORD   0               ;CONTAINS TOTAL ERRORS DETECTED
002554     000          $ERFLG: .BYTE   0               ;CONTAINS ERROR FLAG
002555     000                  .BYTE   0               ;RESERVED--NOT TO BE USED
002556  000000  000000          .WORD   0,0             ;RESERVED--NOT TO BE USED
002562     000          $ITEMB: .BYTE   0               ;CONTAINS ITEM CONTROL BYTE
002563     000                  .BYTE   0               ;RESERVED--NOT TO BE USED
002564  000000          $HLTAD: .WORD   0               ;CONTAINS PC OF LAST HLT INSTRUCTION
002566  000000          $GDADR: .WORD   0               ;CONTAINS ADDRESS OF 'GOOD' DATA
002570  000000          $BDADR: .WORD   0               ;CONTAINS ADDRESS OF 'BAD' DATA
002572  000000          $GDDAT: .WORD   0               ;CONTAINS 'GOOD' DATA
002574  000000          $BDDAT: .WORD   0               ;CONTAINS 'BAD' DATA
002576  000000          $TMP0:  .WORD   0               ;USER DEFINED
002600  000000          $TMP1:  .WORD   0               ;USER DEFINED
002602  000000          $TMP2:  .WORD   0               ;USER DEFINED
002604  000000          TSTADR: .WORD   0               ;TEST ADDRESS
002606  000000          LOLMT:  .WORD   0               ;LOW LIMIT
002610  017776          HILMT:  .WORD   17776           ;HIGH LIMIT
002612  000000          MOD:    .WORD   0               ;MODULE NUMBER
002614  000000          COL:    .WORD   0               ;COLUMN NUMBER
002616  000000          ROW:    .WORD   0               ;ROW NUMBER
002620  000000          PCHRG:  .WORD   0               ;PRECHARGE
002622  023420          CYCNT:  .WORD   10000.          ;CYCLE COUNT
002624  177777          TSTPAT: .WORD   -1              ;TEST PATTERN
002626  177777          BCKGRD: .WORD   -1              ;BACK GROUND PATTERN
002630  000000          INSTAD: .WORD   0               ;ADDRESS OF INSTRUCTION UNDER TEST
002632  000400          MSKBIT: .WORD   BIT08           ;MASK BIT
002634  000000          SV400:  .WORD   0
002636  000000          SVIN1:  .WORD   0
002640  000000          SVIN2:  .WORD   0
002642  000000          ESCAPE: .WORD   0
002644  000207          $BELL:  .ASCIZ  <207>           ;CODE FOR BELL
002646     077          $QUES:  .ASCII  /?/             ;QUESTION MARK
002647     015          $CRLF:  .ASCII  <15>            ;CARRIAGE RETURN
002650  000012          $LF:    .ASCIZ  <12>            ;LINE FEED
002652  000000          PASFLG: .WORD   0
002654  000000          SELF:   .WORD   0
002656  177777          REDONE: .WORD   -1
002660  000000          LOBASE: .WORD   0
002662  000000          HIBASE: .WORD   0
002664  000000          LSTBK:  .WORD   0
002666  000000          PAGE:   .WORD   0
002670  000000          TO1:    .WORD   0               ;RELOCATE TO HERE IF 0 CHOSEN
002672  000000          LODER:  .WORD   0               ;LOADER IS HERE
```

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

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


002674                          $ERRTB:
                                ;ITEM 1
                                        EM1             ;TEST PATTERN
                                        DH1             ;ERROR  FAILING
                                                        ;PC     ADDRESS EXPECT  RCV'D   COL     ROW     PRECHARGE
                                        DT1             ;$HLTAD $BDADR  $GDDAT  $BDDAT  $TMP0   $TMP1   $TMP2
                                        DF1             ;ALL NUMBERS ARE TYPED IN OCTAL

                                ;ITEM 2
                                        EM2             ;INSTRUCTION CHANGED
                                        DH2             ;ERROR  INST    EXPECT  RCV'D,
                                                        ;PC     ADDRESS DATA    DATA    COL     ROW     PRECHARGE
                                        DT1             ;$HLTAD $BDADR  $GDDAT  $BDDAT  $TMP0   $TMP1   $TMP2
                                        DF1             ;ALL NUMBERS ARE TYPED IN OCTAL

                                ;ITEM 3
                                        EM3             ;BACK GROUND CHANGED
                                        DH3             ;ERROR  BCKGRD  EXPECT  RCV'D
                                                        ;PC     ADDRESS DATA    DATA    COL     ROW     PRECHARGE
                                        DT1             ;$HLTAD $BDADR  $GDDAT  $BDDAT  $TMP0   $TMP1   $TMP2
                                        DF1             ;ALL NUMBERS ARE TYPED IN OCTAL
                                ;*******************************************************************************
```

002674 007614
002676 007707
002700 010270
002702 010310
002704 007633
002706 010016
002710 010270
002712 010310
002714 007661
002716 010143
002720 010270
002722 010310

```
                                        ;*****************************************************************************
                                        ;THIS IS WHERE THE TEST INSTRUCTIONS ARE STORED
002724  005010                          INSTR1: CLR     (R0)
002726  077102                          INSTR2: SOB     R1,INSTR1
002730  000207                          INSTR3: RTS     PC
                                        ;END OF TEST INSTRUCTIONS
002732  005010                          INSIDE: CLR     (R0)
002734  077102                                  SOB     R1,INSIDE
002736  000207                                  RTS     PC

002740  000000  177777  177777  OUTSID: .WORD   0,-1,-1
```

```
                                ;*******************************************************************************
                                ;
                                ;
                                ;
                                ;
                                ;THIS IS THE STARTING POINT IF LIMITS ARE TO BE SELECTED
                                ;ALL TYPING SHOULD BE DONE ONLY AFTER SEEING SECTION
                                ;4.3.1 AT THE BEGINNING OF THIS DOCUMENT
                                ;
002746  010706                  BEGIN1: MOV     PC,     SP      ;SET STACK POINTER AND
002750  062706  176130                  ADD     #1100-.,SP      ;MAKE IT POSITION INDEPENDANT
002754  005067  177672                  CLR     PASFLG          ;PASFLG IS COUNTER FOR WHEN
                                                                ;BOTH PICKS & DROPS
002760  004767  004254                  JSR     PC,     TYPE
002764  002647                          $CRLF
002766  012767  000200  176326          MOV     #BIT07,$KT11    ;USE MEM MANG. IF THERE
002774  004767  176300                  JSR     PC,     $SIZE   ;SIZE MEM
003000  004767  004234                  JSR     PC,     TYPE    ;TYPE "LAST ADDRESS IN MEMORY"
003004  010771                          MEMEND                  ;
003006  005767  176310                  TST     $KT11           ;IS MEMORY MANAGEMENT THERE?
003012  100006                          BPL     2$              ;BRANCH IF NO MEM MANAGEMENT
003014  016777  176746  177644          MOV     $LSTBK, @PAGE   ;PAGE CONTAINS KIPAR6 OR KIPAR5
003022  012777  000001  175052          MOV     #1,     @SR0    ;TURN ON MEM. MANAGEMENT
003030  016746  176730          2$:     MOV     $LSTADR,-(SP)   ;GET READY TO TYPE ADDRESS IN OCTAL
003034  004067  004260                  JSR     R0,$B2OCT
003040     006                          .BYTE   6
003041     003                          .BYTE   3               ;PHYSICAL ADDRESS IF MEM MANG. THERE
003042  005767  176254                  TST     $KT11           ;
003046  100002                          BPL     3$              ;
003050  005077  175026                  CLR     @SR0            ;
003054  004767  004160          3$:     JSR     PC,     TYPE    ;
003060  002647                          $CRLF
003062  004767  004152                  JSR     PC,     TYPE    ;GO TO TYPE "TYPE STARTING
                                                                ;BANK NUMBER AND CR
003066  010534                  1$:     STARTB                  ;REFER TO SECTION 4.3.1 BEFORE CHOOSING"
003070  012767  010653  177770          MOV     #START2,1$
003076  004767  176672                  JSR     PC,     .RECD   ;GET STARTING BANK NUMBER
003102  000000                  STBANK: .WORD   0               ;AND LEAVE IT HERE
003104  004767  004130                  JSR     PC,     TYPE    ;GO TO TYPE "TYPE NUMBER OF 4K
003110  010674                  1$:     NUMBER                  ;BANKS TO BE TESTED AND CR"
003112  012767  010753  177770          MOV     #NUMB2, 1$      ;
003120  004767  176650                  JSR     PC,     .RECD   ;GET NUMBER OF BANKS
003124  000000                  BANKS:  .WORD   0               ;AND LEAVE IT HERE
003126  005067  177522                  CLR     SELF
003132  012767  024042  002166          MOV     #BEGIN2+20000, BEG2     ;THESE FOUR LINES
003140  012767  007736  002162          MOV     #7736, LENGTH   ;TO SET UP FOR
003146  012767  000100  002156          MOV     #FIRST, FROM    ;NO LIMIT CHOOSING ON RESTART
003154  012767  020100  002152          MOV     #FIRST+20000, TO
                                ;
                                ;
                                ;THE FOLLOWING INSTRUCTIONS IN THIS SECTION IS FOR
                                ;RELOCATION IF STARTING BANK CHOSEN IS 0
                                ;IF RELOCATION IS ALREADY DONE BY OPERATOR THEN NO FURTHER
                                ;RELOCATION IS DONE
                                ;
                                ;
```

```
  958  003162  016704  176600                MOV     $LSTBK,R4
  959  003166  162704  000140                SUB     #140,R4
  960  003172  022704  001400                CMP     #1400,R4
  961  003176  101002                        BHI     1$
  962  003200  012704  001400                MOV     #1400,R4
  963  003204  000241              1$:       CLC
  964  003206  006104                        ROL     R4
  965  003210  006104                        ROL     R4
  966  003212  006104                        ROL     R4
  967  003214  006104                        ROL     R4
  968  003216  006104                        ROL     R4
  969  003220  006104                        ROL     R4
  970  003222  010402                        MOV     R4,R2
  971  003224  005767  177442                TST     LODER
  972  003230  001027                        BNE     5$                      ;BRANCH IF SOMETHING IN LODER
  973                                                                        ;THAT IS LOADER ALREADY PULLED IN
  974  003232  010403                        MOV     R4,R3
  975  003234  062703  020000                ADD     #20000,R3
  976  003240  010367  177426                MOV     R3,LODER
  977  003244  010705                        MOV     PC,R5   ;TO
  978  003246  062705  006244                ADD     #LAST+2-.,R5
  979  003252  012700  003132                MOV     #17774-LAST/2,R0;LENGTH
  980  003256  014325              2$:       MOV     -(R3),(R5)+
  981  003260  077002                        SOB     R0,2$
  982  003262  004767  003752                JSR     PC,TYPE
  983  003266  011402                        RELOD
  984  003270  012746  000324                MOV     #LOADER,-(SP)
  985  003274  004067  004020                JSR     R0,$B20CT
  986  003300      006                       .BYTE   6
  987  003301      002                       .BYTE   2
  988  003302  004767  003732                JSR     PC,TYPE
  989  003306  011434                        RELOD2
  990  003310  005767  177566        5$:     TST     STBANK
  991  003314  001056                        BNE     4$
  992  003316  010746                        MOV     PC,-(SP)
  993  003320  042726  017777                BIC     #17777,(SP)+
  994  003324  001052                        BNE     4$
  995  003326  010406                        MOV     R4,SP
  996  003330  062706  001100                ADD     #1100,SP
  997  003334  062704  000100                ADD     #FIRST,R4
  998  003340  010467  177324                MOV     R4,TO1
  999  003344  016700  001760                MOV     LENGTH,R0
 1000  003350  016701  001756                MOV     FROM,R1
 1001  003354  012124              3$:       MOV     (R1)+,(R4)+
 1002  003356  077002                        SOB     R0,3$
 1003  003360  062702  004042                ADD     #BEGIN2,R2
 1004  003364  004767  003650                JSR     PC,TYPE
 1005  003370  011264                        PROTO
 1006  003372  016746  177272                MOV     TO1,-(SP)
 1007  003376  062716  000104                ADD     #104,(SP)
 1008  003402  004067  003712                JSR     R0,$B20CT
 1009  003406      006                       .BYTE   6
 1010  003407      002                       .BYTE   2
 1011  003410  004767  003624                JSR     PC,TYPE
 1012  003414  002647                        $CRLF
 1013  003416  004767  003616                JSR     PC,TYPE
```

```
1014  003422  011332                        PROBAK
1015  003424  016746  177240                MOV     T01,-(SP)
1016  003430  062716  000200                ADD     #RESTOR-100,(SP)
1017  003434  004067  003660                JSR     R0,$B20CT
1018  003440     006                        .BYTE   6
1019  003441     002                        .BYTE   2
1020  003442  004767  003572                JSR     PC,TYPE
1021  003446  002647                        $CRLF
1022  003450  000112                        JMP     (R2)
1023  003452  000573              4$:       BR      BEGIN2
1024
1025                                        ;THIS IS THE STARTING POINT IF NO LIMITS ARE TO
1026                                        ;BE SET AND ALL OF EXISTING MEMORY IS TO BE TESTED
1027
1028
1029  003454  010706              BEGN11:   MOV     PC,     SP      ;SETUP STACK POINTER AND
1030  003456  062706  175422                ADD     #1100-.,SP      ;MAKE IT POSITION INDEPENDANT
1031  003462  005067  177164                CLR     PASFLG          ;PASFLG IS COUNTER WHEN BOTH PICKS
1032                                                                ;AND DROPS ARE USED
1033  003466  012767  000200  175626        MOV     #BIT07, $KT11   ;BIT 7 SAYS SEGMENTATION IS THERE
1034  003474  004767  175600                JSR     PC,     $SIZE   ;SIZE MEMORY
1035  003500  004767  003534                JSR     PC,     TYPE    ;
1036  003504  010771                        MEMEND                  ;
1037  003506  005767  175610                TST     $KT11           ;
1038  003512  100006                        BPL     2$              ;
1039  003514  016777  176246  177144        MOV     $LSTBK, @PAGE   ;
1040  003522  012777  000001  174352        MOV     #1,     @SR0    ;
1041  003530  016746  176230        2$:     MOV     $LSTADR,-(SP)   ;
1042  003534  004067  003560                JSR     R0,$B20CT
1043  003540     006                        .BYTE   6
1044  003541     003                        .BYTE   3
1045  003542  005767  175554                TST     $KT11           ;
1046  003546  100002                        BPL     3$              ;
1047  003550  005077  174326                CLR     @SR0            ;
1048  003554  004767  003460        3$:     JSR     PC,     TYPE    ;
1049  003560  002647                        $CRLF
1050  003562  012767  000001  177312        MOV     #1,     STBANK  ;STARTING BANK # = 1 LOC = 20000
1051  003570  016767  176172  177066        MOV     $LSTBK, LSTBK   ;LAST BANK IS STORED IN LSTBK
1052  003576  062767  000040  177060        ADD     #40,LSTBK       ;
1053  003604  006067  177054                ROR     LSTBK           ;DIVIDE LSTBK BY 200
1054  003610  006067  177050                ROR     LSTBK           ;TO GIVE NUMBER OF
1055  003614  006067  177044                ROR     LSTBK           ;BANKS PRESENT IN
1056  003620  006067  177040                ROR     LSTBK           ;EXISTING MEMORY
1057  003624  006067  177034                ROR     LSTBK           ;
1058  003630  006067  177030                ROR     LSTBK           ;
1059  003634  006067  177024                ROR     LSTBK           ;
1060  003640  005367  177020                DEC     LSTBK           ;NOW DECREASE ONE FOR BANKS
1061  003644  016767  177014  177252        MOV     LSTBK,  BANKS   ;TO BE TESTED AND STORE
1062  003652  012767  177777  176774        MOV     #-1,    SELF    ;1 = RELOCATION REQUIRED
1063  003660  012767  024042  001440        MOV     #BEGIN2+20000, BEG2
1064  003666  012767  007736  001434        MOV     #7736,LENGTH
1065  003674  012767  000100  001430        MOV     #FIRST, FROM
1066  003702  012767  020100  001424        MOV     #FIRST+20000, TO
1067  003710  005067  176742                CLR     REDONE          ;RELOCATION EXIT COUNTER
1068  003714  005767  176752                TST     LODER
1069  003720  001050                        BNE     BEGIN2          ;BRANCH IF SOMETHING IN LODER
```

```
1070                                                    ;THAT IS LOADER ALREADY PULLED IN
1071  003722  016704  176040                  MOV     $LSTBK,R4
1072  003726  162704  000140                  SUB     #140,R4
1073  003732  022704  001400                  CMP     #1400,R4
1074  003736  101002                          BHI     1$
1075  003740  012704  001400                  MOV     #1400,R4
1076  003744  000241              1$:         CLC
1077  003746  006104                          ROL     R4
1078  003750  006104                          ROL     R4
1079  003752  006104                          ROL     R4
1080  003754  006104                          ROL     R4
1081  003756  006104                          ROL     R4
1082  003760  006104                          ROL     R4
1083  003762  010402                          MOV     R4,R2
1084  003764  010403                          MOV     R4,R3
1085  003766  062703  020000                  ADD     #20000,R3
1086  003772  010367  176674                  MOV     R3,LODER
1087  003776  010705                          MOV     PC,R5   ;TO
1088  004000  062705  005512                  ADD     #LAST+2-.,R5
1089  004004  012700  003132                  MOV     #17774-LAST/2,R0;LENGTH
1090  004010  014325              4$:         MOV     -(R3),(R5)+
1091  004012  077002                          SOB     R0,4$
1092  004014  004767  003220                  JSR     PC,TYPE
1093  004020  011402                          RELOD
1094  004022  012746  000324                  MOV     #LOADER,-(SP)
1095  004026  004067  003266                  JSR     R0,$B20CT
1096  004032     006                          .BYTE   6
1097  004033     002                          .BYTE   2
1098  004034  004767  003200                  JSR     PC,TYPE
1099  004040  011434                          RELOD2
1100
1101                                    ;THE CONSOLE SWITCHES ARE TO BE SET.  LOOK AT SECTION
1102                                    ;4.1 AT THE FRONT OF THIS DOC
1103
1104  004042  004767  003172          BEGIN2: JSR     PC,     TYPE
1105  004046  002647                          $CRLF
1106  004050  004767  003164                  JSR     PC,     TYPE    ;GO TO TYPE "SET SWITHCHES
1107  004054  010350              1$:         OPMODE                  ;FOR OPERATNG MODE IE.  PICKUPS
1108  004056  012767  010515  177770          MOV     #OPMOD2,1$
1109  004064  000000                          HALT                    ;OR DROPS OR BOTH THEN HIT CONTINUE"
1110  004066  032737  040000  177570          BIT     #BIT14,@#SWR    ;PARITY CHECKING BIT SET?
1111  004074  001402                          BEQ     BEGIN3          ;IF NOT BRANCH
1112  004076  004767  176074                  JSR     PC,MAMF         ;GO TO ENABLE PARITY
1113
1114                                    ;CONVERT THE TYPED LIMITS INTO HIGH AND LOW LIMITS
1115                                    ;IF SELF STARTING USED THEN "STBANK" AND "BANKS"
1116                                    ;HAVE ALREADY BEEN FILLED FOR USE HERE
1117
1118  004102  016700  176774          BEGIN3: MOV     STBANK, R0      ;MAKE WORKING LOCATION FOR LOW LIMIT
1119  004106  005200                          INC     R0
1120  004110  012703  177600                  MOV     #-200,  R3      ;R3 WILL HAVE PAGE ADDRESS
1121  004114  016701  177004                  MOV     BANKS,  R1      ;MAKE WORKING LOCATION FOR HIGH LIMIT
1122  004120  012702  160000                  MOV     #-20000,R2      ;R2 WILL HAVE LOW LIMIT
1123  004124  062702  020000          1$:     ADD     #20000, R2      ;GETTING LOW LIMIT
1124  004130  062703  000200                  ADD     #200,   R3      ;GETTING LOW LIMIT PAGE ADDRESS
1125  004134  077005                          SOB     R0,     1$      ;REPEAT R0 TIMES
```

```
1126  004136  010367  176516                    MOV     R3,     LOBASE  ;STORE LOW LIMIT PAGE ADDRESS
1127  004142  010267  176440                    MOV     R2,     LOLMT   ;SET LOW LIMIT
1128  004146  010267  176440                    MOV     R2,     MOD     ;SET MODULE NUMBER
1129  004152  162703  000200                    SUB     #200,   R3      ;GETTING READY FOR HIGH LIMIT PAGE
1130  004156  062702  020000          BEGIN4:   ADD     #20000, R2      ;GETTING HIGH LIMIT
1131  004162  062703  000200                    ADD     #200,   R3      ;GETTING HIGH LIMIT PAGE ADDRESS
1132  004166  077105                    SOB     R1,     BEGIN4  ;REPEAT FOF BANK NUMBER
1133  004170  005302                    DEC     R2              ;GETTING HIGH LIMIT WITHIN BANK
1134  004172  010267  176412                    MOV     R2,     HILMT   ;STORE HIGH LIMIT
1135  004176  010367  176460                    MOV     R3,     HIBASE  ;STORE HIGH LIMIT PAGE ADDRESS
1136  004202  022767  000006  176672            CMP     #6,     STBANK  ;DOES STARTING BANK NEED SEGMENTATION?
1137  004210  103401                    BLO     1$              ;BRANCH IF YES
1138  004212  000407                    BR      2$
1139  004214  162767  000200  176436  1$:       SUB     #200,   LOBASE  ;SEGMENTATION REQUIREG SO
1140  004222  005167  176424                    COM     PASFLG          ;
1141  004226  000167  000716                    JMP     SEGMT1          ;SUBTRACT TO ENABLE REPEATED ADDITION
1142  004232  022703  001600          2$:       CMP     #1600,  R3      ;IS SEGMENTATION REQUIRED FOR HIGH?
1143  004236  101003                    BHI     BEGIN5          ;IF NO THEN BRANCH
1144  004240  012767  157776  176342            MOV     #157776,HILMT   ;IF YES HIGH LIMIT = END OF 32K
1145
1146                                    ;SETUP ALL TAGS TO THEIR APPROPIATE VALUES
1147
1148  004246  005067  176342          BEGIN5:   CLR     COL             ;MAKE COLUMN ZERO
1149  004252  005067  176340                    CLR     ROW             ;MAKE ROW ZERO
1150  004256  005067  176336                    CLR     PCHRG           ;MAKE PRECHARGE ZERO
1151  004262  012767  023420  176332            MOV     #10000.,        CYCNT   ;SETUP CYCLE COUNT
1152  004270  012767  177777  176326            MOV     #-1,    TSTPAT  ;SETUP TEST PATTERN
1153  004276  012767  177777  176322            MOV     #-1,    BCKGRD  ;SETUP BACKGROUND
1154  004304  012767  000400  176320            MOV     #BIT08, MSKBIT  ;SETUP MASK
1155
1156
1157
1158
1159  004312  032737  010000  177570  BEGIN6:   BIT     #SW12,@#SWR     ;IS SWITCH 12 SET?
1160  004320  001013                    BNE     1$              ;IF SET BRANCH TO CHECK SW11
1161  004322  005167  176324                    COM     PASFLG          ;COMPLEMENT PASS FLAG
1162                                                                    ;IS BOTH PICKUPS AND DROPS DONE
1163  004326  001037                    BNE     3$              ;IF YES BRANCH
1164  004330  005067  176272                    CLR     BCKGRD          ;IF NOT SETUP TO
1165  004334  005067  176264                    CLR     TSTPAT          ;CATCH
1166  004340  012767  177777  176372            MOV     #-1,    OUTSID  ;BIT PICKS
1167  004346  000437                    BR      4$
1168  004350  005067  176276          1$:       CLR     PASFLG  ;
1169  004354  032737  010000  177570            BIT     #SW12,@#SWR     ;TEST SW12
1170  004362  001410                    BEQ     2$              ;IF NOT SET BRANCH TO CATCH DROPS
1171  004364  005067  176236                    CLR     BCKGRD          ;IF NOT SET TO CATCH PICKUPS
1172  004370  005067  176230                    CLR     TSTPAT          ;
1173  004374  012767  177777  176336            MOV     #-1,OUTSID      ;
1174  004402  000421                    BR      4$              ;
1175  004404  012767  177777  176212  2$:       MOV     #-1,    TSTPAT  ;
1176  004412  012767  177777  176206            MOV     #-1,    BCKGRD  ;
1177  004420  005067  176314                    CLR     OUTSID          ;
1178  004424  000410                    BR      4$
1179  004426  012767  177777  176170  3$:       MOV     #-1,    TSTPAT
1180  004434  012767  177777  176164            MOV     #-1,    BCKGRD
1181  004442  005067  176272                    CLR     OUTSID
```

```
1182  004446  005067  176066          4$:     CLR     $PASS           ;CLEAR PASS COUNT
1183
1184
1185
1186
1187                                  ;MAKE THE TEST ADDRESS AND THE INSTRUCTION ADDRESS
1188
1189
1190
1191
1192  004452  016746  176134          LOOP1:  MOV     MOD,    -(SP)   ;PICKUP MODULE NUMBER
1193  004456  042716  017777                  BIC     #17777,(SP)     ;STRIP AWAY ANY JUNK
1194  004462  016746  176126                  MOV     COL,-(SP)       ;PICKUP COLUMN NUMBER
1195  004466  042716  160777                  BIC     #^C17000,(SP)   ;STRIP AWAY JUNK
1196  004472  032767  020000  176114          BIT     #BIT13,COL      ;CHECK C4=1
1197  004500  001402                          BEQ     1$              ;BR IF NO
1198  004502  052716  000002                  BIS     #BIT01,(SP)     ;PUT C4 AT PROPER PLACE IN BUS ADDRESS
1199  004506  016746  176106          1$:     MOV     PCHRG,-(SP)     ;PICK UP PRECHARGE
1200  004512  042716  177763                  BIC     #^C14,(SP)      ;STRIP JUNK
1201  004516  016746  176074                  MOV     ROW,-(SP)       ;GET THE ROW NUMBER
1202  004522  042716  177017                  BIC     #^C760,(SP)     ;STRIP THE JUNK IF ANY
1203  004526  012600                          MOV     (SP)+,R0        ;FORM THE BUS ADDRESS WITH ROW
1204  004530  052600                          BIS     (SP)+,R0        ;PRECHARGE
1205  004532  052600                          BIS     (SP)+,R0        ;COLUMN
1206  004534  052600                          BIS     (SP)+,R0        ;MODULE
1207  004536  010067  176042                  MOV     R0,TSTADR       ;SAVE AS THE TEST ADDRESS
1208  004542  010067  176062                  MOV     R0,INSTAD       ;AND AS THE INSTRUCTION ADDRESS
1209
1210
1211
1212                                  ;MOVE EITHER  "0,-1,-1,"  OR THE  "CLEAR R0"
1213                                  ;INTO THE LOCATION INSTR1: THIS WILL FINALLY BE THE TEST INSTRUCTION
1214
1215
1216
1217  004546  010700                  LPERR:  MOV     PC,     R0      ;THESE TWO INSTRUCTIONS
1218                                                                  ;GETS THE LOCATION
1219  004550  062700  176154                  ADD     #INSTR1-.,R0    ;OF INSTR1 IN TO R0
1220                                                                  ;EVEN AFTER RELOCATION
1221  004554  013767  177570  000160          MOV     @#SWR,  SWITCH  ;GET SWITCH SETTING
1222                                                                  ;TO CHECK FOR
1223  004562  042767  177577  000152          BIC     #177577,        SWITCH  ;IF SW07=0 DO OUTSIDE
1224
1225                                                                  ;IF SW07=1 DO INSIDE
1226
1227  004570  001404                          BEQ     1$              ;IF SW07=0 THEN BRANCH
1228
1229  004572  010701                          MOV     PC,     R1      ;SW7=1 SO SETUP TO
1230                                                                  ;MOVE INSIDE INTO INSTR1
1231  004574  062701  176136                  ADD     #INSIDE-.,R1    ;MAKE INSIDE POSITION
1232                                                                  ;INDEPENDANT
1233  004600  000403                          BR      2$
1234
1235  004602  010701                  1$:     MOV     PC,     R1      ;SW07=0 SO SETUP TO
1236                                                                  ;MOVE OUTSIDE INTO INSTR1
1237  004604  062701  176134                  ADD     #OUTSID-.,R1    ;MAKE OUTSIDE POSITION
```

```
                                                                ;INDEPENDANT
004610  012120                  2$:     MOV     (R1)+,  (R0)+   ;THESE THREE INSTRUCTIONS

004612  012120                          MOV     (R1)+,  (R0)+   ;MOVES THE APPROPIATE

004614  012120                          MOV     (R1)+,  (R0)+   ;GROUP INTO INSTR1

004616  105067  175732                  CLRB    $ERFLG          ;CLEAR ERROR FLAG




                                ;FILL 4K WITH BACKGROUND PATTERN


004622  016700  176000          FILL4K:MOV     BCKGRD, R0      ;GETTING READY TO MOVE
                                                                ;BACKGROUND INTO 4K
004626  016701  175760                  MOV     MOD,    R1      ;STARTING FROM MODULE
                                                                ;NUMBER
004632  012702  010000                  MOV     #10000, R2      ;10000=4K
004636  010021                  1$:     MOV     R0,     (R1)+   ;FILL BACKGROUND
004640  077202                          SOB     R2,     1$      ;REPEAT 4K TIMES

                                        ;THIS MOVES CONTENTS OF TEST PATTERN (TSTPAT)
                                        ;HERE = -1 INTO MOD AND ALL LOCATIONS GOT
                                        ;BY INCREASING COLUMN BY ONE TILL ONE 4K
                                        ;IS COMPLETE
004642  016700  175736          LDPAT:  MOV     TSTADR,R0       ;MOVE TEST ADDRESS
004646  010001                          MOV     R0,R1           ;DUPLICATE IT IN R1 TO BE REINSTATED AFTER ADDING COLUMN
004650  042701  017003                  BIC     #^C160774,R1    ;DON'T KEEP THE COL
004654  016702  175734                  MOV     COL,R2          ;PICK UP THE COL
004660  012703  020002                  MOV     #BIT13!BIT01,R3 ;GET READY TO TRANSFER BIT 13 TO BIT 1
004664  016704  175734                  MOV     TSTPAT,R4       ;MOVE TEST PATTERN=-1
004670  062702  001000          1$:     ADD     #1000,R2        ;ADD ONE TO COLUMN
004674  042702  040000                  BIC     #BIT14,R2       ;IF IT WAS ALREADY 37 THEN ADDING ONE WILL GIVE BIT 14
004700  010205                          MOV     R2,R5           ;FORM AN ADDRESS
004702  032705  020000                  BIT     #BIT13,R5       ;POSITION C4 IF NEEDED
004706  001401                          BEQ     2$              ;IF BIT 13 IS ZERO THEN BRANCH
004710  074305                          XOR     R3,R5           ;IF BIT 13=1 THEN TRANSFER TO BIT 1
004712  050105                  2$:     BIS     R1,R5           ;REINSTATE OTHER BITS AFTER ADDING TO COLUMN
004714  010415                          MOV     R4,(R5)         ;STORE THE PATTERN
004716  020005                          CMP     R0,R5           ;TSTADR?
004720  001363                          BNE     1$              ;BR IF NO




                                ;THE FOLLOWING JSR LOADS INSRTUCTION (INSTR1) INTO
                                ;PROPER PLACE


004722  004767  000732          LOOP2:  JSR     PC,     LDINT

004726  000431                          BR      NEXT
```

```
1294
1295  004730  011067  175700                  MOV     (R0),   SV400
1296
1297  004734  016701  175662                  MOV     CYCNT,  R1      ;DO TEST SEQ. THIS
1298                                                                    ;MANY TIMES
1299  004740  005727                          TST     (PC)+
1300  004742  000000                  SWITCH: .WORD   0
1301  004744  001406                          BEQ     1$
1302  004746  016746  175656                  MOV     INSTAD,-(SP)    ;SETUP THE STARTING ADDRESS
1303  004752  011637  177570                  MOV     (SP),@#DISPLAY  ;DISPLAY THE TEST ADDRESS
1304  004756  004736                          JSR     PC,@(SP)+       ;GO TO THE TEST SEQ.
1305  004760  000406                          BR      3$
1306  004762  016702  175642          1$:     MOV     INSTAD,R2
1307  004766  010237  177570                  MOV     R2,@#DISPLAY
1308  004772  011210                  2$:     MOV     (R2),(R0)
1309  004774  077102                          SOB     R1,2$
1310  004776  016710  175632          3$:     MOV     SV400,(R0)
1311
1312
1313                                  ;CHECK THE 1K BLOCK
1314
1315
1316  005002  004767  001074                  JSR     PC,CHECK        ;GO CHECK THE 1K BLOCK
1317  005006  000401                          BR      NEXT            ;RETURN HERE ON ERROR
1318  005010  000744                          BR      LOOP2
1319  005012  032737  000001  177570  NEXT:   BIT     #BIT00,@#SWR    ;LOCK ON THIS PRECHARGE?
1320  005020  001007                          BNE     1$              ;BR IF YES
1321  005022  062767  000004  175570          ADD     #4,PCHRG        ;GO TO NEXT PRECHARGE
1322  005030  042767  177763  175562          BIC     #^C14,PCHRG     ;GET RID OF TRASH
1323  005036  001205                          BNE     LOOP1           ;LOOP IF MORE TO DO
1324  005040  032737  000002  17757C  1$:     BIT     #BIT01,@#SWR    ;LOCK ON THIS MODULE?
1325  005046  001133                          BNE     SEOP            ;BR IF YES
1326  005050  062767  020000  175534          ADD     #BIT13,MOD      ;INCREMENT THE MOD
1327  005056  103407                          BCS     SEGMT           ;IF ADDING BIT13 MAKES 0
1328                                                                  ;THEN BRANCH
1329  005060  026767  175526  175522          CMP     MOD,HILMT       ;DON'T LET IT GET TO BIG
1330  005066  103401                          BLO     LOOP12
1331  005070  000402                          BR      SEGMT           ;GO TO CHECK IF SEGMENTATION IS REQUIRED
1332  005072  000167  177354          LOOP12: JMP     LOOP1           ;RETURN
1333                                  ;IS SEGMENTATION REQUIRED?  IF SO TURN ON MEMORY
1334                                  ;MANAGEMENT AND SETUP ALL REQUIRED REGISTERS
1335                                  ;ONLY KERNEL MODE INSTRUCTION SPACE IS USED
1336
1337  005076  022767  001600  175556  SEGMT:  CMP     #1600,  HIBASE  ;SEGMENTATION REQUIRED FOR HIGH LIMIT?
1338  005104  101114                          BHI     SEOP            ;IF NO THEN BRANCH
1339  005106  022767  001600  175544          CMP     #1600,  LOBASE  ;IF YES THEN TEST IF LOW LIMIT REQ SEGMENTATION?
1340  005114  101415                          BLOS    SEGMT1          ;IF LOW LIMIT DOES REQ SEGMENTATION BRANCH
1341  005116  012767  001600  175534          MOV     #1600,  LOBASE  ;IF NO MAKE LOW LIMIT END OF 32K
1342  005124  012767  140000  175454  SEGO:   MOV     #140000,LOLMT   ;LOLMT LOOKS AT PAGE PAGE OR 6
1343  005132  012767  140000  175452          MOV     #140000,MOD     ;SAME WITH MODULE NUMBER
1344  005140  012767  157776  175442          MOV     #157776,HILMT   ;HIGH LIMIT LOOKS AT PAGE  END OF 4K
1345  005146  000420                          BR      SEGMT2          ;BRANCH TO ENABLE SEGMENTATION
1346  005150  062767  000200  175502  SEGMT1: ADD     #200,   LOBASE  ;DO NEXT 4K
1347  005156  012767  140000  175422  SEG10:  MOV     #140000,LOLMT   ;MAKE LOW LIMIT AND MODULE
1348  005164  012767  140000  175420          MOV     #140000,MOD     ;0 ON PAGE "PAGE OR 6"
1349  005172  012767  157776  175410          MOV     #157776,HILMT   ;HIGH LIMIT SAME 4K ON PAGE
```

```
1350  005200  026767  175456  175452          CMP     HIBASE, LOBASE  ;IS IT THE END?
1351  005206  103453                          BLO     $EOP            ;IF YES BRANCH
1352  005210  016777  175444  175450  SEGMT2: MOV     LOBASE, @PAGE ;SET LOW LIMIT ON PAGE 4
1353  005216  005077  172666                  CLR     @SR3            ;USE ONLY I SPACE IN KERNAL
1354  005222  012700  077406                  MOV     #77406, R0      ;MOVE INTO R0
1355  005226  010077  172660                  MOV     R0,     @KIPDR0 ;SEGMENT DESCRIPTOR
1356  005232  010077  172656                  MOV     R0,     @KIPDR1 ;REGISTER IS SET FOR
1357  005236  010077  172654                  MOV     R0,     @KIPDR2 ;128 BLOCKS WITH NO
1358  005242  010077  172652                  MOV     R0,     @KIPDR3 ;SYSTEM TRAPS.  THESE
1359  005246  010077  172650                  MOV     R0,     @KIPDR4 ;REGISTERS ARE ALSO
1360  005252  010077  172646                  MOV     R0,     @KIPDR5 ;CALLED PAGE
1361  005256  010077  172644                  MOV     R0,     @KIPDR6 ;DESCRIPTOR REGISTERS
1362  005262  010077  172642                  MOV     R0,     @KIPDR7 ;ONLY PAGE 0 AND 4 IS USED
1363  005266  005077  172640                  CLR     @KIPAR0 ;MAKE PAGE ADDRESS 0=0
1364  005272  012777  000001  172602          MOV     #1,     @SR0    ;ENABLE SEGMEMTATION
1365  005300  032737  010000  177570          BIT     #SW12,  @#SWR   ;TEST SWITCH 12
1366  005306  001003                          BNE     2$              ;BRANCH IF SET
1367  005310  005167  175336                  COM     PASFLG          ;SW12 NOT SETIE. DO BOTH
1368  005314  000402                          BR      3$              ;SO COMPLEMENT AND RESTART
1369  005316  005067  175330          2$:     CLR     PASFLG          ;SW12 SET IE. DO ONE SO RESTART
1370  005322  000167  176764          3$:     JMP     BEGIN6
1371
1372
1373
1374                                  ;*****************************************************************
1375                                  ;END OF PASS ROUTINE
1376                                  ;INCREMENT THE PASS NUMBER
1377                                  ;IF SW10=0 PRINT END PASS ON END OF PROGRAM
1378                                  ;IF THERE IS A MONITOR GO TO IT.
1379                                  ;IF NONE JUMP TO BEGIN3
1380                                  ;IF LIMITS ARE NOT CHOSEN AND ALL OF MEMORY
1381                                  ;IS TO BE TESTED THIS RELOCATES PROGRAM FROM
1382                                  ;FIRST 4K TO SECOND 4K AND SETS UP FOR
1383                                  ;TESTING THE FIRST 4K.  THEN RELOCATES
1384                                  ;TO FIRST 4K TO BE ABLE TO START AGAIN.
1385
1386
1387
1388  005326  024042                  BEG2:   20000+BEGIN2
1389  005330  007736                  LENGTH: 7736
1390  005332  000100                  FROM:   FIRST
1391  005334  020100                  TO:     20000 + FIRST
1392
1393
1394
1395  005336  005767  175312          $EOP:   TST     SELF            ;WAS SELF STARTING USED
1396  005342  001523                          BEQ     $EOP1           ;IF NOT IE. LIMITS WERE CHOSEN THEN BRANCH
1397                                                                  ;HAS ENTRY INTO THIS SECTION BEEN MADE?
1398                                                                  ;IF YES IE. THIS IS SECOND TIME THEN BRANCH
1399  005344  005767  173752                  TST     $KT11           ;
1400  005350  100005                          BPL     5$
1401  005352  005077  172524                  CLR     @SR0            ;
1402  005356  000240                          NOP
1403  005360  000240                          NOP
1404  005362  000240                          NOP
1405  005364  016700  177740          5$:     MOV     LENGTH, R0      ;IF NOT IE. FIRST TIME OR THIRD TIME
```

```
1406  005370  016701  177736                        MOV     FROM,   R1      ;GETTING READY TO RELOCATE
1407  005374  016702  177734                        MOV     TO,     R2      ;GETTING READY TO RELOCATE
1408  005400  012767  000000  175474                MOV     #0,     STBANK  ;FILL IN TYPED INFORMATION
1409  005406  012767  000001  175510                MOV     #1,     BANKS   ;ABOUT STARTING ADDRESSES
1410  005414  005067  175166                        CLR     LOLMT           ;SETUP LIMITS
1411  005420  005067  175166                        CLR     MOD             ;MODULE
1412  005424  012767  017776  175156                MOV     #17776, HILMT   ;AND HIGH
1413  005432  012122                    1$:         MOV     (R1)+,  (R2)+   ;FROM INTO TO
1414  005434  077002                                SOB     R0,     1$      ;REPEAT
1415  005436  005167  175214                        COM     REDONE          ;BEEN THROUGH ONCE?
1416  005442  016705  175210                        MOV     REDONE,R5       ;
1417  005446  001401                                BEQ     2$              ;BRANCH IF SECOND TIME THROUGH
1418  005450  000433                                BR      3$              ;FIRST TIME GO TO 3$
1419  005452  012737  000001  003102        2$:     MOV     #1,     @#STBANK
1420  005460  016737  175200  003124                MOV     LSTBK,  @#BANKS
1421  005466  012706  001100                        MOV     #1100,SP
1422  005472  000240                                NOP
1423  005474  010537  002656                        MOV     R5,@#REDONE
1424  005500  012737  000100  005332                MOV     #FIRST,@#FROM
1425  005506  012737  020100  005334                MOV     #20000+FIRST,@#TO
1426  005514  000240                                NOP
1427  005516  000240                                NOP
1428  005520  000240                                NOP
1429  005522  000240                                NOP
1430  005524  000240                                NOP
1431  005526  000240                                NOP
1432  005530  000137  005612                        JMP     @#SEOP1
1433  005534  000137  004102                4$:     JMP     @#BEGIN3
1434  005540                            3$:
1435  005540  012706  021100                        MOV     #21100,SP
1436  005544  010537  022656                        MOV     R5,@#20000+REDONE
1437  005550  012701  005332                        MOV     #FROM,R1        ;
1438  005554  012702  005334                        MOV     #TO,R2          ;
1439  005560  062701  020000                        ADD     #20000,R1       ;
1440  005564  062702  020000                        ADD     #20000,R2       ;
1441  005570  016703  177536                        MOV     FROM,R3         ;
1442  005574  016711  177534                        MOV     TO,(R1)         ;
1443  005600  010312                                MOV     R3,(R2)         ;
1444  005602  005137  022652                        COM     @#20000+PASFLG
1445  005606  000137  024312                        JMP     @#BEGIN6+20000
1446
1447  005612  005767  175034        $EOP1:  TST     PASFLG          ;
1448  005616  001011                                BNE     $EOP3           ;
1449  005620  005267  174714        $EOP2:  INC     $PASS           ;INCREMENT PASS COUNT
1450  005624  032737  002000  177570          BIT     #SW10,@#SWR     ;TYPE END PASS?
1451  005632  001003                                BNE     $EOP3           ;
1452  005634  004767  001400                        JSR     PC,TYPE
1453  005640  010335                                EPASS
1454  005642  005767  173454        $EOP3:  TST     $KT11
1455  005646  100002                                BPL     1$
1456  005650  005077  172226                        CLR     @$RD
1457  005654  000167  176222        1$:     JMP     BEGIN3
1458
1459
1460
1461
```

```
1462  005660                                LDINT:
1463  005660  010246                             MOV     R2,-(SP)        ;PUSH R2 ON STACK
1464  005662  010346                             MOV     R3,-(SP)        ;PUSH R3 ON STACK
1465  005664  016700  174740              1$:    MOV     INSTAD,R0       ;GET THE INSTRUCTION ADDRESS
1466  005670  042700  177017                     BIC     #^C760,R0       ;KEEP ROW
1467  005674  020067  174716                     CMP     R0,ROW          ;AT THE TEST ROW?
1468  005700  001412                             BEQ     4$              ;BR IF YES
1469  005702  016746  174722                     MOV     INSTAD,-(SP)
1470  005706  016776  174714  000000             MOV     BCKGRD,@(SP)    ;RESTORE BACK GROUND
1471  005714  062716  000002                     ADD     #2,(SP)
1472  005720  016736  174714                     MOV     SVIN2,@(SP)+
1473  005724  000403                             BR      5$
1474  005726  016777  174672  174674  4$:       MOV     TSTPAT,@INSTAD
1475  005734  016777  174664  174642  5$:       MOV     TSTPAT,@TSTADR  ;STORE THE TEST PATTERN
1476  005742  016700  174662                     MOV     INSTAD,R0       ;GET THE INSTRUCTIONS ADDRESS
1477  005746  004767  000600              2$:    JSR     PC,FORMAD       ;GO FORM NEXT INSTUCTION ADDRESS
1478  005752  010003                             MOV     R0,R3           ;CHECK FOR OUT OF RANGE
1479  005754  016702  174624                     MOV     TSTADR,R2       ;PICKUP THE TSTADR FOR CHECKING
1480  005760  020302                             CMP     R3,R2           ;IS INSTR. ADDRESS = TEST ADDRESS?
1481  005762  001443                             BEQ     3$              ;BR IF YES
1482  005764  062703  000002                     ADD     #2,R3           ;ADDRESS WHERE INSTR2 GOES
1483  005770  020302                             CMP     R3,R2           ;IS IT THE SAME AS TSTADR
1484  005772  001437                             BEQ     3$              ;BR IF YES
1485  005774  062703  000002                     ADD     #2,R3           ;INSTR3'S ADDRESS
1486  006000  020302                             CMP     R3,R2           ;SAME AS TSTADR?
1487  006002  001433                             BEQ     3$              ;BR IF YES
1488  006004  020367  174600                     CMP     R3,HILMT        ;WILL IT GO OFF THE END?
1489  006010  101356                             BHI     2$              ;BR IF YES
1490  006012  010067  174612                     MOV     R0,INSTAD       ;SAVE NEW ADDRESS
1491  006016  011067  174614                     MOV     (R0),SVIN1      ;SAVE WHATS AT THE 1ST TWO
1492  006022  016067  000002  174610             MOV     2(R0),SVIN2     ;TEST INSTRUCTIONS ADDRESSES
1493  006030  016710  174670                     MOV     INSTR1,(R0)     ;LOAD THE INSTRUCTIONS
1494  006034  016760  174666  000002             MOV     INSTR2,2(R0)
1495  006042  016760  174662  000004             MOV     INSTR3,4(R0)
1496  006050  016746  174556                     MOV     MSKBIT,-(SP)    ;PICKUP THE MASK BIT
1497  006054  074016                             XOR     R0,(SP)         ;HALF ADD TO THE ADDRESS
1498  006056  012600                             MOV     (SP)+,R0        ;GET THE ADDRESS TO BE WRITTEN INTO
1499  006060  020002                             CMP     R0,R2           ;IS IT THE SAME AS THE TEST ADDRESS?
1500  006062  001700                             BEQ     1$              ;BR IF YES
1501  006064  062766  000002  000004             ADD     #2,4(SP)        ;ADJUST THE RETURN ADDRESS
1502  006072                              3$:
1503  006072  012603                             MOV     (SP)+,R3        ;POP STACK INTO R3
1504  006074  012602                             MOV     (SP)+,R2        ;POP STACK INTO R2
1505  006076  000207                             RTS     PC              ;GO BACK
1506
1507
1508
1509  006100  000000                      TCOL:  0
1510
1511  006102  010246                      CHECK: MOV     R2,-(SP)        ;PUSH R2 ON STACK
1512  006104  010346                             MOV     R3,-(SP)        ;PUSH R3 ON STACK
1513  006106  010446                             MOV     R4,-(SP)        ;PUSH R4 ON STACK
1514  006110  010546                             MOV     R5,-(SP)        ;PUSH R5 ON STACK
1515  006112  016701  174466                     MOV     TSTADR,R1
1516  006116  016702  174506                     MOV     INSTAD,R2       ;ADDRESS OF THE TEST INSTR1
1517  006122  010200                             MOV     R2,R0
```

```
1518  006124  010203                          MOV     R2,R3
1519  006126  062703  000002                  ADD     #2,R3                   ;ADDRESS OF THE TEST INSTR2
1520  006132  010004                          MOV     R0,R4                   ;USED TO KEEP TRACK OF THE TEST ROW
1521  006134  042704  177017                  BIC     #^C760,R4
1522  006140  010067  177734                  MOV     R0,TCOL                 ;USED TO KEEP TRACK OF THE TEST COL
1523  006144  042767  160775  177726          BIC     #^C17002,TCOL
1524  006152  010746                          MOV     PC,     -(SP)           ;SETUP THE ESCAPE ON ERROR ADDRESS
1525  006154  062716  000352                  ADD     #CHK1-.,(SP)            ;MAKE IT POSITION INDEPENDANT
1526  006160  012667  174456                  MOV     (SP)+,ESCAPE
1527  006164  000407                          BR      1$
1528
1529  006166  042700  017762          8$:     BIC     #17762,R0               ;FORM ADDRESS
1530  006172  050400                          BIS     R4,R0                   ;GET THE ROW
1531  006174  056700  177700                  BIS     TCOL,R0                 ;AND THE COLUMN
1532  006200  020002                          CMP     R0,R2                   ;IS THIS THE INSTRUCTION ADDRESS?
1533  006202  001551                          BEQ     CHK1                    ;BR IF YES
1534
1535  006204  011005                  1$:     MOV     (R0),R5                 ;PICK UP THE DATA
1536  006206  020001                          CMP     R0,R1                   ;AT TSTADR?
1537  006210  001016                          BNE     2$                      ;BR IF NO
1538
1539  006212  020567  174406                  CMP     R5,TSTPAT               ;DID THE TSTPAT UNDER TEST CHANGE?
1540  006216  001515                          BEQ     7$                      ;BR IF NO
1541  006220  010567  174350                  MOV     R5,$BDDAT
1542  006224  016767  174374  174340          MOV     TSTPAT,$GDDAT
1543  006232  010067  174332                  MOV     R0,$BDADR
1544  006236  004767  000352                  JSR     PC,$HLT
1545  006242  000001                          1
1546  006244  000502                          BR      7$
1547
1548  006246  020002                  2$:     CMP     R0,R2                   ;AT INSTAD
1549  006250  001020                          BNE     4$                      ;BR IF NO
1550  006252  020567  174446                  CMP     R5,INSTR1               ;DID INSTR1 UNDER TEST CHANGE?
1551  006256  001412                          BEQ     3$                      ;BR IF NO
1552  006260  010567  174310                  MOV     R5,$BDDAT
1553  006264  016767  174434  174300          MOV     INSTR1,$GDDAT
1554  006272  010067  174272                  MOV     R0,$BDADR
1555  006276  004767  000312                  JSR     PC,$HLT
1556  006302  000002                          2
1557  006304  016710  174326          3$:     MOV     SVIN1,(R0)              ;RESTORE THIS LOCATION
1558  006310  000460                          BR      7$
1559
1560  006312  020003                  4$:     CMP     R0,R3                   ;AT THE 2ND INSTRUCTION
1561  006314  001020                          BNE     6$                      ;BR IF NO
1562  006316  020567  174404                  CMP     R5,INSTR2               ;DID IT CHANGE?
1563  006322  001412                          BEQ     5$                      ;BR IF NO
1564  006324  010567  174244                  MOV     R5,$BDDAT
1565  006330  016767  174372  174234          MOV     INSTR2,$GDDAT
1566  006336  010067  174226                  MOV     R0,$BDADR
1567  006342  004767  000246                  JSR     PC,$HLT
1568  006346  000002                          2
1569  006350  016710  174264          5$:     MOV     SVIN2,(R0)              ;RESTORE THIS INSTRUCTION
1570  006354  000436                          BR      7$
1571  006356  026704  174234          6$:     CMP     ROW,R4                  ;ON THE SAME ROW AS TSTADR?
1572  006362  001016                          BNE     10$
1573  006364  026705  174234                  CMP     TSTPAT,R5               ;DID THE PATTERN CHANGE?
```

```
1574  006370  001430                        BEQ     7$
1575  006372  010567  174176                MOV     R5,$BDDAT
1576  006376  016767  174222  174166        MOV     TSTPAT,$GDDAT
1577  006404  010067  174160                MOV     R0,$BDADR
1578  006410  004767  000200                JSR     PC,$HLT
1579  006414  000001                        1
1580  006416  000415                        BR      7$
1581  006420  020567  174202          10$:  CMP     R5,BCKGRD               ;DID BACK GROUND CHANGE
1582  006424  001412                        BEQ     7$                      ;BR IF NO
1583  006426  010567  174142                MOV     R5,$BDDAT
1584  006432  016767  174170  174132        MOV     BCKGRD,$GDDAT
1585  006440  010067  174124                MOV     R0,$BDADR
1586  006444  004767  000144                JSR     PC,$HLT
1587  006450  000003                        3
1588
1589  006452  062704  000020          7$:   ADD     #20,R4                  ;GO TO THE NEXT ROW
1590  006456  042704  177017                BIC     #^C760,R4               ;DON'T LET IT GET TO BIG
1591  006462  001241                        BNE     8$                      ;GO DO NEXT ROW IF NOT TIME FOR A NEW COL
1592
1593  006464  062767  001000  177406  11$:  ADD     #1000,TCOL              ;GO TO THE NEXT COL
1594  006472  042767  140775  177400        BIC     #^C37002,TCOL           ;DON'T LET IT GET TO BIG
1595  006500  032767  020000  177372        BIT     #BIT13,TCOL
1596  006506  001627                        BEQ     8$                      ;IF MORE TO DO GO DO IT
1597  006510  010146                        MOV     R1,-(SP)
1598  006512  012701  020002                MOV     #BIT13!BIT01,R1
1599  006516  074167  177356                XOR     R1,TCOL
1600  006522  012601                        MOV     (SP)+,R1
1601  006524  000620                        BR      8$
1602
1603  006526  012605          CHK1:   MOV     (SP)+,R5                ;POP STACK INTO R5
1604  006530  012604                  MOV     (SP)+,R4                ;POP STACK INTO R4
1605  006532  012603                  MOV     (SP)+,R3                ;POP STACK INTO R3
1606  006534  012602                  MOV     (SP)+,R2                ;POP STACK INTO R2
1607  006536  105767  174012          TSTB    $ERFLG                  ;WAS THERE ANY ERRORS?
1608  006542  001002                  BNE     EXIT                    ;IF YES EXIT
1609  006544  062716  000002          ADD     #2,(SP)                 ;SET FOR NO ERRORS EXIT
1610  006550  000207          EXIT:   RTS     PC
1611
1612
1613
1614
1615  006552  010046          FORMAD: MOV     R0,-(SP)                ;SAVE R0
1616  006554  042700  160015          BIC     #^C17762,R0             ;STRIP EVERYTHING BUT THE ROW AND COL
1617  006560  062700  000020          ADD     #20,R0                  ;ADD A ROW
1618  006564  032700  020000          BIT     #BIT13,R0               ;C4
1619  006570  001405                  BEQ     1$                      ;BR IF NO
1620  006572  010146                  MOV     R1,-(SP)
1621  006574  012701  020002          MOV     #BIT13!BIT01,R1
1622  006600  074100                  XOR     R1,R0
1623  006602  012601                  MOV     (SP)+,R1
1624  006604  042716  017762    1$:   BIC     #17762,(SP)             ;STRIP THE ROW AND COL
1625  006610  052600                  BIS     (SP)+,R0                ;FORM ADDRESS
1626  006612  000207                  RTS     PC
1627
1628
1629                    ;*****************************************************************************
```

```
                                        ;THIS ROUTINE IS THE HLT HANDLER
                                        ;SW10=1 BELL ON ERROR
                                        ;SW13=1 INHIBIT ERROR TYPEOUTS
                                        ;SW15=1 HALT ON ERROR
                                        ;GO TO TYPERR ON ERROR
006614                                  $HLT:
006614  032737  002000  177570                 BIT     #SW10,@#SWR             ;BELL ON ERROR?
006622  001403                                 BEQ     1$                      ;NO - SKIP
006624  004767  000410                         JSR     PC,TYPE
006630  002644                                 $BELL
006632  005267  173714                 1$:     INC     $ERTTL                  ;COUNT THE NUMBER OF ERRORS
006636  001775                                 BEQ     1$                      ;INSURE $ERTTL NOT=0
006640  032737  020000  177570                 BIT     #SW13,@#SWR             ;SKIP TYPEOUT IF SET
006646  001015                                 BNE     2$                      ;SKIP TYPEOUTS
006650  011667  173710                         MOV     (SP),$HLTAD             ;GET ADDRESS OF HLT INSTRUCTION
006654  117767  173704  177700                 MOVB    @$HLTAD,$ITEMB          ;SAVE THE ERROR ITEM CODE
006662  162767  000002  173674                 SUB     #2,$HLTAD
006670  004767  000046                         JSR     PC,TYPERR               ;GO TO USER ERROR ROUTINE
006674  004767  000340                         JSR     PC,TYPE
007000  002647                                 $CRLF
007002  005737  177570                 2$:     TST     @#SWR                   ;HALT ON ERROR
007006  100001                                 BPL     3$                      ;SKIP IF CONTINUE
007010  000000                                 HALT                            ;HALT ON ERROR!
007012  112767  177777  173634         3$:     MOVB    #-1,$ERFLG
007020  062716  000002                         ADD     #2,(SP)
007024  032737  001000  177570                 BIT     #BIT09,@#SWR
007032  001002                                 BNE     4$
007034  016716  173702                         MOV     ESCAPE,(SP)
007040  000207                         4$:     RTS     PC
                                        
007042  010046                         TYPERR: MOV     R0,-(SP)                ;PUSH R0 ON STACK
007044  010146                                 MOV     R1,-(SP)                ;PUSH R1 ON STACK
007046  116700  173610                         MOVB    $ITEMB,R0               ;PICKUP THE ITEM NUMBER
007052  005300                                 DEC     R0                      ;FORM AN INDEX FOR THE TABLE
007054  006300                                 ASL     R0
007056  006300                                 ASL     R0
007060  006300                                 ASL     R0
007062  010746                                 MOV     PC,-(SP)
007064  062716  173710                         ADD     #$ERRTB-.,(SP)
007070  062600                                 ADD     (SP)+,R0
007072  012067  000012                         MOV     (R0)+,1$                ;PICKUP THE ERROR MESSAGE
007076  004767  000236                         JSR     PC,TYPE
007002  002647                                 $CRLF
007004  004767  000230                         JSR     PC,TYPE
                                        
007010  000000                         1$:     0                               ;ERROR MESSAGE POINTER GOES HERE
007012  012067  000014                         MOV     (R0)+,2$                ;PICKUP THE DATA HEADER POINTER
007016  032737  000400  177570                 BIT     #BIT08,@#SWR            ;TYPE THE DATA HEADER?
007024  001003                                 BNE     11$                     ;BR IF NO
007026  004767  000206                         JSR     PC,TYPE
007032  000000                         2$:     0
007034  012001                         11$:    MOV     (R0)+,R1                ;GET THE DATA STRING POINTER
007036  011000                                 MOV     (R0),R0                 ;GET THE DATA FORMAT POINTER
007040  010746                                 MOV     PC,     -(SP)
007042  062716  170736                         ADD     #-.,    (SP)
007046  061600                                 ADD     (SP),   R0
```

```
1686  007050  062601                        ADD     (SP)+,  R1
1687  007052  016767  173552  173516        MOV     INSTAD,$TMP0            ;SETUP THE COLUMN FOR TYPING
1688  007060  000367  173512                SWAB    $TMP0
1689  007064  006267  173506                ASR     $TMP0
1690  007070  042767  177740  173500        BIC     #^C37,$TMP0
1691  007076  016767  173526  173474        MOV     INSTAD,$TMP1            ;SET THE ROW FOR TYPING
1692  007104  006267  173470                ASR     $TMP1
1693  007110  006267  173464                ASR     $TMP1
1694  007114  006267  173460                ASR     $TMP1
1695  007120  006267  173454                ASR     $TMP1
1696  007124  042767  177740  173446        BIC     #^C37,$TMP1
1697  007132  016767  173472  173442        MOV     INSTAD,$TMP2            ;SETUP PRECHARGE FOR TYPING
1698  007140  006267  173436                ASR     $TMP2
1699  007144  006267  173432                ASR     $TMP2
1700  007150  042767  177774  173424        BIC     #^C3,$TMP2
1701  007156  000410                        BR      5$
1702  007160  005711                10$:    TST     (R1)                    ;TERMINATOR?
1703  007162  001003                        BNE     3$                      ;BR IF NO
1704  007164  012601                        MOV     (SP)+,R1                ;POP STACK INTO R1
1705  007166  012600                        MOV     (SP)+,R0                ;POP STACK INTO R0
1706  007170  000207                        RTS     PC
1707  007172  004067  000074        3$:     JSR     R0,$SPACE               ;GO TYPE SOME SPACES
1708  007176  000000                4$:     .WORD   0                       ;HOW MANY GOES HERE
1709  007200  112067  000030        5$:     MOVB    (R0)+,6$                ;PICKUP NUMBER OF DIGITS TO TYPE
1710  007204  112067  000025                MOVB    (R0)+,7$                ;PICKUP THE LEADING ZEROS FLAG
1711  007210  112067  177762                MOVB    (R0)+,4$                ;PICKUP THE NUMBER OF SPACES
1712  007214  012146                        MOV     (R1)+,-(SP)             ;PICKUP THE DATA
1713  007216  010746                        MOV     PC,-(SP)
1714  007220  062716  170560                ADD     #-.,(SP)
1715  007224  062616                        ADD     (SP)+,(SP)
1716  007226  013646                        MOV     @(SP)+,-(SP)
1717  007230  004067  000064                JSR     R0,$B2OCT               ;GO TYPE AN OCTAL DIGIT
1718  007234      000        6$:     .BYTE   0                       ;HOW MANY DIGITS
1719  007235      000        7$:     .BYTE   0                       ;LEADING ZERO SWITCH
1720  007236  000750                        BR      10$                     ;LOOP
1721
1722                        ;*****************************************************************************
1723
1724                        ;THIS IS A SMALL ROUTINE USED TO MAKE THE TYPE POINTERS POSITION INDEPENT
1725
1726  007240  010746        TYPE:   MOV     PC,-(SP)                ;GET THE POINTER
1727  007242  062716  170536        ADD     #-.,(SP)                ;GET THE PC
1728  007246  067616  000002        ADD     @2(SP),(SP)             ;STRIP THE RELATIVE ADDRESS
1729  007252  012667  000004        MOV     (SP)+,1$                ;AND SAVE IT FOR TYPE OUT
1730  007256  004767  171626        JSR     PC,$TYPE                ;GO TYPE TO
1731  007262  000000        1$:     .WORD   0                       ;POINTER TO MESSAGE
1732  007264  062716  000002        ADD     #2,(SP)                 ;SETUP TO RETURN
1733  007270  000207                RTS     PC                      ;GO BACK
1734
1735                        ;*****************************************************************************
1736                        ;ROUTINE TO TYPE SPACES ON THE TTY
1737                        ;CALL:
1738                        ;       JSR     R0,$SPACE
1739                        ;       NUMBER OF SPACES
1740                        ;       RETURN HERE
1741                        ;
```

```
1742  007272  012067  000002          $SPACE: MOV     (R0)+,2$        ;GET NUMBER OF SPACES DESIRED
1743                                                                  ;AND SET RETURN ADR.
1744  007276  005327                  1$:     DEC     (PC)+           ;FINISHED?
1745  007300  000000                  2$:     0
1746  007302  002001                          BGE     3$              ;BR IF NO
1747  007304  000200                          RTS     R0              ;GO HOME
1748  007306                          3$:
1749  007306  004767  177726                JSR     PC,TYPE
1750  007312  007316                          4$
1751  007314  000770                          BR      1$
1752  007316    040     000           4$:     .BYTE   40,0            ;STORAGE FOR ONE SPACE AND TERMINATOR
1753                                  ;*************************************************************************
1754                                  ;BINARY TO OCTAL (ASCII) AND TYPE
1755                                  ;$B2OCT---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
1756                                  ;CALL:
1757                                  ;       MOV     NUM,-(SP)               ;NUMBER TO BE TYPED
1758                                  ;       JSR     R0,$B2OCT               ;CALL FOR TYPEOUT
1759                                  ;       .BYTE   N                       ;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
1760                                  ;       .BYTE   M                       ;M=0 TO 3
1761                                                                          ;0=SUPPRESS LEADING ZEROS
1762                                                                          ;1=TYPE LEADING ZEROS
1763                                                                          ;2=SUPPRESS LEADING ZEROS
1764                                                                          ;WITH PHYSICAL ADDRESS
1765                                                                          ;3=TYPE LEADING ZEROS
1766                                                                          ;WITH PHYSICAL ADDRESS
1767                                  ;
1768                                  ;$B201----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST $B5OCT OR $B2016
1769                                  ;CALL:
1770                                  ;       MOV     NUM,-(SP)
1771                                  ;       JSR     R0,$B201
1772                                  ;
1773                                  ;$B2016---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
1774                                  ;CALL:
1775                                  ;       MOV     NUM,-(SP)
1776                                  ;       JSR     R0,$B2016
1777
1778  007320  112067  000267          $B2OCT: MOVB    (R0)+,$OMODE+1          ;PICKUP THE NUMBER OF DIGITS TO TYPE
1779  007324  112067  000261                  MOVB    (R0)+,$OFILL            ;GET THE ZERO FILL SWITCH
1780  007330  000406                          BR      $B201
1781  007332  112767  000001  000251  $B2016: MOVB    #1,$OFILL               ;SET THE ZERO FILL SWITCH
1782  007340  112767  000006  000245          MOVB    #6,$OMODE+1             ;SET FOR SIX(6) DIGITS
1783  007346  112767  000005  000234  $B201:  MOVB    #5,$OCNT                ;SET THE ITERATION COUNT
1784  007354  010346                          MOV     R3,-(SP)                ;SAVE R3
1785  007356  010446                          MOV     R4,-(SP)                ;SAVE R4
1786  007360  010546                          MOV     R5,-(SP)                ;SAVE R5
1787  007362  116704  000225                  MOVB    $OMODE+1,R4             ;GET THE NUMBER OF DIGITS TO TYPE
1788  007366  005404                          NEG     R4
1789  007370  062704  000006                  ADD     #6,R4                   ;SUBTRACT IT FOR MAX. ALLOWED
1790  007374  110467  000212                  MOVB    R4,$OMODE               ;SAVE IT FOR USE
1791  007400  116704  000205                  MOVB    $OFILL,R4               ;GET THE ZERO FILL SWITCH
1792  007404  042704  177776                  BIC     #177776,R4              ;
1793  007410  016605  000010                  MOV     10(SP),R5               ;PICKUP THE INPUT NUMBER
1794  007414  005003                          CLR     R3                      ;CLEAR THE OUTPUT WORD
1795  007416  132767  000002  000165          BITB    #2,$OFILL               ;
1796  007424  001424                          BEQ     1$                      ;
1797  007426  005767  171670                  TST     $KT11
```

```
1798  007432  100021                        BPL     1$
1799  007434  032777  000001  170440        BIT     #1,@SR0                 ;
1800  007442  001415                        BEQ     1$                      ;
1801  007444  017746  173216                MOV     @PAGE,-(SP)             ;
1802  007450  042705  160000                BIC     #160000,R5              ;
1803  007454  006316                        ASL     (SP)                    ;
1804  007456  006316                        ASL     (SP)                    ;
1805  007460  006316                        ASL     (SP)                    ;
1806  007462  006316                        ASL     (SP)                    ;
1807  007464  006316                        ASL     (SP)                    ;
1808  007466  006103                        ROL     R3                      ;
1809  007470  006316                        ASL     (SP)                    ;
1810  007472  006103                        ROL     R3                      ;
1811  007474  062605                        ADD     (SP)+,R5                ;
1812  007476  006105                1$:     ROL     R5                      ;ROTATE MSB INTO "C"
1813  007500  000404                        BR      3$                      ;GO DO MSB
1814  007502  006105                2$:     ROL     R5                      ;FORM THIS DIGIT
1815  007504  006105                        ROL     R5
1816  007506  006105                        ROL     R5
1817  007510  010503                        MOV     R5,R3
1818  007512  006103                3$:     ROL     R3                      ;GET LSB OF THIS DIGIT
1819  007514  105367  000072                DECB    $OMODE                  ;TYPE THIS DIGIT?
1820  007520  100017                        BPL     7$                      ;BR IF NO
1821  007522  042703  177770                BIC     #177770,R3              ;GET RID OF JUNK
1822  007526  001002                        BNE     4$                      ;TEST FOR 0
1823  007530  005704                        TST     R4                      ;SUPPRESS THIS 0?
1824  007532  001403                        BEQ     5$                      ;BR IF YES
1825  007534  005204                4$:     INC     R4                      ;DON'T SUPPRESS ANYMORE 0'S
1826  007536  052703  000060                BIS     #'0,R3                  ;MAKE THIS DIGIT ASCII
1827  007542  052703  000040          5$:   BIS     #' ,R3                  ;MAKE ASCII IF NOT ALREADY
1828  007546  110367  000034                MOVB    R3,8$                   ;SAVE FOR TYPING
1829  007552  004767  177462                JSR     PC,TYPE
1830  007556  007606                        8$
1831  007560  105367  000024          7$:   DECB    $OCNT                   ;COUNT BY 1
1832  007564  003346                        BGT     2$                      ;BR IF MORE TO DO
1833  007566  002402                        BLT     6$                      ;BR IF DONE
1834  007570  005204                        INC     R4                      ;INSURE LAST DIGIT ISN'T A BLANK
1835  007572  000743                        BR      2$                      ;GO DO THE LAST DIGIT
1836  007574  012605                6$:     MOV     (SP)+,R5                ;RESTORE R5
1837  007576  012604                        MOV     (SP)+,R4                ;RESTORE R4
1838  007600  012603                        MOV     (SP)+,R3                ;RESTORE R3
1839  007602  012616                        MOV     (SP)+,(SP)              ;SET THE STACK FOR RETURNING
1840  007604  000200                        RTS     R0                      ;RETURN
1841  007606     000                8$:     .BYTE   0                       ;STORAGE FOR ASCII DIGIT
1842  007607     000                        .BYTE   0                       ;TERMINATOR FOR TYPE ROUTINE
1843  007610     000          $OCNT:    .BYTE   0                       ;OCTAL DIGIT COUNTER
1844  007611     000          $OFILL:   .BYTE   0                       ;ZERO FILL SWITCH
1845  007612  000000          $OMODE:   0                               ;NUMBER OF DIGITS TO TYPE
```

```
1846
1847
1848                                    ;*****************************************************************************
1849                                    ;*****************************************************************************
1850                                    ;ASCII MESSAGES
1851  007614  042524  052123  050040  EM1:    .ASCIZ  /TEST PATTERN/<15><12>
1852  007622  052101  042524  047122
1853  007630  005015     000
1854  007633     111  051516  051124  EM2:    .ASCIZ  /INSTRUCTION CHANGED/<15><12>
1855  007640  041525  044524  047117
1856  007646  041440  040510  043516
1857  007654  042105  005015     000
1858  007661     102  041501  020113  EM3:    .ASCIZ  /BACK GROUND CHANGED/<15><12>
1859  007666  051107  052517  042116
1860  007674  041440  040510  043516
1861  007702  042105  005015     000
1862  007707     105  051122  051117  DH1:    .ASCII  /ERROR   FAILING/<15><12>
1863  007714  020040  043040  044501
1864  007722  044514  043516  005015
1865  007730  041520  020040  020040          .ASCIZ  /PC      ADDRESS EXPECT  RCV'D   COL  ROW  PRECHARGE/<15><12>
1866  007736  020040  042101  051104
1867  007744  051505  020123  054105
1868  007752  042520  052103  020040
1869  007760  041522  023526  020104
1870  007766  020040  047503  020114
1871  007774  051040  053517  020040
1872  010002  051120  041505  040510
1873  010010  043522  006505  000012
1874  010016  051105  047522  020122  DH2:    .ASCII  /ERROR   INST    EXPECT  RCV'D/<15><12>
1875  010024  020040  047111  052123
1876  010032  020040  020040  054105
1877  010040  042520  052103  020040
1878  010046  041522  023526  006504
1879  010054     012
1880  010055     120  020103  020040          .ASCIZ  /PC      ADDRESS DATA    DATA    COL  ROW  PRECHARGE/<15><12>
1881  010062  020040  040440  042104
1882  010070  042522  051523  042040
1883  010076  052101  020101  020040
1884  010104  042040  052101  020101
1885  010112  020040  041440  046117
1886  010120  020040  047522  020127
1887  010126  050040  042522  044103
1888  010134  051101  042507  005015
1889  010142     000
1890  010143     105  051122  051117  DH3:    .ASCII  /ERROR   BCKGRD  EXPECT  RCV'D/<15><12>
1891  010150  020040  041040  045503
1892  010156  051107  020104  042440
1893  010164  050130  041505  020124
1894  010172  051040  053103  042047
1895  010200  005015
1896  010202  041520  020040  020040          .ASCIZ  /PC      ADDRESS DATA    DATA    COL  ROW  PRECHARGE/<15><12>
1897  010210  020040  042101  051104
1898  010216  051505  020123  040504
1899  010224  040524  020040  020040
1900  010232  040504  040524  020040
1901  010240  020040  047503  020114
```

```
 1902  010246  051040  053517  020040
 1903  010254  051120  041505  040510
 1904  010262  043522  006505  000012
 1905                                    .EVEN
 1906  010270  002564  002570  002572  DT1:    .WORD   $HLTAD,$BDADR,$GDDAT,$BDDAT,$TMP0,$TMP1,$TMP2,0
 1907  010276  002574  002576  002600
 1908  010304  002602  000000
 1909  010310     006     001     002  DF1:    .BYTE   6,1,2
 1910  010313     006     003     002          .BYTE   6,3,2
 1911  010316     006     001     002          .BYTE   6,1,2
 1912  010321     006     001     003          .BYTE   6,1,3
 1913  010324     002     000     003          .BYTE   2,0,3
 1914  010327     002     000     005          .BYTE   2,0,5
 1915  010332     002     000     000          .BYTE   2,0,0
 1916  010335     105  042116  050040  EPASS:  .ASCIZ  /END PASS/<15><12>
 1917  010342  051501  006523  000012
 1918  010350  042523  020124  053523  OPMODE: .ASCII  /SET SWITCHES FOR OPERATING MODE REFER SECTION 4.1/<15><12>
 1919  010356  052111  044103  051505
 1920  010364  043040  051117  047440
 1921  010372  042520  040522  044524
 1922  010400  043516  046440  042117
 1923  010406  020105  042522  042506
 1924  010414  020122  042523  052103
 1925  010422  047511  020116  027064
 1926  010430  006461     012
 1927  010433     040  042511  020056          .ASCIZ  / IE. PICKUPS OR DROPS OR BOTH THEN HIT CONTINUE/<15><12>
 1928  010440  044520  045503  050125
 1929  010446  020123  051117  042040
 1930  010454  047522  051520  047440
 1931  010462  020122  047502  044124
 1932  010470  052040  042510  020116
 1933  010476  044510  020124  047503
 1934  010504  052116  047111  042525
 1935  010512  005015     000
 1936  010515     123  052105  051440  OPMOD2: .ASCIZ  /SET SWITCHES/<15><12>
 1937  010522  044527  041524  042510
 1938  010530  006523  000012
 1939  010534  054524  042520  021040  STARTB: .ASCII  /TYPE "STARTING BANK" NUMBER AND CR /<15><12>
 1940  010542  052123  051101  044524
 1941  010550  043516  041040  047101
 1942  010556  021113  047040  046525
 1943  010564  042502  020122  047101
 1944  010572  020104  051103  006440
 1945  010600     012
 1946  010601     122  043105  051105          .ASCIZ  /REFER TO SECTION 4.3.1 BEFORE CHOOSING /<15><12>
 1947  010606  052040  020117  042523
 1948  010614  052103  047511  020116
 1949  010622  027064  027063  020061
 1950  010630  042502  047506  042522
 1951  010636  041440  047510  051517
 1952  010644  047111  020107  005015
 1953  010652     000
 1954  010653     123  040524  052122  START2: .ASCIZ  /STARTING BANK?/<15><12>
 1955  010660  047111  020107  040502
 1956  010666  045516  006477  000012
 1957  010674  054524  042520  047040  NUMBER: .ASCIZ  /TYPE NUMBER OF 4K BANKS TO BE TESTED AND CR /<15><12>
```

```
 1958  010702  046525  042502  020122
 1959  010710  043117  032040  020113
 1960  010716  040502  045516  020123
 1961  010724  047524  041040  020105
 1962  010732  042524  052123  042105
 1963  010740  040440  042116  041440
 1964  010746  020122  005015     000
 1965  010753     043  047440  020106  NUMB2:  .ASCIZ  /# OF BANKS?/<15><12>
 1966  010760  040502  045516  037523
 1967  010766  005015     000
 1968  010771     114  051501  020124  MEMEND: .ASCIZ  /LAST MEMORY ADDRESS IS /
 1969  010776  042515  047515  054522
 1970  011004  040440  042104  042522
 1971  011012  051523  044440  020123
 1972  011020     000
 1973  011021     120  051101  052111  PAERR:  .ASCIZ  /PARITY ERROR DETECTED AT LOC.  /
 1974  011026  020131  051105  047522
 1975  011034  020122  042504  042524
 1976  011042  052103  042105  040440
 1977  011050  020124  047514  027103
 1978  011056  020040     000
 1979  011061     111  020116  040520  NOPARE: .ASCIZ  /IN PARITY TEST NO PARITY ERROR DECTED/<15><12>
 1980  011066  044522  054524  052040
 1981  011074  051505  020124  047516
 1982  011102  050040  051101  052111
 1983  011110  020131  051105  047522
 1984  011116  020122  042504  052103
 1985  011124  042105  005015     000
 1986  011131     120  051101  052111  PARERR: .ASCIZ  /PARITY ERROR HAS OCCURED SO PARITY TEST HAS STARTED/<15><12>
 1987  011136  020131  051105  047522
 1988  011144  020122  040510  020123
 1989  011152  041517  052503  042522
 1990  011160  020104  047523  050040
 1991  011166  051101  052111  020131
 1992  011174  042524  052123  044040
 1993  011202  051501  051440  040524
 1994  011210  052122  042105  005015
 1995  011216     000
 1996  011217     123  051117  054522  SORRY:  .ASCIZ  /SORRY LAST INPUT INVALID TRY AGAIN/<15><12>
 1997  011224  046040  051501  020124
 1998  011232  047111  052520  020124
 1999  011240  047111  040526  044514
 2000  011246  020104  051124  020131
 2001  011254  043501  044501  006516
 2002  011262  000012
 2003  011264  051120  043517  040522  PROTO:  .ASCII  /PROGRAM RELOCATED/<15><12>
 2004  011272  020115  042522  047514
 2005  011300  040503  042524  006504
 2006  011306     012
 2007  011307     122  051505  040524          .ASCIZ  /RESTART ADDRESS   /
 2008  011314  052122  040440  042104
 2009  011322  042522  051523  020040
 2010  011330  000040
 2011  011332  047524  050040  052125  PROBAK: .ASCIZ  /TO PUT PROGRAM BACK TO FIRST 4K START  /
 2012  011340  050040  047522  051107
 2013  011346  046501  041040  041501
```

```
 2014  011354  020113  047524  043040
 2015  011362  051111  052123  032040
 2016  011370  020113  052123  051101
 2017  011376  020124  000040
 2018  011402  047524  051040  051505  RELOD:  .ASCIZ  /TO RESTORE LOADER START  /
 2019  011410  047524  042522  046040
 2020  011416  040517  042504  020122
 2021  011424  052123  051101  020124
 2022  011432  000040
 2023  011434  050040  052514  020123  RELOD2: .ASCIZ  / PLUS RELOCATION FACTOR/<15><12>
 2024  011442  042522  047514  040503
 2025  011450  044524  047117  043040
 2026  011456  041501  047524  006522
 2027  011464  000012
 2028  011466  047514  042101  051105  FIN:    .ASCIZ  /LOADER RESTORED/<15><12>
 2029  011474  051040  051505  047524
 2030  011502  042522  006504  000012
 2031                                  .EVEN
 2032  011510  000000                  LAST:   .WORD   0
 2033          000001                  .END
```

F04


```
BANKS   003124            944#  1061*  1121   1409*  1420*
BCKGRD  002626            836#  1153*  1164*  1171*  1176*  1180*  1255   1470   1581   1584
BEGIN1  002746            451    911#
BEGIN2  004042            946   1003   1023   1063   1069   1104#  1388
BEGIN3  004102            452    789   1111   1118#  1433   1457
BEGIN4  004156           1130#  1132
BEGIN5  004246           1143   1148#
BEGIN6  004312           1159#  1370   1445
BEGN11  003454            449   1029#
BEG2    005326            946*  1063*  1388#
BIT0  = 000001            394#
BIT00 = 000001            384#   394   1319
BIT01 = 000002            383#   393   1198   1271   1324   1598   1621
BIT02 = 000004            382#   392
BIT03 = 000010            381#   391
BIT04 = 000020            380#   390
BIT05 = 000040            379#   389
BIT06 = 000100            378#   388
BIT07 = 000200            377#   387    917   1033
BIT08 = 000400            376#   386    838   1154   1677
BIT09 = 001000            375#   385   1655
BIT1  = 000002            393#
BIT10 = 002000            374#
BIT11 = 004000            373#
BIT12 = 010000            372#
BIT13 = 020000            371#  1196   1271   1276   1326   1595   1598   1618   1621
BIT14 = 040000            370#  1110   1274
BIT15 = 100000            369#   570
BIT2  = 000004            392#
BIT3  = 000010            391#
BIT4  = 000020            390#
BIT5  = 000040            389#
BIT6  = 000100            388#
BIT7  = 000200            387#
BIT8  = 000400            386#
BIT9  = 001000            385#
BPTVEC= 000014            401#
CHECK   006102           1316   1511#
CHK1    006526           1525   1533   1603#
COL     002614            831#  1148*  1194   1196   1270
CYCNT   002622            834#  1151*  1297
DF1     010310            875    882    889   1909#
DH1     007707            872   1862#
DH2     010016            879   1874#
DH3     010143            886   1890#
DISPLA= 177570            326#  1303*  1307*
DT1     010270            874    881    888   1906#
EMTVEC= 000030            404#
EM1     007614            871   1851#
EM2     007633            878   1854#
EM3     007661            885   1858#
EPASS   010335           1453   1916#
ERRVEC= 000004            397#   568*   629*   638*   647*   725*   726*   733*   745*   785    786*
ESCAPE  002642            842#  1526*  1657
EXIT    006550           1608   1610#
FILL4K  004622           1255#
```

```
FIN     011466          486  2028#
FIRST   000100          410#  948   949   997  1065  1066  1390  1391  1424  1425
FORMAD  006552         1477  1615#
FROM    005332          466   948*  1000  1065* 1390# 1406  1424* 1437  1441
FRSTAD  000216          458#
HIBASE  002662          851# 1135*  1337  1350
HILMT   002610          829# 1134* 1144*  1329  1344* 1349* 1412* 1488
INSIDE  002732          897#  898  1231
INSTAD  002630          837# 1208*  1302  1306  1465  1469  1474* 1476  1490* 1516  1687  1691  1697
INSTR1  002724          893#  894  1219  1493  1550  1553
INSTR2  002726          894# 1494  1562  1565
INSTR3  002730          895# 1495
IN2AD   000214          457#
IOTVEC= 000020          402#
KIPAR0  000132          436#  582*  753*  1363*
KIPAR1  000134          437#  583*  755*  770   800*
KIPAR2  000136          438#  584*
KIPAR3  000140          439#  585*
KIPAR4  000142          440#  586*  770*
KIPAR5  000144          441#  587*  608
KIPAR6  000146          442#  588*  603
KIPAR7  000150          443#  589*  632   758*
KIPDR0  000112          425#  572*  754*  1355*
KIPDR1  000114          426#  573*  757*  1356*
KIPDR2  000116          427#  574*  756*  1357*
KIPDR3  000120          428#  575* 1358*
KIPDR4  000122          429#  576* 1359*
KIPDR5  000124          430#  577* 1360*
KIPDR6  000126          431#  578* 1361*
KIPDR7  000130          432#  579*  759*  1362*
LAST    011510          480   482   978   979  1088  1089  2032#
LDINT   005660         1291  1462#
LDPAT   004642         1267#
LENGTH  005330          467   947*  999  1064* 1389# 1405
LOADER  000324          479#  984  1094
LOBASE  002660          850# 1126* 1139* 1339  1341* 1346* 1350  1352
LODER   002672          481   855#  971   976* 1068  1086*
LOLMT   002606          828# 1127* 1342* 1347* 1410*
LOOP1   004452         1192# 1323  1332
LOOP12  005072         1330  1332#
LOOP2   004722         1291# 1318
LPERR   004546         1217#
LSTBK   002664          852# 1051* 1052* 1053* 1054* 1055* 1056* 1057* 1058* 1059* 1060*  1061  1420
MAMF    002176          723# 1112
MEMEND  010771          920  1036  1968#
MMVEC = 000250          407#  748*  787*
MOD     002612          830# 1128* 1192  1257  1326* 1329  1343* 1348* 1411*
MSKBIT  002632          838# 1154* 1496
NEXT    005012         1293  1317  1319#
NOPARE  011061          795  1979#
NUMBER  010674          941  1957#
NUMB2   010753          942  1965#
OPMODE  010350         1107  1918#
OPMOD2  010515         1108  1936#
OUTSID  002740          901# 1166* 1173* 1177* 1181* 1237
PAERR   011021          769  1973#
```

```
PAGE    002666          620*   853#   923*  1039*  1352*  1801
PARCSR= 172100          719#   727
PARERR  011131          739   1986#
PARSRV  002254          723    738#   782
PARVEC= 000114          406#   723*   724*   742*   783*
PASFLG  002652          847#   913*  1031*  1140*  1161*  1168*  1367*  1369*  1444*  1447
PC     =%000007         338#   479    485*   526*   527*   534*   539*   563    566    592    627    636    659*
                        682*   693*   698*   701*   706*   708*   734*   738*   740    743    746    768*   778*
                        781    784    788    794*   895*   899*   911    915*   918*   919*   932*   934*   938*
                        940*   943*   977    982*   988*   992   1004*  1011*  1013*  1020*  1029   1034*  1035*
                       1048*  1087   1092*  1098*  1104*  1106*  1112*  1217   1229   1235   1291*  1299   1304*
                       1316*  1452*  1477*  1505*  1524   1544*  1555*  1567*  1578*  1586*  1610*  1626*  1638*
                       1647*  1648*  1658*  1667   1671*  1673*  1679*  1683   1706*  1713   1726   1730*  1733*
                       1744*  1749*  1829*
PCHRG   002620          833#  1150*  1199   1321*  1322*
PROBAK  011332         1014   2011#
PROTO   011264         1005   2003#
PS     = 177776         323#   324
PSW    = 177776         324#
PWRVEC= 000024          403#
REDONE  002656          849#  1067*  1415*  1416   1423*  1436*
RELOD   011402          983   1093   2018#
RELOD2  011434          989   1099   2023#
RESTOR  000300          465'  1016
RESVEC= 000010          398#
ROW     002616          832#  1149*  1201   1467   1571
R0     =%000000         329#   519    520*   521    524*   560    634*   639*   641*   646*   650    654*   727*
                        730*   749*   764    771    772*   801*   893*   897*   926*   979*   981*   985*   999*
                       1002*  1008*  1017*  1042*  1089*  1091*  1095*  1118*  1119*  1125*  1203*  1204*  1205*
                       1206*  1207   1208   1217*  1219*  1239*  1241*  1243*  1255*  1260   1267*  1268   1281
                       1295   1308*  1310*  1354*  1355   1356   1357   1358   1359   1360   1361   1362   1405*
                       1414*  1465*  1466*  1467   1476*  1478   1490   1491   1492   1493*  1494*  1495*  1497
                       1498*  1499   1517*  1520   1522   1529*  1530*  1531*  1532   1535   1536   1543   1548
                       1554   1557*  1560   1566   1569*  1577   1585   1615   1616*  1617*  1618   1622*  1625*
                       1660   1662*  1663*  1664*  1665*  1666*  1669*  1670   1676   1681   1682*  1685*  1705*
                       1707*  1709   1710   1711   1717*  1742   1747*  1778   1779   1840*
R1     =%000001         330#   465*   468    482*   484*   559    562*   640*   642    643    645*   649    655*
                        894*   898*  1000*  1001   1121*  1132*  1229*  1231*  1235*  1237*  1239   1241   1243
                       1257*  1260*  1268*  1269*  1279   1297*  1309*  1406*  1413   1437*  1439*  1442*  1515*
                       1536   1597   1598*  1599   1600*  1620   1621*  1622   1623*  1661   1681*  1686*  1702
                       1704*  1712
R2     =%000002         331#   466*   468*   481*   483*   728*   731*   752*   754    757    759    970*   1003*
                       1022   1083*  1122*  1123*  1127   1128   1130*  1133*  1134   1259*  1261*  1270*  1273*
                       1274*  1275   1306*  1307   1308   1407*  1413*  1438*  1440*  1443*  1463   1479*  1480
                       1483   1486   1499   1504*  1511   1516*  1517   1518   1532   1548   1606*
R3     =%000003         332#   467*   469*   558    571*   572    573    574    575    576    577    578    579
                        592*   594*   595*   596*   597*   598*   599    610*   611    612    613    614    656*
                        974*   975*   976    980   1084*  1085*  1086   1090   1120*  1124*  1126   1129*  1131*
                       1135   1142   1271*  1278   1441*  1443   1464   1478*  1480   1482*  1483   1485*  1486
                       1488   1503*  1512   1518*  1519*  1560   1605*  1784   1794*  1808*  1810*  1817*  1818*
                       1821*  1826*  1827*  1828   1838*
R4     =%000004         333#   479*   480*   483    557    603*   608*   620    621*   631*   632    634    657*
                        958*   959*   960    962*   964*   965*   966*   967*   968*   969*   970    974    995
                        997*   998   1001*  1071*  1072*  1073   1075*  1077*  1078*  1079*  1080*  1081*  1082*
                       1083   1084   1272*  1280   1513   1520*  1521*  1530   1571   1589*  1590*  1604*  1785
                       1787*  1788*  1789*  1790   1791*  1792*  1823   1825*  1834*  1837*
```

```
R5      =%000005        334#    556     561*    648     658*    977*    978*    980*    1087*   1088*   1090*   1275*   1276
                        1278*   1279*   1280*   1281    1416*   1423    1436    1514    1535*   1539    1541    1550    1552
                        1562    1564    1573    1575    1581    1583    1603*   1786    1793*   1802*   1811*   1812*   1814*
                        1815*   1816*   1817    1836*
R6      =%000006        335#    337
R7      =%000007        336#    338
SEGMT   000076          1327    1331    1337#
SEGMT1  005150          1141    1340    1346#
SEGMT2  005210          1345    1352#
SEGVEC  000152          447#
SEGO    005124          611*    612*    615*    1342#
SEG10   005156          613*    614*    616*    1347#
SELF    002654          848#    945*    1062*   1395
SORRY   011217          709     1996#
SP      =%000006        337#    519*    520     521*    523     524     525*    528     530*    532*    538     556*    557*
                        558*    559*    560*    561     566*    567*    568     627*    628*    629     636*    637*    638
                        648*    654     655     656     657     658     674*    677*    678*    679     684     685*    686*
                        687*    691     695     696*    697*    699     711*    712*    713*    714*    715*    740*    741*
                        742     743*    744*    745     746*    747*    748     771*    781*    782*    783     784*    785*
                        786     788*    789*    790     911*    912*    925*    984*    992*    993*    995*    996*    1006*
                        1007*   1015*   1016*   1029*   1030*   1041*   1094*   1192*   1193*   1194*   1195*   1198*   1199*
                        1200*   1201*   1202*   1203    1204    1205    1206    1302*   1303    1304    1421*   1435*   1463*
                        1464*   1469*   1470*   1471*   1472*   1496*   1497*   1498    1501*   1503    1504    1511*   1512*
                        1513*   1514*   1524*   1525*   1526    1597*   1600    1603    1604    1605    1606    1609*   1615*
                        1620*   1623    1624*   1625    1644    1654*   1657*   1660*   1661*   1667*   1668*   1669    1683*
                        1684*   1685    1686    1704    1705    1712*   1713*   1714*   1715*   1716*   1726*   1727*   1728*
                        1729    1732*   1784*   1785*   1786*   1793    1801*   1803*   1804*   1805*   1806*   1807*   1809*
                        1811    1836    1837    1838    1839*
SR0     000102          416#    569     626*    653*    760*    777*    924*    931*    1040*   1047*   1364*   1401*   1456*
                        1799
SR1     000104          417#
SR2     000106          418#
SR3     000110          419#    1353*
STARTB  010534          936     1939#
START2  010653          937     1954#
STBANK  003102          939#    990     1050*   1118    1136    1408*   1419*
SVIN1   002636          840#    1491*   1557
SVIN2   002640          841#    1472    1492*   1569
SV400   002634          839#    1295*   1310
SWITCH  004742          1221*   1223*   1300#
SWR     = 177570        325#    326     1110    1159    1169    1221    1319    1324    1365    1450    1636    1642    1650
                        1655    1677
SW0     = 000001        366#
SW00    = 000001        356#    366
SW01    = 000002        355#    365
SW02    = 000004        354#    364
SW03    = 000010        353#    363
SW04    = 000020        352#    362
SW05    = 000040        351#    361
SW06    = 000100        350#    360
SW07    = 000200        349#    359
SW08    = 000400        348#    358
SW09    = 001000        347#    357
SW1     = 000002        365#
SW10    = 002000        346#    1450    1636
SW11    = 004000        345#
```

```
SW12   = 010000          344#  1159   1169   1365
SW13   = 020000          343#  1642
SW14   = 040000          342#
SW15   = 100000          341#
SW2    = 000004          364#
SW3    = 000010          363#
SW4    = 000020          362#
SW5    = 000040          361#
SW6    = 000100          360#
SW7    = 000200          359#
SW8    = 000400          358#
SW9    = 001000          357#
TBITVE= 000014           399#
TCOL    006100          1509#  1522*  1523*  1531   1593*  1594*  1595   1599*
TKB     001772           670#   677
TKS     001770           669#   675
TO      005334           949*  1066*  1391#  1407   1425*  1438   1442
TO1     002670           465    854#   998*  1006   1015
TRAPVE= 000034           405#
TRTVEC= 000014           400#
TSTADR  002604           827#  1207*  1267   1475*  1479   1515
TSTPAT  002624           835#  1152*  1165*  1172*  1175*  1179*  1272   1474   1475   1539   1542   1573   1576
TYPE    007240           485    682    693    701    706    708    738    768    778    794    915    919    932
                         934    940    982    988   1004   1011   1013   1020   1035   1048   1092   1098   1104
                        1106   1452   1638   1648   1671   1673   1679   1726#  1749   1829
TYPERR  006742          1647   1660#
$BDADR  002570           821#  1543*  1554*  1566*  1577*  1585*  1906
$BDDAT  002574           823#  1541*  1552*  1564*  1575*  1583*  1906
$BELL   002644           843#  1639
$B20CT  007320           772    926    985   1008   1017   1042   1095   1717   1778#
$B201   007346          1780   1783#
$B2016  007332          1781#
$CORE   001652           565    567    636#
$COROU  001706           637    645#
$CRLF   002647           694    707    779    845#   916    933   1012   1021   1049   1105   1649   1672
$EOP    005336          1325   1338   1351   1395#
$EOP1   005612          1396   1432   1447#
$EOP2   005620          1449#
$EOP3   005642          1448   1451   1454#
$ERFLG  002554           814#  1245*  1607   1653*
$ERRTB  002674           869#  1668
$ERTTL  002552           813#  1640*
$FILLS  001105           511#
$GOADR  002566           820#
$GODAT  002572           822#  1542*  1553*  1565*  1576*  1584*  1906
$HLT    006614          1544   1555   1567   1578   1586   1635#
$HLTAD  002564           819#  1644*  1645   1646*  1906
$ICNT   002544           810#
$ITEMB  002562           817#  1645*  1662
$KTOUT  001646           628    634#
$KT11   001322           564#   570*   651    750    775    917*   921    929   1033*  1037   1045   1399   1454
                        1797
$LF     002650           846#
$LPADR  002546           811#
$LPERR  002550           812#
$LSTAD  001764           649*   660#   925   1041
```

```
$LSTBK  001766          650* 661* 923  958  1039 1051 1071
$NULL   001104          510* 530
$OCNT   007610         1783* 1831* 1843#
$OMODE  007612         1778* 1782* 1787  1790* 1819* 1845#
$PASS   002540          808# 1182* 1449*
$QUES   002646          844#
$SEG    001326          566#
$SIZE   001300          556# 918  1034
$SIZEX  001712          635  646#
$SPACE  007272         1707  1742#
$TMP0   002576          824# 1687* 1688* 1689* 1690* 1906
$TMP1   002600          825# 1691* 1692* 1693* 1694* 1695* 1696* 1906
$TMP2   002602          826# 1697* 1698* 1699* 1700* 1906
$TPB    001102          509# 538*
$TPFLG  001106          512# 515
$TPS    001100          508# 536
$TSTNM  002542          809#
$TYPE   001110          515# 1730
$OFILL  007611         1779* 1781* 1791  1795  1844#
.     = 011512          409# 448# 462#  480   492#  540#  567  628  637  741  744  747  782
                        785  789  912   978   1030  1088  1219 1231 1237 1525 1668 1684 1714
                       1727
.RECD   001774          673# 938  943
```

HLT   321#
SCOPE   322#

TEST MOS MEMORY MACY11 27(732) 22-SEP-76 14:48 PAGE 54
DCMSBB.P11 CROSS REFERENCE TABLE -- PERMANENT SYMBOLS

```
ADD      480   525   567   628   631   637   640   641   697   741   744   747   782   785   789
         800   912   975   978   996   997  1003  1007  1016  1030  1052  1085  1088  1123  1124
        1130  1131  1219  1231  1237  1273  1321  1326  1346  1439  1440  1471  1482  1485  1501
        1519  1525  1589  1593  1609  1617  1654  1668  1669  1684  1685  1686  1714  1715  1727
        1728  1732  1789  1811
ASL      712   713   714   731  1664  1665  1666  1803  1804  1805  1806  1807  1809
ASR      686   687  1689  1692  1693  1694  1695  1698  1699
BCC      732
BCS     1327
BEQ      516   600   705  1111  1170  1197  1227  1277  1301  1396  1417  1468  1481  1484  1487
        1500  1533  1540  1551  1563  1574  1582  1596  1619  1637  1641  1796  1800  1824
BGE     1746
BGT      633  1832
BHI      961  1074  1143  1338  1489
BIC      598   678   703   711   993  1193  1195  1200  1202  1223  1269  1274  1322  1466  1521
        1523  1529  1590  1594  1616  1624  1690  1696  1700  1792  1802  1821
BIS      570   715  1198  1204  1205  1206  1279  1530  1531  1625  1826  1827
BIT     1110  1159  1169  1196  1276  1319  1324  1365  1450  1595  1618  1636  1642  1655  1677
        1799
BITB    1795
BLO     1137  1330  1351
BLOS    1340
BLT      533  1833
BNE      522   529   644   680   692   972   991   994  1069  1160  1163  1282  1320  1323  1325
        1366  1448  1451  1537  1549  1561  1572  1591  1608  1643  1656  1678  1703  1822
BPL      537   565   652   676   751   776   922   930  1038  1046  1400  1455  1651  1798  1820
BR       518   535   604   635   688   710   716   765   796  1023  1138  1167  1174  1178  1233
        1293  1305  1317  1318  1331  1345  1368  1418  1473  1527  1546  1558  1570  1580  1601
        1701  1720  1751  1780  1813  1835
CLC      593   963  1076
CLR      582   621   639   653   674   733   749   753   755   756   777   893   897   913   931
         945  1031  1047  1067  1145  1149  1150  1164  1165  1168  1171  1172  1177  1181  1182
        1353  1363  1369  1401  1410  1411  1456  1794
CLRB    1245
CMP      599   632   643   704   960  1073  1136  1142  1281  1329  1337  1339  1350  1467  1480
        1483  1486  1488  1499  1532  1536  1539  1548  1550  1560  1562  1571  1573  1581
CMPB     528   679   681
COM     1140  1161  1367  1415  1444
DEC     1060  1133  1663  1744
DECB     532  1819  1831
EMT      321
HALT     487   517  1109  1652
INC     1119  1449  1640  1825  1834
IOT      322
JMP      449   451   452   470   790  1022  1141  1332  1370  1432  1433  1445  1457
JSR      485   527   534   682   693   701   706   708   738   768   772   778   794   915   918
         919   926   932   934   938   940   943   982   985   988  1004  1008  1011  1013  1017
        1020  1034  1035  1042  1048  1092  1095  1098  1104  1106  1112  1291  1304  1316  1452
        1477  1544  1555  1567  1578  1586  1638  1647  1648  1671  1673  1679  1707  1717  1730
        1749  1829
MOV      465   466   467   468   479   481   482   483   519   520   524   530   556   557   558
         559   560   561   562   566   568   571   572   573   574   575   576   577   578   579
         583   584   585   586   587   588   589   592   603   608   609   610   611   612   613
         614   615   616   620   626   627   629   634   636   638   647   648   649   650   654
         655   656   657   658   681   696   723   724   725   726   727   728   730   740   742
         743   745   746   748   752   754   757   758   759   760   770   771   781   783   784
```

```
                786   787   788   801   911   917   923   924   925   937   942   946   947   948   949
                958   962   970   974   976   977   979   980   984   992   995   998   999  1000  1001
               1006  1015  1029  1033  1039  1040  1041  1050  1051  1061  1062  1063  1064  1065  1066
               1071  1075  1083  1084  1086  1087  1089  1090  1094  1108  1118  1120  1121  1122  1126
               1127  1128  1134  1135  1144  1151  1152  1153  1154  1166  1173  1175  1176  1179  1180
               1192  1194  1199  1201  1203  1207  1208  1217  1221  1229  1235  1239  1241  1243  1255
               1257  1259  1260  1267  1268  1270  1271  1272  1275  1280  1295  1297  1302  1303  1306
               1307  1308  1310  1341  1342  1343  1344  1347  1348  1349  1352  1354  1355  1356  1357
               1358  1359  1360  1361  1362  1364  1405  1406  1407  1408  1409  1412  1413  1416  1419
               1420  1421  1423  1424  1425  1435  1436  1437  1438  1441  1442  1443  1463  1464  1465
               1469  1470  1472  1474  1475  1476  1478  1479  1490  1491  1492  1493  1494  1495  1496
               1498  1503  1504  1511  1512  1513  1514  1515  1516  1517  1518  1520  1522  1524  1526
               1535  1541  1542  1543  1552  1553  1554  1557  1564  1565  1566  1569  1575  1576  1577
               1583  1584  1585  1597  1598  1600  1603  1604  1605  1606  1615  1620  1621  1623  1644
               1657  1660  1661  1667  1670  1676  1681  1682  1683  1687  1691  1697  1704  1705  1712
               1713  1716  1726  1729  1742  1784  1785  1786  1793  1801  1817  1836  1837  1838  1839
MOVB            521   538   677   699  1645  1653  1662  1709  1710  1711  1778  1779  1781  1782  1783
               1787  1790  1791  1828
NEG            1788
NOP             673  1402  1403  1404  1422  1426  1427  1428  1429  1430  1431
RESET           780
ROL             594   595   596   597   964   965   966   967   968   969  1077  1078  1079  1080  1081
               1082  1808  1810  1812  1814  1815  1816  1818
ROR             685  1053  1054  1055  1056  1057  1058  1059
RTI             726   802
RTS             526   539   659   698   734   895   899  1505  1610  1626  1658  1706  1733  1747  1840
SOB             469   484   894   898   981  1002  1091  1125  1132  1261  1309  1414
SUB             645   646   959  1072  1129  1139  1646
SWAB           1688
TST             523   569   630   642   651   684   695   750   764   775   921   929   971   990  1037
               1045  1068  1299  1395  1399  1447  1454  1650  1702  1797  1823
TSTB            515   536   563   675  1607
XOR            1278  1497  1599  1622
.ASCII          844   845  1862  1874  1890  1918  1939  2003
.ASCIZ          843   846  1851  1854  1858  1865  1880  1896  1916  1927  1936  1946  1954  1957  1965
               1968  1973  1979  1986  1996  2007  2011  2018  2023  2028
.BLKB           540
.BYTE           510   511   512   513   773   774   814   815   817   818   927   928   986   987  1009
               1010  1018  1019  1043  1044  1096  1097  1718  1719  1752  1841  1842  1843  1844  1909
               1910  1911  1912  1913  1914  1915
.ENABL          289
.END           2033
.EQUIV          321   322   324   337   338   357   358   359   360   361   362   363   364   365   366
                385   386   387   388   389   390   391   392   393   394
.EVEN          1905  2031
.PAGE           408   448   462   491   662   718   737   803   856   891   902  1846
.REM              1
.TITLE          290
.WORD           410   660   661   717   808   809   810   811   812   813   816   819   820   821   822
                823   824   825   826   827   828   829   830   831   832   833   834   835   836   837
                838   839   840   841   842   847   848   849   850   851   852   853   854   855   901
                939   944  1300  1708  1731  1906  2032
```

```
TEST MOS MEMORY MACY11 27(732)  22-SEP-75  14:48  PAGE 56
DCMSBB.P11      CROSS REFERENCE TABLE -- PERMANENT SYMBOLS

 ERRORS DETECTED:  0
 DEFAULT GLOBALS GENERATED:  0

 #,DCMSBB.SEQ/SOL/CRF/PAGNUM/NL:TOC=DCMSBB
 RUN-TIME:  5 12 3 SECONDS
 RUN-TIME RATIO: 87/20=4.2
 CORE USED:  8K  (15 PAGES)
```