RM03

DUAL PORT LOGIC TEST PART 2

MD-11-DZRMH-A

EP-DZRMH-A-DL-A

OCT 1977

digital

FICHE 1 OF 1

MADE IN USA

```
                        .REM    ↑
```

IDENTIFICATION

PRODUCT CODE:	MAINDEC-11-DZRMH-A-D
PRODUCT NAME:	RM03 DUAL PORT LOGIC TEST, PART 2
DATE CREATED:	15 JULY 77
MAINTAINER:	DIAGNOSTIC GROUP
AUTHOR:	DOUG RIIKONEN

RM03 DUAL PORT LOGIC TEST, PART 2

CONTENTS

1. ABSTRACT

THE RM03 DUAL PORT LOGIC TEST PERFORMS A SERIES OF TESTS WHICH VERIFY THAT THE RM03 DUAL PORT LOGIC IS FUNCTIONING PROPERLY. ONLY THE CONTROL LOGIC IS TESTED BY THIS PROGRAM; DATA HANDLING IN THE DUAL PORT MODE IS NOT TESTED BY THIS PROGRAM.

BOTH PORTS OF THE DRIVE ARE CABLED TO THE SAME MASSBUS BY A SPECIAL ADAPTER CABLE. THIS ARRANGEMENT ALLOWS THE DUAL PORT LOGIC TO BE TESTED FROM ONE PDP-11/RH11 OR RH70.

THIS PROGRAM IS THE SECOND PART OF THE RM03 DUAL PORT OPTION LOGIC TEST, AND IS USED TO TEST THE "PORT SELECT" SWITCH.

2. REQUIREMENTS

2.1 EQUIPMENT

PDP-11 PROCESSOR
16K OF MEMORY
KW11-L OR KW11-P CLOCK
TELETYPE
RH11 OR RH70 WITH AN RM03
RM03 DUAL PORT TEST CABLE

2.2 PREREQUISITE PROGRAMS

A. RM03 DISKLESS DIAGNOSTIC

B. RM03 FUNCTIONAL TEST

THE PRELIMINARY PROGRAMS MUST BE RUN TWICE: ONCE FROM EACH CONTROLLER (PORT).

C. RM03 DUAL PORT LOGIC TEST
PART 1

2.3 OTHER PROGRAMS

DYNAMIC OPERATION OF THE DUAL PORT OPTION IS TESTED BY THE RM03 PERFORMANCE EXERCISER PROGRAM.

3. LOADING PROCEDURES

THE PROGRAM MAY BE LOADED BY THE ABSOLUTE PAPER TAPE LOADER OR IT MAY BE LOADED FROM THE APPROPIATE MEDIA USING THE ASSOCIATED'XXDP' LOADER. THE PROGRAM MAY NOT

BE INCLUDED IN AN 'XXDP' CHAIN.

4. STARTING PROCEDURES

4.1 STARTING ADDRESSES

A. THE NORMAL STARTING ADDRESS OF THE PROGRAM IS LOCATION 200(8). STARTING AT THIS ADDRESS ALLOWS THE OPERATOR TO SELECT (OR RESELECT) THE ADDRESS OF THE DRIVE TO BE TESTED.

B. THE RESTART ADDRESS IS LOCATION 204(8). THE PROGRAM WILL USE THE CURRENT DRIVE (DCL) ADDRESS.

C. THE PROGRAM CAN BE STARTED AT LOCATION 210(8) TO ALLOW THE RH11 OR RH70 ADDRESS TO BE CHANGED.

4.2 UNIBUS & VECTOR ADDRESS

THE PROGRAM ASSUMES THE FOLLOWING UNIBUS AND VECTOR ADDRESSES. THESE ADDRESSES MAY BE CHANGED PRIOR TO STARTING THE PROGRAM FROM ANY OF THE STARTING LOCATIONS.

MEMORY LOCATION	CONTENTS	FUNCTION
1142	177560	TTY KEYBOARD STATUS REG
1144	177562	TTY KEYBOARD BUFFER REG
1146	177564	TTY PRINTER STATUS REG
1150	177566	TTY PRINTER BUFFER REG
1210	172540	KW11-P STATUS REG
1212	172542	KW11-L COUNTER BUFFER
1214	104	KW11-P VECTOR ADDRESS
1216	177546	KW11-L STATUS REGISTER
1220	100	KW11-L VECTOR ADDRESS

4.3 OPERATOR ACTION

A. CONNECT THE DUAL PORT TEST CABLE BETWEEN BUS A & BUS B ON THE DRIVE BEING TESTED. (SEE SECTION 5.4)
B. LOAD THE PROGRAM INTO MEMORY IN THE PROCESSOR CONTROLLING THE MASSBUS USED FOR TESTING.
C. SWITCH THE 'PORT SELECT' SWITCH ON THE DRIVE TO BE TESTED TO THE 'A/B' POSITION. CYCLE THE DRIVE UP.
D. LOAD THE APPROPIATE STARTING ADDRESS (200(8) OR 210(8)) INTO THE SWITCH REGISTER (OR THE 'SOFTWARE' SWITCH REGISTER, SEE SECTION 5.2.)
E. PRESS START.
F. ENTER THE DRIVE NUMBER.
G. ENTER THE NUMBER OF THE TEST TO BE RUN. ('CARRIAGE RETURN' OR '0' WILL RUN ALL TESTS.)
H. THE PROGRAM MAY BE STOPPED AT ANY TIME AND RESTARTED

FROM LOCATION 204.

5. OPERATING PROCEDURES

5.1 OPERATIONAL SWITCH SETTINGS

WITH ALL SWITCHES SET TO ZERO, THE PROGRAM WILL TYPE
ALL ERRORS AND CONTINUE TESTING.

THE SWITCH SETTINGS ARE:

SW<15>=1...HALT ON ERROR
SW<14>=1...LOOP ON TEST
SW<13>=1...INHIBIT ERROR TYPEOUTS
SW<11>=1...INHIBIT TEST ITERATIONS
SW<10>=1...RING TTY BELL ON ERROR
SW<09>=1...LOOP ON ERROR

5.2 'SOFTWARE' SWITCH REGISTER

IF THE PROGRAM IS BEING RUN ON A SWITCHLESS PROCESSOR (I.E. AN 11/34)
THE PROGRAM WILL DETERMINE THAT THE HARDWARE SWITCH REGISTER IS
NOT PRESENT AND WILL USE A 'SOFTWARE' SWITCH REGISTER. THE
'SOFTWARE' SWITCH REGISTER IS LOCATED AT LOCATION 176 (8). THE
SETTINGS OF THE 'SOFTWARE' SWITCHES ARE CONTROLLED THROUGH A KEYBOARD
ROUTINE WHICH IS CALLED BY TYPING A 'CONTROL G'. THE PROGRAM WILL
RECOGNIZE THE 'CONTROL G' AT ANY TIME EXCEPT WHEN THE PROGRAM
IS AT A HIGHER PRIORITY PROCESSING AN RM03 INTERRUPT. THE
'SOFTWARE' SWITCH VALUES ARE ENTERED AS AN OCTAL NUMBER IN RESPONSE
TO THE PROMPT FROM THE SWITCH ENTRY ROUTINE:

'SWR = NNNNNN NEW ='

EACH TIME SWITCH SETTING ARE ENTERED, THE ENTIRE SWITCH REGISTER
IMAGE MUST BE ENTERED. LEADING ZEROS ARE NOT REQUIRED. 'RUBOUT' AND
'CONTROL U' FUNCTIONS MAY BE USED TO CORRECT TYPING ERRORS
DURING SWITCH ENTRY.

ON PROCESSORS WITH HARDWARE SWITCH REGISTERS, THE 'SOFTWARE' SWITCH
REGISTER MAY BE USED. IF THE PROGRAM FINDS ALL 16 SWITCHES IN THE
'UP' POSITION, ALL SWITCH REGISTER REFERENCES WILL BE TO THE
'SOFTWARE' REGISTER AND THE PROCEDURES DESCRIBED ABOVE MUST
BE FOLLOWED.

5.3 TEST SELECTION

INDIVIDUAL TESTS ARE SELECTED IN RESPONSE TO THE 'ENTER
TEST NUMBER:' MESSAGE. ANY VALID TEST NUMBER CAN BE
ENTERED. EACH ENTRY MUST BE TERMINATED BY A CARRIAGE
RETURN (CR). THE LOOP ON TEST SWITCH, SW<15>, MUST BE SET
TO ALL CONTINUOUS EXECUTION OF THE SELECTED TEST.

TO RUN ALL TESTS IN SEQUENCE, ENTER EITHER A '0' FOLLOWED

BY A CARRIAGE RETURN OR A CARRIAGE RETURN BY ITSELF. THE
PROGRAM WILL THEN EXECUTE ALL TESTS IN SEQUENCE.

THE 'RUBOUT KEY' (RO) CAN BE USED TO DELETE THE LAST
CHARACTER ENTERED. SUCCESSIVELY STRIKING THE RO KEY
WILL DELETE CHARACTERS UNTIL THE PREVIOUS CHARACTERS HAVE
BEEN DELETED. CHARACTERS DELETED BY THE RO KEY WILL BE
TYPED AND WILL BE SEPARATED BY '\' FROM THE CHARACTERS
ENTERED BY THE OPERATOR.

THE OPERATOR CAN DELETE AN ENTIRE ENTRY BY TYPING A
'CONTROL U' .

5.4 TEST CABLE CONNECTION

TO TEST THE RM03 DUAL PORT OPTION WITH THIS PROGRAM,
A SPECIAL TEST CABLE MUST BE USED. (THE TEST CABLE IS
P/N 7010507-02). THE TEST CABLE CONNECTS MASSBUS A & MASSBUS
B TOGETHER AT THE DRIVE BEING TESTED AND IS CONSTRUCTED SO
THAT BIT 0 OF THE MASSBUS UNIT SELECT LINES IS COMPLEMENTED.

WITH THE TEST CABLE CONNECTED TO THE DRIVE UNDER TEST,
THE DRIVE APPEARS AS TWO UNITS ON THE MASSBUS: EACH PORT
OF THE RM03 WILL RESPOND TO A DIFFERENT MASSBUS ADDRESS.

THE ADDRESS OF EACH PORT WILL DEPEND UPON THE DRIVE'S
ADDRESS PLUG.

THE PROGRAM WILL TYPEOUT THE APPARENT ADDRESSES OF BOTH
PORTS. (ONE PORT WILL HAVE THE ADDRESS OF THE DRIVE; THE
OTHER PORT WILL HAVE THE ADDRESS DEVELOPED BY THE CABLE).

```
**************************************************************
* ANY OTHER DRIVE ON THE MASSBUS WHICH HAS AN ADDRESS IN  *
* CONFLICT WITH EITHER OF THE TEST ADDRESSES MUST BE      *
* POWERED DOWN.                                           *
**************************************************************
```

THE TEST CABLE CONNECTION TO THE DRIVE UNDER TEST WILL
DEPEND ON WHICH PROCESSOR/RH11 IS TO TEST THE DRIVE. IF
THE DRIVE IS TO BE TESTED BY THE PROCESSOR ON PORT A,
THE TEST CABLE IS CONNECTED FROM 'BUS A OUT' TO 'BUS B IN'.
IF THE DRIVE IS TO BE TESTED BY THE PORT B PROCESSOR, THE
TEST CABLE IS CONNECTED FROM 'BUS B OUT' TO 'BUS A IN'.

WHEN THE DUAL PORT TEST CABLE IS CONNECTED, THE ATTENTION
BITS FOR PORTS A & B ARE ASSERTED IN THE SAME BIT POSITION
WHEN 'RMAS' (ATTENTION SUMMARY REGISTER) IS READ. THE ATTENTION
BIT POSITION IS DETERMINED BY THE ADDRESS OF THE DRIVE.
THE ATTENTION BIT THAT APPEARS FOR THE DRIVE IS THE
INCLUSIVE 'OR' OF THE PORT A & PORT B ATTENTION BITS. BECAUSE
OF THIS, THE PROGRAM LOOKS AT ONLY THE ATTENTION BIT IN
'RMDS1' (DRIVE STATUS REGISTER) TO DETERMINE THE STATE
OF THE SELECTED PORT'S ATTENTION BIT.

6. ERRORS

WHEN THE PROGRAM ENCOUNTERS AN ERROR, THE ERROR ROUTINE IS CALLED AND IF SW<13> IS NOT SET, THE ERROR MESSAGE PERTAINING TO THE ERROR WILL BE TYPED. EACH ERROR TYPEOUT WILL CONTAIN THE FOLLOWING:

A. AN ERROR MESSAGE
B. A DATA HEADER LINE
C. A DATA LINE CONTAINING:

    1. THE TEST NUMBER
    2. THE PC (PROGRAM COUNTER VALUE) WHERE THE ERROR CALL WAS MADE
    3. CONTENTS OF THE APPROPIATE REGISTERS

7. MISCELLANEOUS

7.1 RESTRICTIONS

TO RUN THIS PROGRAM, THE SYSTEM MUST HAVE EITHER A KW11-P OR A KW11-L CLOCK. ADDITIONALLY, THE RM03 UNDER TEST MUST HAVE THE DUAL PORT TEST CABLE CONNECTED.

7.2 LIMITATIONS

THIS PROGRAM DOES NOT TEST DATA TRANSFERS THROUGH EITHER PORT AND DOES NOT TEST THE DYNAMIC OPERATION OF THE DUAL PORT OPTION.

7.3 EXECUTION TIME

THE PROGRAM TAKES ABOUT 60 SECONDS PER PASS (DEPENDING ON OPERATOR INTERVENTION EFFICIENCY).

7.4 REQUIRED TESTS

IF THE PROGRAM IS BEING EXECUTED IN SINGLE TEST MODE, THE OPERATOR MUST CALL AND RUN THE FOLLOWING TESTS BEFORE OTHER TESTS ARE RUN:

A. TEST 2 AND TEST 3. THESE TESTS SET 'VV-A' AND 'VV-B' RESPECTIVLY. THESE TESTS MUST BE PERFORMED AT LEAST ONCE BEFORE TESTS 4 - 10 ARE RUN.

B. TEST 4 AND TEST 5. THESE TESTS DETERMINE AND STORE FOR LATER USE THE TIMEOUT ONE-SHOT VALUE MEASURED THROUGH EACH PORT. THESE TESTS MUST BE PERFORMED AT LEAST ONCE BEFORE TESTS 6 - 10 ARE RUN.

7.5 DISK SURFACE USAGE

THE DIAGNOSTIC DOES NOT USE THE DISK SURFACE. HOWEVER, THE DRIVE MUST BE CYCLED UP AND ON LINE FOR THE DIAGNOSTIC TO BE RUN.

7.8 LOOP ON ERROR OPTION

IF SW<09> IS SET, THE PROGRAM WILL LOOP ON A FAILING TEST UNTIL EITHER THE SWITCH IS RESET OR THE ERROR STOPS OCCURING. BECAUSE THE PROGRAM MUST RESET THE RM03 TO A KNOWN STATE BEFORE LOOPING ON THE ERROR, THE TEST FOR SW<09> IS PERFORMED AT THE END OF THE TEST - NOT AT THE POINT WHERE THE ERROR WAS DETECTED.

8. TEST DESCRIPTIONS

8.1 METHOD USED TO VERIFY THAT DRIVE IS IN NEUTRAL

THE PROGRAM DETERMINES THE THE DRIVE IS IN NEUTRAL BY CHECKING THE CONTENTS OF THE DRIVE STATUS REGISTER (RMDS1) THROUGH BOTH PORTS. THE PROGRAM MASKS OUT THE PORT DEPENDENT BITS ('ATA' & 'VV') AND VERIFIES THAT CORRECT STATUS IS READ THROUGH BOTH PORTS. (THE CORRECT STATUS IS 'MOL', 'PGM', 'DPR', & 'DRY'.) IF NEITHER PORT SEES ALL ZEROS FROM RMDS1, THE PROGRAM CONCLUDES THAT THE DRIVE IS IN NEUTRAL AND THAT ANY BIT DISCREPANCY BETWEEN PORTS INDICATES A FAILURE IN THE PATH FOR THAT BIT.

8.2 METHOD USED TO VERIFY THAT THE DRIVE HAS BEEN SEIZED

THE PROGRAM VERIFIES THAT THE DRIVE HAS BEEN SEIZED BY CHECKING THE DRIVE STATUS REGISTER (RMDS1) THROUGH THE SEIZING PORT AND VERIFING THAT CORRECT STATUS IS SEEN. WHEN RMDS1 IS READ THROUGH THE OPPOSITE PORT, ZEROS SHOULD BE SEEN. IF BOTH CONDITIONS EXIST, (I.E., THE OPPOSITE PORT), THE PROGRAM CONCLUDES THAT THE DRIVE HAS BEEN SEIZED BY THE SPECIFIED PORT.

TEST 1 DRIVE ACCESS TEST

VERIFY THAT THE DRIVE CAN BE ACCESSED THROUGH BOTH PORTS

A. SELECT DRIVE, VERIFY THAT THE DRIVE IS PRESENT, THAT THE DRIVE IS A DUAL PORT RM03, THAT THE DRIVE IS ONLINE (RMDS1 HAS 'MOL' 'PGM' 'DPR' & 'DRY' BITS SET), AND THE THE DRIVE SERIAL NUMBER READ THROUGH BOTH PORTS IS THE SAME.

B. THE TEST IS REPEATED THROUGH BOTH PORTS.

```
                        TEST 2  SET 'VV' FOR PORT A

                        SET VOLUME VALID

                          A.  ISSUE A DRIVE CLEAR COMMAND THROUGH PORT A.

                          B.  ISSUE A READIN PRESET COMMAND THROUGH PORT A.  VERIFY
                                THAT THE 'VV' BIT IS SET FOR PORT A.

                          C.  ISSUE A RELEASE COMMAND THROUGH PORT A.  VERIFY THAT
                                THE DRIVE RETURNED TO NEUTRAL AND THAT NEITHER ATTENTION
                                BIT IS SET.


                        TEST 3  SET 'VV' FOR PORT B

                        SET VOLUME VALID

                          A.  ISSUE A DRIVE CLEAR COMMAND THROUGH PORT B.

                          B.  ISSUE A READIN PRESET COMMAND THROUGH PORT B.  VERIFY
                                THAT THE 'VV' BIT IS SET FOR PORT B.

                          C.  ISSUE A RELEASE COMMAND THROUGH PORT B.  VERIFY THAT
                                THE DRIVE RETURNED TO NEUTRAL AND THAT NEITHER ATTENTION
                                BIT IS SET.


                        TEST 4  MEASURE THE TIMEOUT ONE-SHOT THROUGH PORT A

                        MEASURE THE TIMEOUT ONE-SHOT VALUE THROUGH PORT A

                          A.  WRITE 0'S INTO RMDS1 THROUGH PORT A AND VERIFY THAT THE
                                DRIVE HAS BEEN SEIZED.

                          B.  WAIT FOR TIMEOUT TO OCCUR.  MEASURE THE DURATION OF THE TIMEOUT
                                ONE-SHOT AND SAVE THE VALUE FOR LATER USE.


                          C.  VERIFY THAT THE TIMEOUT OCCURRED AND THAT THE DRIVE RETURNS
                                TO NEUTRAL


                        TEST 5  MEASURE THE TIMEOUT ONE-SHOT THROUGH PORT B

                        MEASURE THE TIMEOUT ONE-SHOT VALUE THROUGH PORT B

                          A.  WRITE 0'S INTO RMDS1 THROUGH PORT B AND VERIFY THAT THE
                                DRIVE HAS BEEN SEIZED.
```

```
                        B.  WAIT FOR TIMEOUT TO OCCUR.  MEASURE THE DURATION OF THE TIMEOUT
                            ONE-SHOT AND SAVE THE VALUE FOR LATER USE.

                        C.  VERIFY THAT THE TIMEOUT OCCURRED AND THAT THE DRIVE RETURNS
                            TO NEUTRAL


                    TEST 6  TEST 'CONTROLLER SELECT' SWITCH, DRIVE CYCLED UP

                    TEST THE OPERATION OF THE 'CONTROLLER SELECT' SWITCH (DRIVE CYCLED UP).

                        A.    SWITCH TO CONTROLLER 'A' POSITION.  VERIFY THAT THE DRIVE IS IN
                              NEUTRAL AND THAT THE STATUS BITS IN RMDS1, AS READ THROUGH BOTH
                              PORTS, ARE CORRECT.

                        B.    SWITCH TO CONTROLLER 'B' POSITION. VERIFY THAT THE DRIVE IS IN
                              NEUTRAL AND THAT THE STATUS BITS IN RMDS1, AS READ THROUGH BOTH
                              PORTS, ARE CORRECT.

                        C.  RETURN THE 'CONTROLLER SELECT' SWITCH TO THE 'A/B' POSITION.  VERIFY
                            THE DRIVE STATE.


                    TEST 7  TEST 'CONTROLLER SELECT' SWITCH LOCKED ON PORT A

                    TEST THE OPERATION OF THE 'CONTROLLER SELECT' SWITCH (DRIVE CYCLED DOWN).

                        A.    CYCLE THE DRIVE DOWN.

                        B.    SWITCH TO CONTROLLER A POSITION.  VERIFY THAT THE DRIVE IS IN
                              NEUTRAL AND THAT THE STATUS BITS IN RMDS1, AS READ THROUGH BOTH
                              PORTS, ARE CORRECT.

                        C.  SWITCH THE 'CONTROLLER SELECT' SWITCH TO A; CYCLE THE DRIVE UP.

                        D.    WHEN THE DRIVE CYCLES UP, VERIFY THAT 'VV-A IS RESET, AND
                              THAT 'ATA-A IS SET.


                        E.    ISSUE A DRIVE CLEAR COMMAND AND A READIN PRESET COMMAND THROUGH
                              PORT A.

                        F.    VERIFY THAT THE DRIVE CANNOT BE ACCESSED THROUGH PORT B AND
                              'NED' SETS WHEN ATEMPTING TO ACCESS THE DRIVE THROUGH
                              PORT B.  ATTEMPT TO SET PORT REQUEST BY WRITING 0'S
                              INTO RMDS1 THROUGH PORT B.

                        G.    ISSUE A RELEASE COMMAND THROUGH PORT A.  VERIFY THAT THE
                              DRIVE REMAINS LOCKED ON PORT A.
```

```
                        TEST 10 TEST 'CONTROLLER SELECT' SWITCH LOCKED ON PORT B

                        TEST THE OPERATION OF THE 'CONTROLLER SELECT' SWITCH (DRIVE CYCLED DOWN).

                          A.    CYCLE THE DRIVE DOWN.

                          B.    SWITCH TO CONTROLLER B POSITION.  VERIFY THAT THE DRIVE IS IN
                                NEUTRAL AND THAT THE STATUS BITS IN RMDS1, AS READ THROUGH BOTH
                                PORTS, ARE CORRECT.

                          C.  SWITCH THE 'CONTROLLER SELECT' SWITCH TO B; CYCLE THE DRIVE UP.

                          D.    WHEN THE DRIVE CYCLES UP, VERIFY THAT 'VV-B IS RESET, AND
                                THAT 'ATA-B IS SET.

                          E.    ISSUE A DRIVE CLEAR COMMAND AND A READIN PRESET COMMAND THROUGH
                                PORT B.

                          F.    VERIFY THAT THE DRIVE CANNOT BE ACCESSED THROUGH PORT A AND
                                'NED' SETS WHEN ATEMPTING TO ACCESS THE DRIVE THROUGH
                                PORT A.  ATTEMPT TO SET PORT REQUEST BY WRITING 0'S
                                INTO RMDS1 THROUGH PORT A.

                          G.    ISSUE A RELEASE COMMAND THROUGH PORT B.  VERIFY THAT THE

                                DRIVE REMAINS LOCKED ON PORT B.

                          H.    CYCLE THE DRIVE DOWN.  CHANGE THE 'CONTROLLER SELECT' SWITCH TO
                                A/B; CYCLE THE DRIVE UP.

                          I.  VERIFY THAT BOTH PORTS CAN ACCESS THE DRIVE, THAT BOTH ATTENTION
                                BITS ARE SET, AND THAT BOTH 'VV' BITS ARE RESET.
```

```
                                ;PROGRAM REVISION #001



                                .TITLE  MD-11-DZRMH-A, RM03 DUAL CONTRLR LOGIC TEST - PART 2
                                ;*COPYRIGHT (C) 1977
                                ;*DIGITAL EQUIPMENT CORP.
                                ;*MAYNARD, MASS. 01754
                                ;*
                                ;*PROGRAM BY D. RIIKONEN
                                ;*
                                ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                                ;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
                                ;*

                                .SBTTL  OPERATIONAL SWITCH SETTINGS
                                ;*
                                ;*      SWITCH                  USE
                                ;*      ------          --------------------
                                ;*        15            HALT ON ERROR
                                ;*        14            LOOP ON TEST
                                ;*        13            INHIBIT ERROR TYPEOUTS
                                ;*        11            INHIBIT ITERATIONS
                                ;*        10            BELL ON ERROR
                                ;*         9            LOOP ON ERROR



                                .SBTTL  BASIC DEFINITIONS

                                ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
001100                          STACK=  1100
                                .EQUIV  EMT,ERROR       ;;BASIC DEFINITION OF ERROR CALL
                                .EQUIV  IOT,SCOPE       ;;BASIC DEFINITION OF SCOPE CALL

                                ;*MISCELLANEOUS DEFINITIONS
000011                          HT=     11              ;;CODE FOR HORIZONTAL TAB
000012                          LF=     12              ;;CODE FOR LINE FEED
000015                          CR=     15              ;;CODE FOR CARRIAGE RETURN
000200                          CRLF=   200             ;;CODE FOR CARRIAGE RETURN-LINE FEED
177776                          PS=     177776          ;;PROCESSOR STATUS WORD
                                .EQUIV  PS,PSW
177774                          STKLMT= 177774          ;;STACK LIMIT REGISTER
177772                          PIRQ=   177772          ;;PROGRAM INTERRUPT REQUEST REGISTER
177570                          DSWR=   177570          ;;HARDWARE SWITCH REGISTER
177570                          DDISP=  177570          ;;HARDWARE DISPLAY REGISTER

                                ;*GENERAL PURPOSE REGISTER DEFINITIONS
000000                          R0=     %0              ;;GENERAL REGISTER
000001                          R1=     %1              ;;GENERAL REGISTER
000002                          R2=     %2              ;;GENERAL REGISTER
000003                          R3=     %3              ;;GENERAL REGISTER
000004                          R4=     %4              ;;GENERAL REGISTER
000005                          R5=     %5              ;;GENERAL REGISTER
000006                          R6=     %6              ;;GENERAL REGISTER
000007                          R7=     %7              ;;GENERAL REGISTER
000006                          SP=     %6              ;;STACK POINTER
```

```
                        PC=     %7              ;;PROGRAM COUNTER

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

                        ;*"SWITCH REGISTER" SWITCH DEFINITIONS
100000                  SW15=   100000
040000                  SW14=   40000
020000                  SW13=   20000
010000                  SW12=   10000
004000                  SW11=   4000
002000                  SW10=   2000
001000                  SW09=   1000
000400                  SW08=   400
000200                  SW07=   200
000100                  SW06=   100
000040                  SW05=   40
000020                  SW04=   20
000010                  SW03=   10
000004                  SW02=   4
000002                  SW01=   2
000001                  SW00=   1
                        .EQUIV  SW09,SW9
                        .EQUIV  SW08,SW8
                        .EQUIV  SW07,SW7
                        .EQUIV  SW06,SW6
                        .EQUIV  SW05,SW5
                        .EQUIV  SW04,SW4
                        .EQUIV  SW03,SW3
                        .EQUIV  SW02,SW2
                        .EQUIV  SW01,SW1
                        .EQUIV  SW00,SW0

                        ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
100000                  BIT15=  100000
040000                  BIT14=  40000
020000                  BIT13=  20000
010000                  BIT12=  10000
004000                  BIT11=  4000
002000                  BIT10=  2000
001000                  BIT09=  1000
000400                  BIT08=  400
000200                  BIT07=  200
000100                  BIT06=  100
000040                  BIT05=  40
000020                  BIT04=  20
000010                  BIT03=  10
000004                  BIT02=  4
000002                  BIT01=  2
```

```
                        BIT00=  1
                        .EQUIV  BIT09,BIT9
                        .EQUIV  BIT08,BIT8
                        .EQUIV  BIT07,BIT7
                        .EQUIV  BIT06,BIT6
                        .EQUIV  BIT05,BIT5
                        .EQUIV  BIT04,BIT4
                        .EQUIV  BIT03,BIT3
                        .EQUIV  BIT02,BIT2
                        .EQUIV  BIT01,BIT1
                        .EQUIV  BIT00,BIT0

                        ;*BASIC "CPU" TRAP VECTOR ADDRESSES
000004                  ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
000010                  RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
000014                  TBITVEC=14              ;;"T" BIT
000014                  TRTVEC= 14              ;;TRACE TRAP
000014                  BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
000020                  IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
000024                  PWRVEC= 24              ;;POWER FAIL
000030                  EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
000034                  TRAPVEC=34              ;;"TRAP" TRAP
000060                  TKVEC=  60              ;;TTY KEYBOARD VECTOR
000064                  TPVEC=  64              ;;TTY PRINTER VECTOR
000240                  PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR

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

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

                        .SBTTL  RH11 REGISTERS

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

                        ;CONTROL AND STATUS REGISTER 1 (RMCS1)

000100                  IE=     100             ;INTERRUPT ENABLE (BIT #6)
000200                  RDY=    200             ;READY (BIT #7)
000400                  A16=    400             ;HIGH ORDER BUS ADDRESS BIT (BIT #8)
001000                  A17=    1000            ;HIGH ORDER BUS ADDRESS BIT (BIT #9)
002000                  PSEL=   2000            ;PORT SELECT (BIT #10)
020000                  MCPE=   20000           ;MASSBUSS PARITY ERROR (BIT #13)
040000                  TRE=    40000           ;TRANSFER ERROR (BIT #14)
100000                  SC=     100000          ;SPECIAL CONDITION (BIT #15)

                        ;WORD COUNT REGISTER (RMWC)
                        ;(EACH BIT IS CALLED BY BIT NUMBER)

                        ;BUS ADDRESS REGISTER (RMBA)
                        ;(EACH BIT IS CALLED BY BIT NUMBER)

                        ;CONTROL AND STATUS REGISTER 2 (RMCS2)

000001                  U0=     1               ;UNIT SELECT (BIT #0)
000002                  U1=     2               ;UNIT SELECT (BIT #1)
000004                  U3=     4               ;UNIT SELECT (BIT #2)
```

```
                        BAI=    10              ;BUS ADDRESS INCREMENT INHIBIT (BIT #3)
                        PAT=    20              ;MASSBUS PARITY TEST (BIT #4)
                        CLR=    40              ;CLEAR (BIT #5)
                        IR=     100             ;INPUT READY (BIT #6)
                        OR=     200             ;OUTPUT READY (BIT #7)
                        MDPE=   400             ;MASS BUS PARITY ERROR (BIT #8)
                        MXF=    1000            ;MISSED TRANSFER ERROR (BIT #9)
                        PGE=    2000            ;PROGRAM ERROR (BIT #10)
                        NEM=    4000            ;NON EXISTENT MEMORY (BIT #11)
                        NED=    10000           ;NON EXISTENT DRIVE (BIT #12)
                        UPE=    20000           ;UNIBUS PARITY ERROR (BIT #13)
                        WCE=    40000           ;WRITE CHECK ERROR (BIT #14)
                        DLT=    100000          ;DATA LATE (BIT #15)

                        ;DATA BUFFER REGISTER (RMDB)
                        ;(EACH BIT IS CALLED BY BIT NUMBER)


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

                        .SBTTL  RM03 REGISTERS

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

                        ;CONTROL AND STATUS 1 REGISTER. (#00)

                        GO=     1               ;GO BIT (BIT #0)
                        FO=     2               ;FUNCTION CODE BIT #1
                        F1=     4               ;FUNCTION CODE BIT #2
                        F2=     10              ;FUNCTION CODE BIT #3
                        F3=     20              ;FUNCTION CODE BIT #4
                        F4=     40              ;FUNCTION CODE BIT #5
                        DVA=    4000            ;DEVICE AVAILABLE (BIT #11)

                        ;DRIVE STATUS REGISTER (RMDS1) (#01)

                        ;DF5=   1               DRIVE FORWARD 5"/SEC.  (BIT #0)
                        DFF20=  2               ;DRIVE FORWARD 20"/SEC.  (BIT #1)
                        DIGB=   4               ;DRIVE TO INNER GUARD BAND (BIT #2)
                        GRV=    10              ;GO REVERSE (BIT #3)
                        DL64=   20              ;DIFFERENCE LESS THAN 64 (BIT #4)
                        DE1=    40              ;DIFFERENCE EQUALS 1 (BIT #5)
                        VV=     100             ;VOLUME VALID (BIT #6)
                        DRY=    200             ;DRIVE READY (BIT #7)
                        DPR=    400             ;DRIVE PRESENT (BIT #8)
                        PGM=    1000            ;PROGRAMABLE (BIT #9)
                        LBT=    2000            ;LAST SECTOR TRANSFERRED (BIT #10)
                        WRL=    4000            ;WRITE LOCK (BIT #11)
                        MOL=    10000           ;MEDIUM ON-LINE (BIT #12)
                        PIP=    20000           ;POSITIONING OPERATION IN PROGRESS (BIT #13)
                        ERR=    40000           ;COMPOSITE ERROR (BIT #14)
                        ATA=    100000          ;ATTENTION ACTIVE (BIT #15)

                        ;ERROR REGISTER #01 (RMER1) (#02)

                        ILF=    1               ;ILLEGAL FUNCTION (BIT #0)
```

```
                        ILR=    2               ;ILLEGAL REGISTER (BIT #1)
                        RMR=    4               ;REGISTER MODIFICATION REFUSED (BIT #2)
                        PAR=    10              ;PARITY ERROR (BIT #3)
                        FER=    20              ;FORMAT ERROR (BIT #4)
                        WCF=    40              ;WRITE CLOCK FAIL (BIT #5)
                        ECH=    100             ;ECC HARD ERROR (BIT #6)
                        HCE=    200             ;HEADER COMPARE ERROR (BIT #7)
                        HCRC=   400             ;HEADER CRC ERROR (BIT #8)
                        AOE=    1000            ;ADDRESS OVERFLOW ERROR (BIT #9)
                        IAE=    2000            ;INVALID ADDRESS ERROR (BIT #10)
                        WLE=    4000            ;WRITE LOCK ERROR (BIT #11)
                        DTE=    10000           ;DRIVE TIMING ERROR (BIT #12)
                        OPI=    20000           ;OPERATION INCOMPLETE (BIT #13)
                        UNS=    40000           ;DRIVE UNSAFE (BIT #14)
                        DCK=    100000          ;DATA CHECK ERROR (BIT 15)

                        ;MAINTAINABILITY REGISTER (RMMR1)(#03)

000001                  DMD=    1               ;DIAGINOSTIC MODE (BIT #0)

                        ;ATTENTION SUMMARY PSEUDO-REGISTER (RMAS) (#04)

000001                  AT0=    1               ;DEVICE 0 (BIT #0)
000002                  AT1=    2               ;DEVICE 1 (BIT #1)
000004                  AT2=    4               ;DEVICE 2 (BIT #2)
000010                  AT3=    10              ;DEVICE 3 (BIT #3)
000020                  AT4=    20              ;DEVICE 4 (BIT #4)
000040                  AT5=    40              ;DEVICE 5 (BIT #5)
000100                  AT6=    100             ;DEVICE 6 (BIT #6)
000200                  AT7=    200             ;DEVICE 7 (BIT #7)

                        ;DESIRED SECTOR/TRACK ADDRESS REGISTER (RMDA) (#05)
                        ;(EACH BIT IS CALLED BY BIT NUMBER)

                        ;DRIVE TYPE REGISTER (RMDT) (#06)

000001                  DT00=   1               ;DRIVE TYPE NUMBER BIT 1
000002                  DT01=   2               ;DRIVE TYPE NUMBER BIT 2
000004                  DT02=   4               ;DRIVE TYPE NUMBER BIT 3
000010                  DT03=   10              ;DRIVE TYPE NUMBER BIT 4
000020                  DT04=   20              ;DRIVE TYPE NUMBER BIT 5
000040                  DT05=   40              ;DRIVE TYPE NUMBER BIT 6
000100                  DT06=   100             ;DRIVE TYPE NUMBER BIT 7
000200                  DT07=   200             ;DRIVE TYPE NUMBER BIT 8
000400                  DT08=   400             ;DRIVE TYPE NUMBER BIT 9
004000                  DRQ=    4000            ;DRIVE REQUEST REQUIRED (BIT #11)
020000                  MOH=    20000           ;MOVING HEAD (BIT #13)
040000                  TAP=    40000           ;TAPE DRIVE (BIT #14)
100000                  NBA=    100000          ;NOT BLOCK ADDRESSED (BIT #15)

                        ;LOOK-AHEAD REGISTER (RMLA) (#07)

000100                  SC0=    100             ;SECTOR COUNT FIELD 0 (BIT #6)
000200                  SC1=    200             ;SECTOR COUNT FIELD 1 (BIT #7)
000400                  SC2=    400             ;SECTOR COUNT FIELD 2 (BIT #8)
001000                  SC3=    1000            ;SECTOR COUNT FIELD 3 (BIT #9)
```

```
                        SC4=    2000                    ;SECTOR COUNT FIELD 4 (BIT #10)

                        ;RM03 ERROR REGISTER #2 (RMER2) (#10)

000010                  DPE=    10                      ;DATA PARITY ERROR  (BIT #3)
000200                  DVC=    200                     ;DEVICE CHECK  (BIT #7)
002000                  LBC=    2000                    ;LOSS OF BIT CLOCK (BIT #10)
004000                  LSC=    4000                    ;LOSS OF SYSTEM CLOCK (BIT #11)
010000                  IVC=    10000                   ;INVALID COMMAND (BIT #12)
020000                  OPE=    20000                   ;OPERATOR ERROR  (BIT #13)
100000                  SKI=    100000                  ;SEEK INCOMPLETE (BIT #14)

                        ;OFFSET REGISTER (RMOF) (#11)

000200                  OFD=    200                     ;OFFSET FORWARD (BIT #5)
002000                  HCI=    2000                    ;HEADER COMPARE INHIBIT (BIT #10)
004000                  ECI=    4000                    ;ERROR CORRECTION CODE INHIBIT (BIT #11)
010000                  FMT16=  10000                   ;FORMAT BIT (BIT #12

                        ;DESIRED CYLINDER ADDRESS (RMDC) (#12)
                        ;(EACH BIT IS CALLED BY BIT NUMBER)

                        ;SERIAL NUMBER REGISTER (RMSN) (#14)
                        ;(EACH IS CALLED BY BIT NUMBER)

                        ;ECC POSITION REGISTER (RMEC1) (#16)
                        ;(EACH BIT IS CALLED BY BIT NUMBER)

                        ;ECC PATTERN REGISTER (RMEC2) (#17)
                        ;(EACH BIT IS CALLED BY BIT NUMBER)

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

                        .SBTTL  DEFINITIONS OF THE RH11/RM03 ADDRESS INDEXES

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

000000          RMCS1=0                 ;CONTROL AND STATUS REGISTER #1 (DRIVE REG. 00)
000002          RMWC=2                  ;WORD COUNT REGISTER (NOT A DRIVE REG)
000004          RMBA=4                  ;UNIBUS ADDRESS REGISTER (NOT A DRIVE REG)
000006          RMDA=6                  ;DESIRED SECTOR/TRACK ADDRESS REGISTER (DRIVE REG. 05)
000010          RMCS2=10                ;CONTROL AND STATUS REGISTER #2 (NOT A DRIVE REG)
000012          RMDS1=12                ;DRIVE STATUS REGISTER (DRIVE REG 01)
000014          RMER1=14                ;ERROR REGISTER #1 (DRIVE REG. 02)
000016          RMAS=16                 ;ATTENTION SUMMARY PSEUDO REGISTER (DRIVE REG. 04)
000020          RMLA=20                 ;LOOK AHEAD REGISTER (DRIVE REG. 07)
000022          RMDB=22                 ;DATA BUFFER REGISTER (NOT A DRIVE REG.)
000024          RMMR1=24                        ;MAINTAINABILITY REGISTER (DRIVE REG. 03)
000026          RMDT=26                 ;DRIVE TYPE REGISTER (DRIVE REG. 06)
000030          RMSN=30                 ;SERIAL NUMBER REGISTER (DRIVE REG. 10)
000032          RMOF=32                 ;OFFSET REGISTER (DRIVE REG. 11)
000034          RMDC=34                 ;DESIRED CYLINDER ADDRESS REGISTER (DRIVE REG. 12)
000040          RMMR2=40                ; MAINTENANCE REGISTER #2 (DRIVE REG. 14)
000042          RMER2=42                ;ERROR REGISTER #2 (DRIVE REG. 15)
000044          RMEC1=44                ;ECC POSITION REGISTER (DRIVE REG. 16)
000046          RMEC2=46                ;ECC PATTERN REGISTER (DRIVE REG.  17)
```

```
                                .SBTTL  TRAP CATCHER

        000000                          .=0
                                ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
                                ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
                                ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
        000174                          .=174
000174  000000                 DISPREG: .WORD  0              ;;SOFTWARE DISPLAY REGISTER
000176  000000                 SWREG:   .WORD  0              ;;SOFTWARE SWITCH REGISTER

                                .SBTTL  ACT11 HOOKS

                                ;;*****************************************************************
                                ;HOOKS REQUIRED BY ACT11
        000200                          $SVPC=.                ;SAVE PC
        000046                          .=46
000046  013206                          $ENDAD                 ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
        000052                          .=52
000052  020000                          .WORD   20000          ;;2)SET LOC.52 TO 20000
        000200                          .=$SVPC                ;; RESTORE PC

                                .SBTTL  STARTING ADDRESS = 200

000200  000137  001706                  JMP     START          ;START THE PROGRAM

                                .SBTTL  START THE PROGRAM AND CHANGE THE RH11 ADDRESS = 204

000204  000137  001714                  JMP     START1         ;START AND CHANGE THE RH11 ADDRESS
```

```
                                .SBTTL  COMMON TAGS

                                ;;**********************************************************************
                                ;#THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
                                ;#USED IN THE PROGRAM.

                        001100          .=1100
001100                  $CMTAG:                         ;;START OF COMMON TAGS
001100  000000          $PASS:  .WORD   0               ;;CONTAINS PASS COUNT
001102  000             $TSTNM: .BYTE   0               ;;CONTAINS THE TEST NUMBER
001103  000             $ERFLG: .BYTE   0               ;;CONTAINS ERROR FLAG
001104  000000          $ICNT:  .WORD   0               ;;CONTAINS SUBTEST ITERATION COUNT
001106  000000          $LPADR: .WORD   0               ;;CONTAINS SCOPE LOOP ADDRESS
001110  000000          $LPERR: .WORD   0               ;;CONTAINS SCOPE RETURN FOR ERRORS
001112  000000          $ERTTL: .WORD   0               ;;CONTAINS TOTAL ERRORS DETECTED
001114  000             $ITEMB: .BYTE   0               ;;CONTAINS ITEM CONTROL BYTE
001115  001             $ERMAX: .BYTE   1               ;;CONTAINS MAX. ERRORS PER TEST
001116  000000          $ERRPC: .WORD   0               ;;CONTAINS PC OF LAST ERROR INSTRUCTION
001120  000000          $GDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'GOOD' DATA
001122  000000          $BDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'BAD' DATA
001124  000000          $GDDAT: .WORD   0               ;;CONTAINS 'GOOD' DATA
001126  000000          $BDDAT: .WORD   0               ;;CONTAINS 'BAD' DATA
001130  000000                  .WORD   0               ;;RESERVED--NOT TO BE USED
001132  000000                  .WORD   0
001134  000             $AUTOB: .BYTE   0               ;;AUTOMATIC MODE INDICATOR
001135  000             $INTAG: .BYTE   0               ;;INTERRUPT MODE INDICATOR
001136  000000                  .WORD   0
001140  177570          SWR:    .WORD   DSWR            ;;ADDRESS OF SWITCH REGISTER
001142  177570          DISPLAY: .WORD  DDISP           ;;ADDRESS OF DISPLAY REGISTER
                        $TKS:   177560                  ;;TTY KBD STATUS
                        $TKB:   177562                  ;;TTY KBD BUFFER
                        $TPS:   177564                  ;;TTY PRINTER STATUS REG. ADDRESS
                        $TPB:   177566                  ;;TTY PRINTER BUFFER REG. ADDRESS
001154  000             $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER FOR FILLS
001155  002             $FILLS: .BYTE   2               ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
001156  012             $FILLC: .BYTE   12              ;;INSERT FILL CHARS. AFTER A "LINE FEED"
001157  000             $TPFLG: .BYTE   0               ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
001160  000000          $REGAD: .WORD   0               ;;CONTAINS THE ADDRESS FROM
                                                        ;;WHICH  ($REGO) WAS OBTAINED
001162  000000          $REGO:  .WORD   0               ;;CONTAINS (($REGAD)+0)
001164  000000          $TMP0:  .WORD   0               ;;USER DEFINED
001166  000000          $TMP1:  .WORD   0               ;;USER DEFINED
001170  000000          $TMP2:  .WORD   0               ;;USER DEFINED
001172  000000          $TMP3:  .WORD   0               ;;USER DEFINED
001174  000000          $TMP4:  .WORD   0               ;;USER DEFINED
001176  000000          $TIMES: 0                       ;;MAX. NUMBER OF ITERATIONS
001200  000000          $ESCAPE:0                       ;;ESCAPE ON ERROR ADDRESS
001202  177607  000377  $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
001206  077             $QUES:  .ASCII  /?/             ;;QUESTION MARK
001207  015             $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
001210  000012          $LF:    .ASCIZ  <12>            ;;LINE FEED
                        ;;**********************************************************************
                        CR      =       15
                        LF      =       12
001212  172540          $LKCSR: .WORD   172540          ;ADDR OF KW11-P STATUS REGISTER
001214  172542          $LKCSB: .WORD   172542          ;ADDR OF KW11-P COUNTER BUFFER
```

```
1002  001216  000104  SLPVEC: .WORD   104          ;ADDR OF KW11-P VECTOR
1003  001220  177546  SLKS:   .WORD   177546       ;ADDR OF KW11-L STATUS REGISTER
1004  001222  000100  SLLVEC: .WORD   100          ;ADDR OF KW11-L VECTOR
1005  001224  000000  PORTA:  .WORD   0            ;ADDRESS OF PORT A
1006  001226  000000  PORTB:  .WORD   0            ;ADDRESS OF PORT B
1007  001230  000000  PORTC:  .WORD   0            ;ADDRESS OF DIFFERENT DRIVE
1008  001232  000000  ASR1:   .WORD   0            ;ATA-A OR ATA-B = 1
1009  001234  000000  PTNBR:  .WORD   0            ;CONTAINS THE PORT ADDRESS FOR ERROR TYPEOUTS
1010  001236  000000  SEIZPT: .WORD   0            ;CONTAINS THE ADDRESS OF THE SEIZING PORT
1011  001240  000000  OPPRT:  .WORD   0            ;CONTAINS THE ADDRESS OF THE 'OPPOSITE' PORT
1012  001242  000000  TSTNUM: .WORD   0            ;NUMBER OF THE CURRENT TEST
1013  001244  000000  CKERR:  .WORD   0            ;IF -1, A REGISTER MISCOMPARISON OCCURRED
1014  001246  000000  NOSEIZ: .WORD   0            ;IF -1, THE PORT IN 'SEIZPT' DID NOT SEIZE THE DRIVE
1015  001250  000000  RELERR: .WORD   0            ;IF -1, THE PORT IN 'SEIZPT' DID NOT RELEASE THE DRIVE
1016  001252  000000  TIME:   .WORD   0            ;ELAPSED TIME COUNTER
1017  001254  000000  WATCH:  .WORD   0            ;WATCH DOG TIMER LOCATION
1018  001256  000000  TIMEA:  .WORD   0            ;THE TIMEOUT ONE-SHOT VALUE MEASURED THROUGH PORT A
1019  001260  000000  TIMEAP: .WORD   0            ;PORT A TIMEOUT VALUE + 25%
1020  001262  000000  TIMEB:  .WORD   0            ;THE TIMEOUT ONE-SHOT VALUE MEASURED THROUGH PORT B
1021  001264  000000  TIMEBP: .WORD   0            ;PORT B TIMEOUT VALUE + 25%
1022  001266  000000  KYBCTL: .WORD   0            ;SINGLE TEST INDICATOR
1023  001270  000000  CHGADR: .WORD   0            ;CHANGE THE RH11 ADDRESS INDICATOR
1024
1025                  ;;*****************************************************************
1026
1027                  .SBTTL  RH11/RM03 UNIBUS AND VECTOR ADDRESSES
1028
1029                  ;;*****************************************************************
1030
1031  001272  176700  $RMADR: .WORD   176700       ;RH11/RM03 UNIBUS ADDRESS
1032  001274  000254  $RMVEC: .WORD   254          ;RH11 INTERRUPT VECTOR ADDRESS
1033
```

```
                                .SBTTL  ERROR POINTER TABLE

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

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

001276                          $ERRTB:

                                ;ERROR 1

001276  017735                          EM1     ;DRIVE IS NON-EXISTENT ('NED' BIT SET)
001300  022511                          DH1
001302  024072                          DT1
001304  024312                          DF1

                                ;ERROR 2

001306  020003                          EM2     ;WRONG DRIVE TYPE
001310  022562                          DH2
001312  024106                          DT2
001314  024317                          DF2

                                ;ERROR 3

001316  020024                          EM3     ;CONTROLLER SELECT SWITCH ON DRIVE NOT IN 'A/B'
001320  022511                          DH1
001322  024072                          DT1
001324  024312                          DF1

                                ;ERROR 4

001326  020103                          EM4     ;DRIVE NOT ON LINE
001330  022562                          DH2
001332  024106                          DT2
001334  024317                          DF2

                                ;ERROR 5

001336  020125                          EM5     ;SERIAL NUMBER READ THROUGH EACH PORT NOT THE SAME
001340  022636                          DH5
001342  024124                          DT5
001344  024325                          DF5

                                ;ERROR 6

001346  020207                          EM6     ;TIMEOUT HAS NOT OCCURRED WITHIN 2 SECONDS
001350  022705                          DH6
```

```
1090  001352  024140                    DT6
1091  001354  024332                    DF6
1092
1093                                ;ERROR 7
1094
1095  001356  020261                    EM7     ;TIMEOUT ONE-SHOT IS LESS THAN 500 MS
1096  001360  022734                    DH7
1097  001362  024150                    DT7
1098  001364  024335                    DF7
1099
1100                                ;ERROR 10
1101
1102  001366  020326                    EM10    ;READIN PRESET DOES NOT SET VOLUME VALID FOR THE PORT
1103  001370  022511                    DH1
1104  001372  024072                    DT1
1105  001374  024312                    DF1
1106
1107                                ;ERROR 11
1108
1109  001376  020413                    EM11    ;'GO' BIT RESET DURING UNLOAD COMMAND
1110  001400  022511                    DH1
1111  001402  024072                    DT1
1112  001404  024312                    DF1
1113
1114                                ;ERROR 12
1115
1116  001406  020460                    EM12    ;INCORRECT STATUS DURING UNLOAD COMMAND
1117  001410  022511                    DH1
1118  001412  024072                    DT1
1119  001414  024312                    DF1
1120
1121                                ;ERROR 13
1122
1123  001416  020527                    EM13    ;DRIVE DID NOT RETURN TO NEUTRAL AFTER UNLOAD COMMAND
1124  001420  023001                    DH13
1125  001422  024162                    DT13
1126  001424  024332                    DF6
1127
1128                                ;ERROR 14
1129
1130  001426  020614                    EM14    ;ATTENTION BIT SET ON 'OPPOSITE PORT' AFTER UNLOAD
1131  001430  023057                    DH14
1132  001432  024172                    DT14
1133  001434  024341                    DF14
1134
1135                                ;ERROR 15
1136
1137  001436  020676                    EM15    ;ATTENTION BIT NOT SET ON PORT WHICH ISSUED 'UNLOAD'
1138  001440  022511                    DH1
1139  001442  024072                    DT1
1140  001444  024312                    DF1
1141
1142                                ;ERROR 16
1143
1144  001446  020762                    EM16    ;DRIVE NOT IN NEUTRAL AFTER UNLOAD WITH 'CONTROLLER
1145                                            ;SELECT' SWITCH MOVED FROM 'A/B'
```

```
1146  001450  023001                    DH13
1147  001452  024162                    DT13
1148  001454  024332                    DF6
1149                            ;ERROR 17
1150
1151  001456  021106                    EM17    ;DRIVE LOCKED ON PORT 'A' BY SWITCH WHILE CYCLED UP
1152  001460  023177                    DH17
1153  001462  024210                    DT17
1154  001464  024347                    DF17
1155
1156                            ;ERROR 20
1157
1158  001466  021171                    EM20    ;DRIVE LOCKED ON PORT 'B' BY SWITCH WHILE CYCLED UP
1159  001470  023177                    DH17
1160  001472  024210                    DT17
1161  001474  024347                    DF17
1162
1163                            ;ERROR 21
1164
1165  001476  021254                    EM21    ;STATUS INCORRECT FOR PORT AFTER CYCLE UP
1166  001500  022511                    DH1
1167  001502  024072                    DT1
1168  001504  024312                    DF1
1169
1170                            ;ERROR 22
1171
1172  001506  021325                    EM22    ;REGISTER CONTENTS SEEN WHEN DRIVE SWITCHED ON 'OPPOSITE' PORT
1173  001510  022511                    DH1
1174  001512  024072                    DT1
1175  001514  024312                    DF1
1176
1177                            ;ERROR 23
1178
1179  001516  021423                    EM23    ;'NED' SET WHEN RMDS1 ACCESSED THROUGH PORT NOT SWITCHED
1180  001520  022511                    DH1
1181  001522  024072                    DT1
1182  001524  024312                    DF1
1183
1184                            ;ERROR 24
1185
1186  001526  021517                    EM24    ;DRIVE SWITCHED TO LOCKED OUT PORT WHEN RELEASED/
1187  001530  023216                    DH24
1188  001532  024216                    DT24
1189  001534  024335                    DF7
1190
1191                            ;ERROR 25
1192
1193  001536  021577                    EM25    ;RH11 DIDN'T RESPOND TO ADDRESSING
1194  001540  023322                    DH25
1195  001542  024230                    DT25
1196  001544  024351                    DF25
1197
1198                            ;ERROR 26
1199
1200  001546  000000                    0       ;UNUSED ERROR MESSAGES
1201  001550  000000                    0
```

```
1202  001552  000000                    0
1203  001554  000000                    0
1204
1205                                ;ERROR 27
1206
1207  001556  000000                    0       ;UNUSED ERROR MESSAGES
1208  001560  000000                    0
1209  001562  000000                    0
1210  001564  000000                    0
1211
1212                                ;ERROR 30
1213
1214  001566  021641                    EM30    ;DRIVE NOT SEIZED BY PORT 'N'
1215  001570  023331                    DH30
1216  001572  024234                    DT30
1217  001574  024352                    DF30
1218
1219                                ;ERROR 31
1220
1221  001576  021672                    EM31    ;WRONG STATUS SEEN BY THE SEIZING PORT
1222  001600  022562                    DH2
1223  001602  024106                    DT2
1224  001604  024317                    DF2
1225
1226                                ;ERROR 32
1227
1228  001606  021740                    EM32    ;REGISTER CONTENTS INCORRECT
1229  001610  022562                    DH2
1230  001612  024106                    DT2
1231  001614  024317                    DF2
1232
1233                                ;ERROR 33
1234
1235  001616  021770                    EM33    ;CONTROL BUS PARITY ERROR WHILE READING REGISTER
1236  001620  022511                    DH1
1237  001622  024072                    DT1
1238  001624  024312                    DF1
1239
1240                                ;ERROR 34
1241
1242  001626  022054                    EM34    ;CAN'T ACCESS DRIVE THROUGH EITHER PORT
1243  001630  023454                    DH34
1244  001632  024254                    DT34
1245  001634  024361                    DF34
1246
1247                                ;ERROR 35
1248
1249  001636  022123                    EM35    ;DRIVE NOT IN NEUTRAL AFTER RELEASE, REQUEST NOT SET
1250  001640  023552                    DH35
1251  001642  024266                    DT35
1252  001644  024335                    DF7
1253
1254                                ;ERROR 36
1255
1256  001646  022210                    EM36    ;DRIVE NOT IN NEUTRAL AFTER TIMEOUT, REQUEST NOT SET
1257  001650  023552                    DH35
```

```
                                        DI35
                                        DF7
                                ;ERROR 37
                                        EM37    ;REGISTER CONTENTS INCORRECT AFTER RELEASE/TIMEOUT
                                        DH37
                                        DT30
                                        DF30
                                ;ERROR 40
                                        EM40    ;DRIVE NOT SEIZED BY PORT AFTER RELEASE WIITH REQUEST SET
                                        DH40
                                        DT40
                                        DF7
                                ;ERROR 41
                                        EM41    ;REGISTER WRONG AFTER RELEASE WITH REQUEST SET
                                        DH30
                                        DT30
                                        DF30


                                ;;*****************************************************************
                                .SBTTL  STARTUP AND INITIALIZATION ROUTINES
                                ;;*****************************************************************
001706  005037  001270          START:  CLR     CHGADR          ;CLEAR THE 'CHANGE RH11 ADDRESS' INDICATOR
001712  000403                  START2: BR      START2          ;GO TO THE START
001714  012737  177777  001270  START1: MOV     #-1,CHGADR      ;SET THE 'CHANGE RH11 ADDRESS' INDICATOR
001722  000005                  START2: RESET                   ;CLEAR THE BUS
                                .SBTTL  INITIALIZE THE COMMON TAGS
                                ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
001724  012706  001100                  MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
001730  005026                  1$:     CLR     (R6)+           ;;CLEAR MEMORY LOCATION
001732  022706  001140                  CMP     #SWR,R6 ;;DONE?
001736  001374                          BNE     .-6             ;;LOOP BACK IF NO
001740  012706  001100                  MOV     #STACK,SP       ;;SETUP THE STACK POINTER
                                ;;INITIALIZE A FEW VECTORS
001744  012737  013440  000020          MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
001752  012737  000340  000022          MOV     #340,@#IOTVEC+2 ;;LEVEL 7
001760  012737  013622  000030          MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
001766  012737  000340  000032          MOV     #340,@#EMTVEC+2 ;;LEVEL 7
001774  012737  016422  000034          MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
002002  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
002010  013737  013052  013044          MOV     $ENDCT,$EOPCT   ;;SETUP END-OF-PROGRAM COUNTER
002016  005037  001176                  CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
002022  005037  001200                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
002026  112737  000001  001115          MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
002034  012737  002034  001106          MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
```

```
1314  002042  012737  002042  001110          MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
1315                                          ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
1316                                          ;;EQUAL TO A "-1" SETUP FOR A SOFTWARE SWITCH REGISTER.
1317  002050  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
1318  002054  012737  002110  000004          MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
1319  002062  012737  177570  001140          MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
1320  002070  012737  177570  001142          MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
1321  002076  022777  177777  177034          CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
1322  002104  001012                          BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
1323                                                                  ;;AND  THE HARDWARE SWR IS NOT = -1
1324  002106  000403                          BR      65$             ;;BRANCH IF NO TIMEOUT
1325  002110  012716  002116          64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
1326  002114  000002                          RTI
1327  002116  012737  000176  001140  65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
1328  002124  012737  000174  001142          MOV     #DISPREG,DISPLAY
1329  002132  012637  000004          66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR
1330
1331  002136  005227  177777                  INC     #-1             ;FIRST START ?
1332  002142  001002                          BNE     1$              ;BR IF NOT
1333  002144  104401  016510                  TYPE    ,TITLE          ;TYPE PROGRAM NAME
1334  002150  004737  015032          1$:     JSR     PC,$TKINT       ;SETUP THE TTY KEYBOARD
1335                                  .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
1336  002154  005737  000042                  TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
1337  002160  001006                          BNE     67$             ;;BRANCH IF YES
1338  002162  023727  001140  000176          CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
1339  002170  001005                          BNE     68$             ;;BRANCH IF NO
1340  002172  104406                          GTSWR                   ;;GET SOFT-SWR SETTINGS
1341  002174  000403                          BR      68$
1342  002176  112737  000001  001134  67$:    MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
1343  002204                          68$:
1344  002204  004737  002576                  JSR     PC,CHANGE       ;CHECK/CHANGE THE RH11 ADDRESS
1345  002210  104401  016613                  TYPE    ,ENTERA         ;ENTER DRIVE ADDRESS
1346  002214  104412                          RDOCT                   ;GET THE ADDRESS
1347  002216  012637  001224                  MOV     (SP)+,PORTA     ;STORE THE ADDRESS
1348  002222  023727  001224  000007          CMP     PORTA,#7        ;SEE IF ADDRESS TOO LARGE
1349  002230  101403                          BLOS    2$              ;BR IF NOT
1350  002232  104401  016643                  TYPE    ,ADRERR         ;TYPE ADDRESS ERROR MESSAGE
1351  002236  000744                          BR      1$              ;TRY AGAIN
1352  002240  013737  001224  001226  2$:     MOV     PORTA,PORTB     ;GENERATE THE PORT B ADDRESS
1353  002246  005237  001226                  INC     PORTB           ;INCREMENT THE ADDRESS
1354  002252  042737  000016  001226          BIC     #16,PORTB       ;LEAVE BIT 0
1355  002260  013746  001224                  MOV     PORTA,-(SP)     ;PUT PORT A ADDRESS ON THE STACK
1356  002264  042716  177771                  BIC     #^C6,(SP)       ;SAVE BITS 1 & 2
1357  002270  052637  001226                  BIS     (SP)+,PORTB     ;SET BITS 1 & 2 IN PORT B ADDRESS
1358  002274  104401  016665                  TYPE    ,PORTAIS        ;'PORT A ADDRESS IS '
1359  002300  013746  001224                  MOV     PORTA,-(SP)     ;;SAVE PORTA FOR TYPEOUT
1360                                                                  ;;TYPE PORT A ADDRESS
1361  002304  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
1362  002306     001                          .BYTE   1               ;;TYPE 1 DIGIT(S)
1363  002307     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
1364  002310  104401  016713                  TYPE    ,PORTBIS        ;'PORT B ADDRESS IS',
1365  002314  013746  001226                  MOV     PORTB,-(SP)     ;;SAVE PORTB FOR TYPEOUT
1366                                                                  ;;TYPE PORT B ADDRESS
1367  002320  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
1368  002322     001                          .BYTE   1               ;;TYPE 1 DIGIT(S)
1369  002323     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
```

```
1370  002324  104401  001207              TYPE    $CRLF           ;ANOTHER CR-LF
1371  002330  013737  001224  001230      MOV     PORTA,PORTC     ;GENERATE ADDRESS OF DRIVE NOT TESTED
1372  002336  062737  000006  001230      ADD     #6,PORTC        ;COMPLEMENT SOME BITS
1373  002344  042737  177770  001230      BIC     #^C7,PORTC      ;SAVE ONLY LOWER BITS
1374  002352  013701  001224              MOV     PORTA,R1        ;USE PORT A ADDRESS AS INDEX
1375  002356  116137  024406  001232      MOVB    ATABIT(R1),ASR1 ;GET ATTENTION BIT FOR DRIVE
1376  002364  005037  001256              CLR     TIMEA           ;CLEAR TIMEOUT ONE-SHOT VALUE LOCATION
1377  002370  005037  001260              CLR     TIMEAP          ;CLEAR TIMEOUT ONE-SHOT VALUE LOCATION
1378  002374  005037  001262              CLR     TIMEB           ;CLEAR TIMEOUT ONE-SHOT VALUE LOCATION
1379  002400  005037  001264              CLR     TIMEBP          ;CLEAR TIMEOUT ONE-SHOT VALUE LOCATION
1380  002404  004737  013226              JSR     PC,CKCLK        ;SETUP CLOCK
1381  002410  000137  002424              JMP     EXEC            ;CLOCK HAS BEEN STARTED
1382  002414  104401  016741              TYPE    ,NOCLOCK        ;NO CLOCK ON SYSTEM
1383  002420  000000                3$:   HALT                    ;FATAL ERROR
1384  002422  000776                      BR      3$              ;INTERLOCK THE HALT
1385
1386                                  ;ROUTINE TO GET THE TEST NUMBER FROM THE OPERATOR
1387
1388  002424  000005                EXEC: RESET                   ;CLEAR EVERYTHING
1389  002426  005037  177776              CLR     PS              ;CLEAR THE PROCESSOR STATUS WORD
1390  002432  104401  001207              TYPE    $CRLF           ;CR-LF
1391  002436  013700  001272              MOV     $RMADR,R0       ;RH11 ADDRESS FOR INDEXING
1392  002442  012706  001100              MOV     #STACK,SP       ;LOAD STACK POINTER
1393  002446  004737  013226              JSR     PC,CKCLK        ;START THE CLOCK
1394  002452  000240                      NOP                     ;RETURN IF NO CLOCK
1395  002454  004737  015032              JSR     PC,$TKINT       ;INITIALIZE THE KEYBOARD
1396  002460  005037  001266              CLR     KYBCTL          ;CLEAR SINGLE TEST INDICATOR
1397  002464  005037  001100              CLR     $PASS           ;CLEAR THE PASS COUNT
1398  002470  112737  000001  001115      MOVB    #1,$ERMAX       ;SET ERROR MAX TO 1
1399  002476  012737  002476  001106      MOV     #.,$LPADR       ;INITIAL SETTING FOR LOOP ADDRESS
1400  002504  012737  002504  001110      MOV     #.,$LPERR       ;INITIAL SETTING FOR LOOP ON ERROR ADDRESS
1401  002512  104401  017010        1$:   TYPE    ,TESTNO         ;ASK FOR TEST NUMBER
1402  002516  104412                      RDOCT                   ;GET THE NUMBER
1403  002520  012601                      MOV     (SP)+,R1        ;PUT ENTRY INTO R1
1404  002522  001002                      BNE     2$              ;BR IF NOT ZERO
1405  002524  000137  002720              JMP     TST1            ;ENTER ZERO - PERFORM ALL TESTS
1406  002530  020137  024416        2$:   CMP     R1,MAXTN        ;SEE IF NUMBER GREATER THAN MAXIMUM
1407  002534  003403                      BLE     3$              ;BR IF LESS OR EQUAL
1408  002536  104401  017030              TYPE    ,BADNO          ;BAD ENTRY
1409  002542  000763                      BR      1$              ;TRY AGAIN
1410  002544  005301        3$:   DEC     R1                      ;DECREMENT ENTRY
1411  002546  006301              ASL     R1                      ;SHIFT IT LEFT
1412  002550  016137  024366  002574      MOV     TSTADR(R1),4$   ;GET THE TEST ADDRESS
1413  002556  005237  001266              INC     KYBCTL          ;SET SINGLE TEST INDICATOR
1414  002562  012737  000001  001104      MOV     #1,$ICNT        ;PRESET ITERATION COUNT
1415  002570  000177  000000              JMP     @4$             ;GO TO THE SELECTED TEST
1416  002574  000000                4$:   .WORD   0               ;TEST ADDRESS GOES HERE
1417
1418                                  ;CHANGE THE RH11 UNIBUS ADDRESS USED BY THE PROGRAM
1419
1420  002576  005737  001270        CHANGE: TST   CHGADR          ;CHANGE THE ADDRESS ?
1421  002602  001421                      BEQ     3$              ;BR IF NOT
1422  002604  005037  001270              CLR     CHGADR          ;CLEAR THE INDICATOR
1423  002610  104401  017056        1$:   TYPE    ,ADDRIS         ;TYPE OUT WHAT THE PRESENT ADDRESS IS
1424  002614  013746  001272              MOV     $RMADR,-(SP)    ;PUT THE ADDRESS ON THE STACK
1425  002620  104402                      TYPOC                   ;TYPE THE ACTUAL ADDRESS
```

```
1426  002622  104401  001207                    TYPE    ,$CRLF          ;CR-LF
1427  002626  104401  017136                    TYPE    ,NTRH11         ;ASK FOR NEW ADDRESS
1428  002632  104412                            RDOCT
1429  002634  005716                            TST     (SP)            ;0 OR 'CR' ENTERED ?
1430  002636  001402                            BEQ     2$              ;BR IF EITHER ENTERED (NO ADDRESS CHANGE)
1431  002640  011637  001272                    MOV     (SP),$RMADR     ;NEW RH11 ADDRESS
1432  002644  005726                    2$:     TST     (SP)+           ;CORRECT THE STACK POINTER
1433  002646  012737  002666  000004    3$:     MOV     #4$,@#4         ;LOAD TRAP ADDRESS
1434  002654  013700  001272                    MOV     $RMADR,R0       ;RH11 ADDRESS
1435  002660  005760  000002                    TST     RMWC(R0)        ;SEE IF RH11 RESPONDS AT THAT ADDRESS
1436  002664  000404                            BR      5$              ;BR, RH11 ALIVE AT PRESENT ADDRESS
1437  002666  104025                    4$:     ERROR   25              ;NO RESPONSE TO ADDRESS
1438  002670  062706  000004                    ADD     #4,SP           ;RESET THE STACK POINTER
1439  002674  000745                            BR      1$              ;GET ADDRESS AGAIN
1440  002676  012737  000006  000004    5$:     MOV     #6,@#4          ;RESTORE THE VECTOR
1441  002704  000207                            RTS     PC              ;RETURN
1442
1443                                    ;;**********************************************************************
1444
1445                                    .SBTTL  *** TESTS ***
1446
1447                                    ;;**********************************************************************
1448
1449
1450  002706  013700  001272    TST1AA: MOV     $RMADR,R0       ;;RESTORE R0 AFTER END OF PASS
1451  002712  012760  000040  000010    MOV     #CLR,RMCS2(R0)  ;CLEAR MASSBUS
1452
1453                                    ;;**********************************************************************
1454                                    ;*TEST 1        DRIVE ACCESS TEST
1455                                    ;*
1456                                    ;*VERIFY THAT THE DRIVE CAN BE ACCESSED THROUGH BOTH PORTS
1457                                    ;*
1458                                    ;*  A.  SELECT DRIVE, VERIFY THAT THE DRIVE IS PRESENT, THAT THE
1459                                    ;*      DRIVE IS A DUAL PORT RM03, THAT THE DRIVE IS ONLINE (RMDS1 HAS
1460                                    ;*      'MOL', 'PGM', 'DPR', & 'DRY' BITS SET), AND THE THE DRIVE SERIAL
1461                                    ;*      NUMBER READ THROUGH BOTH PORTS IS THE SAME.
1462                                    ;*
1463                                    ;*  B.  THE TEST IS REPEATED THROUGH BOTH PORTS.
1464                                    ;*
1465                                    ;;**********************************************************************
1466  002720                            TST1:
1467  002720  005737  001266                    TST     KYBCTL          ;PERFORMING ONLY SINGLE TESTS ?
1468  002724  001406                            BEQ     2$              ;BR IF NOT
1469  002726  100002                            BPL     1$              ;BR IF JUST ENTERED TEST
1470  002730  000137  002424                    JMP     EXEC            ;RETURN & GET NEXT TEST NUMBER
1471  002734  012737  177777  001266    1$:     MOV     #-1,KYBCTL      ;SET SINGLE TEST INDICATOR
1472  002742  112737  000001  001102    2$:     MOVB    #1,$TSTNM       ;TEST NUMBER
1473  002750  012737  002772  001106            MOV     #TEST1,$LPADR   ;LOAD LOOP ON TEST ADDRESS
1474  002756  012737  002772  001110            MOV     #TEST1,$LPERR   ;LOAD LOOP ON ERROR ADDRESS
1475  002764  012737  000001  001176            MOV     #1,$TIMES       ;;DO 1 ITERATION
1476  002772  012706  001100            TEST1:  MOV     #STACK,SP       ;SETUP THE STACK POINTER
1477
1478                                    ;;**********************************************************************
1479                                    ;VERIFY THAT DRIVE IS PRESENT THROUGH PORTS A & B
1480
1481  002776  113760  001224  000010            MOVB    PORTA,RMCS2(R0)  ;SELECT PORT A
```

```
1482  003004  013737  001224  001234          MOV     PORTA,PTNBR  ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
1483  003012  005760  000012                  TST     RMDS1(R0)           ;SEE IF DRIVE (PORT A) PRESENT
1484  003016  005037  001244                  CLR     CKERR               ;CLEAR THE 'CHECK ERROR' INDICATOR
1485  003022  016037  000010  001126          MOV     RMCS2(R0),$BDDAT    ;GET CONTENTS OF RMCS2
1486  003030  012737  000010  001122          MOV     #RMCS2,$BDADR  ;FORM REGISTER ADDRESS OF ERROR MESSAGE
1487  003036  060037  001122                  ADD     R0,$BDADR           ;ADD RH11 BASE ADDRESS
1488  003042  005037  001124                  CLR     $GDDAT              ;WHAT REGISTER SHOULD BE
1489  003046  013737  001126  001164          MOV     $BDDAT,$TMP0        ;MOVE REGISTER CONTENTS TO '$TMP0'
1490  003054  042737  167777  001164          BIC     #^CNED,$TMP0  ;SAVE SPECIFIED BITS
1491  003062  023737  001124  001164          CMP     $GDDAT,$TMP0        ;COMPARE THE BITS
1492  003070  001414                          BEQ     64$                 ;BR IF OK
1493  003072  013737  001126  001174          MOV     $BDDAT,$TMP4        ;COPY 'BAD DATA'
1494  003100  042737  010000  001174          BIC     #NED,$TMP4          ;CLEAR THE MASKED BITS
1495  003106  053737  001174  001124          BIS     $TMP4,$GDDAT        ;'OR' WITH GOOD DATA FOR TYPEOUT
1496  003114  104001                          ERROR   1                   ;TYPE MESSAGE 1
1497  003116  005137  001244                  COM     CKERR               ;SET THE REGISTER COMPARE ERROR INDICATOR
1498  003122  000240                  64$:    NOP
1499  003124  005737  001244                  TST     CKERR               ;WAS 'NED' SET ?
1500  003130  001403                          BEQ     .+10                ;BR IF NOT
1501  003132  012760  000040  000010          MOV     #CLR,RMCS2(R0)      ;ISSUE MASSBUS INIT TO CLEAR 'NED'
1502  003140  113760  001226  000010          MOVB    PORTB,RMCS2(R0)     ;SELECT PORT B
1503  003146  013737  001226  001234          MOV     PORTB,PTNBR  ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
1504  003154  005760  000012                  TST     RMDS1(R0)           ;SEE IF DRIVE (PORT B) PRESENT
1505  003160  005037  001244                  CLR     CKERR               ;CLEAR THE 'CHECK ERROR' INDICATOR
1506  003164  016037  000010  001126          MOV     RMCS2(R0),$BDDAT    ;GET CONTENTS OF RMCS2
1507  003172  012737  000010  001122          MOV     #RMCS2,$BDADR  ;FORM REGISTER ADDRESS OF ERROR MESSAGE
1508  003200  060037  001122                  ADD     R0,$BDADR           ;ADD RH11 BASE ADDRESS
1509  003204  005037  001124                  CLR     $GDDAT              ;WHAT REGISTER SHOULD BE
1510  003210  013737  001126  001164          MOV     $BDDAT,$TMP0        ;MOVE REGISTER CONTENTS TO '$TMP0'
1511  003216  042737  167777  001164          BIC     #^CNED,$TMP0  ;SAVE SPECIFIED BITS
1512  003224  023737  001124  001164          CMP     $GDDAT,$TMP0        ;COMPARE THE BITS
1513  003232  001414                          BEQ     66$                 ;BR IF OK
1514  003234  013737  001126  001174          MOV     $BDDAT,$TMP4        ;COPY 'BAD DATA'
1515  003242  042737  010000  001174          BIC     #NED,$TMP4          ;CLEAR THE MASKED BITS
1516  003250  053737  001174  001124          BIS     $TMP4,$GDDAT        ;'OR' WITH GOOD DATA FOR TYPEOUT
1517  003256  104001                          ERROR   1                   ;TYPE MESSAGE 1
1518  003260  005137  001244                  COM     CKERR               ;SET THE REGISTER COMPARE ERROR INDICATOR
1519  003264  000240                  66$:    NOP
1520  003266  005737  001244                  TST     CKERR               ;WAS 'NED' SET ?
1521  003272  001403                          BEQ     .+10                ;BR IF NOT
1522  003274  012760  000040  000010          MOV     #CLR,RMCS2(R0)      ;ISSUE MASSBUS INIT TO CLEAR 'NED'
1523
1524                          ;;*****************************************************************
1525                          ;CONFIRM THAT DRIVE IS AN RM03 AND IS DUAL PORT
1526
1527  003302  113760  001224  000010          MOVB    PORTA,RMCS2(R0)     ;SELECT PORT A
1528  003310  013737  001224  001234          MOV     PORTA,PTNBR  ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
1529  003316  005037  001244                  CLR     CKERR               ;CLEAR THE 'CHECK ERROR' INDICATOR
1530  003322  016037  000026  001126          MOV     RMDT(R0),$BDDAT     ;GET CONTENTS OF RMDT
1531  003330  012737  000026  001122          MOV     #RMDT,$BDADR  ;FORM REGISTER ADDRESS OF ERROR MESSAGE
1532  003336  060037  001122                  ADD     R0,$BDADR           ;ADD RH11 BASE ADDRESS
1533  003342  012737  024024  001124          MOV     #024024,$GDDAT      ;WHAT REGISTER SHOULD BE
1534  003350  013737  001126  001164          MOV     $BDDAT,$TMP0        ;MOVE REGISTER CONTENTS TO '$TMP0'
1535  003356  042737  000003  001164          BIC     #^C177774,$TMP0  ;SAVE SPECIFIED BITS
1536  003364  023737  001124  001164          CMP     $GDDAT,$TMP0        ;COMPARE THE BITS
1537  003372  001414                          BEQ     68$                 ;BR IF OK
```

```
1538  003374  013737  001126  001174          MOV     $BDDAT,$TMP4    ;COPY 'BAD DATA'
1539  003402  042737  177774  001174          BIC     #177774,$TMP4   ;CLEAR THE MASKED BITS
1540  003410  053737  001174  001124          BIS     $TMP4,$GDDAT    ;'OR' WITH GOOD DATA FOR TYPEOUT
1541  003416  104002                          ERROR   2               ;TYPE MESSAGE 2
1542  003420  005137  001244                  COM     CKERR           ;SET THE REGISTER COMPARE ERROR INDICATOR
1543  003424  000240                  68$:    NOP
1544  003426  113760  001226  000010          MOVB    PORTB,RMCS2(R0)  ;SELECT PORT B
1545  003434  013737  001226  001234          MOV     PORTB,PTNBR  ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
1546  003442  005037  001244                  CLR     CKERR           ;CLEAR THE 'CHECK ERROR' INDICATOR
1547  003446  016037  000026  001126          MOV     RMDT(R0),$BDDAT  ;GET CONTENTS OF RMDT
1548  003454  012737  000026  001122          MOV     #RMDT,$BDADR  ;FORM REGISTER ADDRESS OF ERROR MESSAGE
1549  003462  060037  001122                  ADD     R0,$BDADR       ;ADD RH11 BASE ADDRESS
1550  003466  012737  024024  001124          MOV     #024024,$GDDAT  ;WHAT REGISTER SHOULD BE
1551  003474  013737  001126  001164          MOV     $BDDAT,$TMP0    ;MOVE REGISTER CONTENTS TO '$TMP0'
1552  003502  042737  000003  001164          BIC     #^C177774,$TMP0  ;SAVE SPECIFIED BITS
1553  003510  023737  001124  001164          CMP     $GDDAT,$TMP0    ;COMPARE THE BITS
1554  003516  001414                          BEQ     70$             ;BR IF OK
1555  003520  013737  001126  001174          MOV     $BDDAT,$TMP4    ;COPY 'BAD DATA'
1556  003526  042737  177774  001174          BIC     #177774,$TMP4   ;CLEAR THE MASKED BITS
1557  003534  053737  001174  001124          BIS     $TMP4,$GDDAT    ;'OR' WITH GOOD DATA FOR TYPEOUT
1558  003542  104002                          ERROR   2               ;TYPE MESSAGE 2
1559  003544  005137  001244                  COM     CKERR           ;SET THE REGISTER COMPARE ERROR INDICATOR
1560  003550  000240                  70$:    NOP
1561
1562                                  ;;*****************************************************************
1563                                  ;VERIFY THROUGH BOTH PORTS THAT THE DRIVE IS ON LINE AND IN NEUTRAL
1564
1565  003552  113760  001224  000010          MOVB    PORTA,RMCS2(R0)  ;SELECT PORT A
1566  003560  013737  001224  001234          MOV     PORTA,PTNBR  ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
1567  003566  005037  001244                  CLR     CKERR           ;CLEAR THE 'CHECK ERROR' INDICATOR
1568  003572  016037  000012  001126          MOV     RMDS1(R0),$BDDAT  ;GET CONTENTS OF RMDS1
1569  003600  012737  000012  001122          MOV     #RMDS1,$BDADR  ;FORM REGISTER ADDRESS OF ERROR MESSAGE
1570  003606  060037  001122                  ADD     R0,$BDADR       ;ADD RH11 BASE ADDRESS
1571  003612  012737  001000  001124          MOV     #PGM,$GDDAT  ;WHAT REGISTER SHOULD BE
1572  003620  013737  001126  001164          MOV     $BDDAT,$TMP0    ;MOVE REGISTER CONTENTS TO '$TMP0'
1573  003626  042737  176777  001164          BIC     #^CPGM,$TMP0  ;SAVE SPECIFIED BITS
1574  003634  023737  001124  001164          CMP     $GDDAT,$TMP0    ;COMPARE THE BITS
1575  003642  001414                          BEQ     72$             ;BR IF OK
1576  003644  013737  001126  001174          MOV     $BDDAT,$TMP4    ;COPY 'BAD DATA'
1577  003652  042737  001000  001174          BIC     #PGM,$TMP4      ;CLEAR THE MASKED BITS
1578  003660  053737  001174  001124          BIS     $TMP4,$GDDAT    ;'OR' WITH GOOD DATA FOR TYPEOUT
1579  003666  104003                          ERROR   3               ;TYPE MESSAGE 3
1580  003670  005137  001244                  COM     CKERR           ;SET THE REGISTER COMPARE ERROR INDICATOR
1581  003674  000240                  72$:    NOP
1582  003676  005037  001244                  CLR     CKERR           ;CLEAR THE 'CHECK ERROR' INDICATOR
1583  003702  016037  000012  001126          MOV     RMDS1(R0),$BDDAT  ;GET CONTENTS OF RMDS1
1584  003710  012737  000012  001122          MOV     #RMDS1,$BDADR  ;FORM REGISTER ADDRESS OF ERROR MESSAGE
1585  003716  060037  001122                  ADD     R0,$BDADR       ;ADD RH11 BASE ADDRESS
1586  003722  012737  010600  001124          MOV     #MOL!DPR!DRY,$GDDAT  ;WHAT REGISTER SHOULD BE
1587  003730  013737  001126  001164          MOV     $BDDAT,$TMP0    ;MOVE REGISTER CONTENTS TO '$TMP0'
1588  003736  042737  167177  001164          BIC     #^C10600,$TMP0  ;SAVE SPECIFIED BITS
1589  003744  023737  001124  001164          CMP     $GDDAT,$TMP0    ;COMPARE THE BITS
1590  003752  001414                          BEQ     74$             ;BR IF OK
1591  003754  013737  001126  001174          MOV     $BDDAT,$TMP4    ;COPY 'BAD DATA'
1592  003762  042737  010600  001174          BIC     #10600,$TMP4    ;CLEAR THE MASKED BITS
1593  003770  053737  001174  001124          BIS     $TMP4,$GDDAT    ;'OR' WITH GOOD DATA FOR TYPEOUT
```

```
1594  003776  104004                        ERROR   4                       ;TYPE MESSAGE 4
1595  004000  005137  001244                COM     CKERR                   ;SET THE REGISTER COMPARE ERROR INDICATOR
1596  004004  000240                74$:    NOP
1597  004006  113760  001226  000010        MOVB    PORTB,RMCS2(R0)  ;SELECT PORT B
1598  004014  013737  001226  001234        MOV     PORTB,PTNBR  ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
1599  004022  005037  001244                CLR     CKERR                   ;CLEAR THE 'CHECK ERROR' INDICATOR
1600  004026  016037  000012  001126        MOV     RMDS1(R0),$BDDAT  ;GET CONTENTS OF RMDS1
1601  004034  012737  000012  001122        MOV     #RMDS1,$BDADR  ;FORM REGISTER ADDRESS OF ERROR MESSAGE
1602  004042  060037  001122                ADD     R0,$BDADR               ;ADD RH11 BASE ADDRESS
1603  004046  012737  001000  001124        MOV     #PGM,$GDDAT  ;WHAT REGISTER SHOULD BE
1604  004054  013737  001126  001164        MOV     $BDDAT,$TMP0    ;MOVE REGISTER CONTENTS TO '$TMP0'
1605  004062  042737  176777  001164        BIC     #^CPGM,$TMP0  ;SAVE SPECIFIED BITS
1606  004070  023737  001124  001164        CMP     $GDDAT,$TMP0    ;COMPARE THE BITS
1607  004076  001414                        BEQ     76$             ;BR IF OK
1608  004100  013737  001126  001174        MOV     $BDDAT,$TMP4    ;COPY 'BAD DATA'
1609  004106  042737  001000  001174        BIC     #PGM,$TMP4      ;CLEAR THE MASKED BITS
1610  004114  053737  001174  001124        BIS     $TMP4,$GDDAT    ;'OR' WITH GOOD DATA FOR TYPEOUT
1611  004122  104003                        ERROR   3               ;TYPE MESSAGE 3
1612  004124  005137  001244                COM     CKERR                   ;SET THE REGISTER COMPARE ERROR INDICATOR
1613  004130  000240                76$:    NOP
1614  004132  005037  001244                CLR     CKERR                   ;CLEAR THE 'CHECK ERROR' INDICATOR
1615  004136  016037  000012  001126        MOV     RMDS1(R0),$BDDAT  ;GET CONTENTS OF RMDS1
1616  004144  012737  000012  001122        MOV     #RMDS1,$BDADR  ;FORM REGISTER ADDRESS OF ERROR MESSAGE
1617  004152  060037  001122                ADD     R0,$BDADR               ;ADD RH11 BASE ADDRESS
1618  004156  012737  010600  001124        MOV     #MOL!DPR!DRY,$GDDAT  ;WHAT REGISTER SHOULD BE
1619  004164  013737  001126  001164        MOV     $BDDAT,$TMP0    ;MOVE REGISTER CONTENTS TO '$TMP0'
1620  004172  042737  167177  001164        BIC     #^C10600,$TMP0  ;SAVE SPECIFIED BITS
1621  004200  023737  001124  001164        CMP     $GDDAT,$TMP0    ;COMPARE THE BITS
1622  004206  001414                        BEQ     78$             ;BR IF OK
1623  004210  013737  001126  001174        MOV     $BDDAT,$TMP4    ;COPY 'BAD DATA'
1624  004216  042737  010600  001174        BIC     #10600,$TMP4    ;CLEAR THE MASKED BITS
1625  004224  053737  001174  001124        BIS     $TMP4,$GDDAT    ;'OR' WITH GOOD DATA FOR TYPEOUT
1626  004232  104004                        ERROR   4               ;TYPE MESSAGE 4
1627  004234  005137  001244                COM     CKERR                   ;SET THE REGISTER COMPARE ERROR INDICATOR
1628  004240  000240                78$:    NOP
1629
1630                                        ;;*****************************************************************
1631                                        ;VERIFY THAT DRIVE SERIAL NUMBER SEEN THROUGH BOTH PORTS IS THE SAME
1632
1633  004242  113760  001224  000010        MOVB    PORTA,RMCS2(R0) ;SELECT PORT A
1634  004250  016037  000030  001124        MOV     RMSN(R0),$GDDAT ;STORE THE PORT A SERIAL NUMBER
1635  004256  113760  001226  000010        MOVB    PORTB,RMCS2(R0) ;SELECT PORT B
1636  004264  016037  000030  001126        MOV     RMSN(R0),$BDDAT ;STORE THE PORT B SERIAL NUMBER
1637  004272  023737  001124  001126        CMP     $GDDAT,$BDDAT   ;ARE THEY THE SAME ?
1638  004300  001406                        BEQ     1$              ;BR IF THEY ARE
1639  004302  104005                        ERROR   5               ;REPORT THE ERROR
1640  004304  032777  100000  174626        BIT     #SW15,@SWR      ;HALT ON ERROR ?
1641  004312  001001                        BNE     1$              ;BR IF SET - PROGRAM HAS ALREADY HALTED
1642  004314  000000                        HALT                    ;HALT, POSSIBLE CABLE CONNECTION PROBLEM
1643  004316  000004                1$:     SCOPE                   ;LOOP'?
1644
1645
1646
1647                                        ;;*****************************************************************
1648                                        ;*TEST 2        SET 'VV' FOR PORT A
1649                                        ;*
```

```
                                        ;*SET VOLUME VALID
                                        ;*
                                        ;*  A.  ISSUE A DRIVE CLEAR COMMAND THROUGH PORT A.
                                        ;*
                                        ;*  B.  ISSUE A READIN PRESET COMMAND THROUGH PORT A.  VERIFY
                                        ;*      THAT THE 'VV' BIT IS SET FOR PORT A.
                                        ;*
                                        ;*  C.  ISSUE A RELEASE COMMAND THROUGH PORT A.  VERIFY THAT
                                        ;*      THE DRIVE RETURNED TO NEUTRAL AND THAT NEITHER ATTENTION
                                        ;*      BIT IS SET.
                                        ;*
                                        ;;*****************************************************************
004320                                  TST2:
004320  005737  001266                          TST     KYBCTL          ;PERFORMING ONLY SINGLE TESTS ?
004324  001406                                  BEQ     2$              ;BR IF NOT
004326  100002                                  BPL     1$              ;BR IF JUST ENTERED TEST
004330  000137  002424                          JMP     EXEC            ;RETURN & GET NEXT TEST NUMBER
004334  012737  177777  001266    1$:           MOV     #-1,KYBCTL      ;SET SINGLE TEST INDICATOR
004342  112737  000002  001102    2$:           MOVB    #2,$TSTNM       ;TEST NUMBER
004350  012737  004372  001106                  MOV     #TEST2,$LPADR   ;LOAD LOOP ON TEST ADDRESS
004356  012737  004372  001110                  MOV     #TEST2,$LPERR   ;LOAD LOOP ON ERROR ADDRESS
004364  012737  000001  001176                  MOV     #1,$TIMES       ;;DO 1 ITERATION
004372  012706  001100          TEST2:          MOV     #STACK,SP       ;SETUP THE STACK POINTER
004376  113760  001224  000010                  MOVB    PORTA,RMCS2(R0) ;SELECT PORT A
004404  013737  001224  001234                  MOV     PORTA,PTNBR  ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT

                                        ;;*****************************************************************
                                        ;SET VOLUME VALUE FOR PORT

004412  012760  000011  000000                  MOV     #11,RMCS1(R0)   ;ISSUE A DRIVE CLEAR
004420  012760  000021  000000                  MOV     #21,RMCS1(R0)   ;ISSUE A READIN PRESET
004426  012760  010000  000032                  MOV     #FMT16,RMOF(R0)  ;SET FMT16

                                        ;;*****************************************************************
                                        ;VERIFY THAT THE DRIVE STATUS IS CORRECT

004434  005037  001244                          CLR     CKERR           ;CLEAR THE 'CHECK ERROR' INDICATOR
004440  016037  000012  001126                  MOV     RMDS1(R0),$BDDAT  ;GET CONTENTS OF RMDS1
004446  012737  000012  001122                  MOV     #RMDS1,$BDADR   ;FORM REGISTER ADDRESS OF ERROR MESSAGE
004454  060037  001122                          ADD     R0,$BDADR       ;ADD RH11 BASE ADDRESS
004460  012737  011700  001124                  MOV     #MOL!PGM!DPR!DRY!VV,$GDDAT   ;WHAT REGISTER SHOULD BE
004466  013737  001126  001164                  MOV     $BDDAT,$TMP0    ;MOVE REGISTER CONTENTS TO '$TMP0'
004474  042737  106077  001164                  BIC     #^C71700,$TMP0  ;SAVE SPECIFIED BITS
004502  023737  001124  001164                  CMP     $GDDAT,$TMP0    ;COMPARE THE BITS
004510  001414                                  BEQ     64$             ;BR IF OK
004512  013737  001126  001174                  MOV     $BDDAT,$TMP4    ;COPY 'BAD DATA'
004520  042737  071700  001174                  BIC     #71700,$TMP4    ;CLEAR THE MASKED BITS
004526  053737  001174  001124                  BIS     $TMP4,$GDDAT    ;'OR' WITH GOOD DATA FOR TYPEOUT
004534  104010                                  ERROR   10              ;TYPE MESSAGE 10
004536  005137  001244                          COM     CKERR           ;SET THE REGISTER COMPARE ERROR INDICATOR
004542  000240                  64$:            NOP

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

                                        ;RELEASE THE DRIVE FROM PORT A
```

```
1706  004544  113760  001224  000010          MOVB    PORTA,RMCS2(R0)  ;SELECT PORT A
1707  004552  013737  001224  001234          MOV     PORTA,PTNBR  ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
1708  004560  012760  000013  000000          MOV     #13,RMCS1(R0)  ;ISSUE RELEASE THROUGH PORT A
1709
1710                                          ;VERIFY THAT THE DRIVE IS IN NEUTRAL
1711
1712  004566  005037  001250                  CLR     RELERR          ;CLEAR THE 'RELEASE ERROR ' INDICATOR
1713  004572  012737  000012  001122          MOV     #RMDS1,$BDADR   ;FORM THE ADDRESS OF RMDS1 FOR TYPEOUT
1714  004600  060037  001122                  ADD     R0,$BDADR       ;ADD THE I/O BASE ADDRESS
1715  004604  012737  011600  001124          MOV     #MOL!PGM!DPR!DRY,$GDDAT  ;COMPARISON CONSTANT
1716  004612  113760  001224  000010          MOVB    PORTA,RMCS2(R0) ;SELECT PORT A.
1717  004620  016037  000012  001170          MOV     RMDS1(R0),$TMP2 ;GET THE DRIVE STATUS REGISTER FROM PORT A.
1718  004626  013737  001170  001164          MOV     $TMP2,$TMP0     ;COPY IT INTO '$TMP0'
1719  004634  042737  100100  001164          BIC     #ATA!VV,$TMP0   ;CLEAR PORT DEPENDENT BITS FROM THE COPY
1720  004642  113760  001226  000010          MOVB    PORTB,RMCS2(R0) ;SELECT PORT B.
1721  004650  016037  000012  001172          MOV     RMDS1(R0),$TMP3 ;GET THE DRIVE STATUS REGISTER FROM PORT B.
1722  004656  013737  001172  001166          MOV     $TMP3,$TMP1     ;COPY IT INTO '$TMP1'
1723  004664  042737  100100  001166          BIC     #ATA!VV,$TMP1   ;CLEAR PORT DEPENDENT BITS FROM THE COPY
1724  004672  023737  001164  001166          CMP     $TMP0,$TMP1     ;IS THE STATUS REGISTER THE SAME FROM BOTH PORTS ?
1725  004700  001006                          BNE     66$             ;BR IF NOT
1726  004702  005737  001164                  TST     $TMP0           ;REGISTERS ARE THE SAME: ARE THEY ZERO ?
1727  004706  001037                          BNE     68$             ;BR IF NOT
1728  004710  104034                          ERROR   34              ;REPORT DRIVE NOT IN NEUTRAL OR NOT SEIZED
1729  004712  000137  005076                  JMP     70$             ;BYPASS THE REST OF THE CHECKS
1730  004716  013737  001170  001126  66$:    MOV     $TMP2,$BDDAT    ;SET UP POSSIBLE BAD DATA FOR ERROR MESSAGE
1731  004724  013737  001226  001234          MOV     PORTB,PTNBR     ;SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
1732  004732  113760  001226  000010          MOVB    PORTB,RMCS2(R0) ;SELECT PORT B.
1733  004740  005737  001164                  TST     $TMP0           ;SEE IF STATUS EQ 0 FROM PORT A.
1734  004744  001414                          BEQ     67$             ;BR IF ZERO
1735  004746  013737  001224  001234          MOV     PORTA,PTNBR     ;SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
1736  004754  013737  001172  001126          MOV     $TMP3,$BDDAT    ;'BAD DATA' FOR ERROR TYPE OUT
1737  004762  113760  001224  000010          MOVB    PORTA,RMCS2(R0) ;SELECT PORT A.
1738  004770  005737  001166                  TST     $TMP1           ;SEE IF STATUS EQ ZERO FROM PORT B.
1739  004774  001004                          BNE     68$             ;BR IF NOT
1740  004776  012737  177777  001250  67$:    MOV     #-1,RELERR      ;SET 'RELEASE ERROR' INDICATOR
1741  005004  104036                          ERROR   36              ;TYPE ERROR MESSAGE 36
1742  005006  013737  001170  001126  68$:    MOV     $TMP2,$BDDAT    ;LOOK FOR BIT FAILURES WHEN RMDS1 READ
1743  005014  013737  001224  001234          MOV     PORTA,PTNBR     ;CHANGE PORT NUMBER
1744  005022  042737  100100  001170          BIC     #ATA!VV,$TMP2   ;DON'T CHECK ATTN BIT OR VV BIT
1745  005030  023737  001124  001170          CMP     $GDDAT,$TMP2    ;ALL BITS OK ?
1746  005036  001401                          BEQ     69$             ;BR IF OK FROM PORT A.
1747  005040  104037                          ERROR   37              ;REPORT ERROR
1748  005042  013737  001172  001126  69$:    MOV     $TMP3,$BDDAT    ;CHECK RMDS1 FOR BIT FAILURES - FROM PORT B.
1749  005050  013737  001226  001234          MOV     PORTB,PTNBR     ;CHANGE PORT NUMBER
1750  005056  042737  100100  001172          BIC     #ATA!VV,$TMP3   ;DON'T CHECK ATTN BIT OR VV BIT
1751  005064  023737  001124  001172          CMP     $GDDAT,$TMP3    ;SEE IF READ OK FROM PORT B.
1752  005072  001401                          BEQ     70$             ;BR IF OK
1753  005074  104037                          ERROR   37              ;REPORT THE ERROR
1754  005076  000240                  70$:    NOP
1755  005100  000004                          SCOPE                   ;LOOP ?
1756
1757                                  ;;*****************************************************************
1758                                  ;*TEST 3        SET 'VV' FOR PORT B
1759                                  ;*
1760                                  ;*SET VOLUME VALID
1761                                  ;*
```

```
                                        ;*  A.  ISSUE A DRIVE CLEAR COMMAND THROUGH PORT B.
                                        ;*
                                        ;*  B.  ISSUE A READIN PRESET COMMAND THROUGH PORT B.  VERIFY
                                        ;*      THAT THE 'VV' BIT IS SET FOR PORT B.
                                        ;*
                                        ;*  C.  ISSUE A RELEASE COMMAND THROUGH PORT B.  VERIFY THAT
                                        ;*      THE DRIVE RETURNED TO NEUTRAL AND THAT NEITHER ATTENTION
                                        ;*      BIT IS SET.
                                        ;*
                                        ;;*********************************************************************
005102                                  TST3:
005102  005737  001266                          TST     KYBCTL          ;PERFORMING ONLY SINGLE TESTS ?
005106  001406                                  BEQ     2$              ;BR IF NOT
005110  100002                                  BPL     1$              ;BR IF JUST ENTERED TEST
005112  000137  002424                          JMP     EXEC            ;RETURN & GET NEXT TEST NUMBER
005116  012737  177777  001266    1$:           MOV     #-1,KYBCTL      ;SET SINGLE TEST INDICATOR
005124  112737  000003  001102    2$:           MOVB    #3,$TSTNM       ;TEST NUMBER
005132  012737  005154  001106                  MOV     #TEST3,$LPADR   ;LOAD LOOP ON TEST ADDRESS
005140  012737  005154  001110                  MOV     #TEST3,$LPERR   ;LOAD LOOP ON ERROR ADDRESS
005146  012737  000001  001176                  MOV     #1,$TIMES       ;;DO 1 ITERATION
005154  012706  001100                TEST3:    MOV     #STACK,SP       ;SETUP THE STACK POINTER
005160  113760  001226  000010                  MOVB    PORTB,RMCS2(R0) ;SELECT PORT B
005166  013737  001226  001234                  MOV     PORTB,PTNBR  ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT

                                        ;;*********************************************************************
                                        ;SET VOLUME VALUE FOR PORT

005174  012760  000011  000000                  MOV     #11,RMCS1(R0)   ;ISSUE A DRIVE CLEAR
005202  012760  000021  000000                  MOV     #21,RMCS1(R0)   ;ISSUE A READIN PRESET
005210  012760  010000  000032                  MOV     #FMT16,RMOF(R0) ;SET FMT16

                                        ;;*********************************************************************
                                        ;VERIFY THAT THE DRIVE STATUS IS CORRECT

005216  005037  001244                          CLR     CKERR           ;CLEAR THE 'CHECK ERROR' INDICATOR
005222  016037  000012  001126                  MOV     RMDS1(R0),$BDDAT  ;GET CONTENTS OF RMDS1
005230  012737  000012  001122                  MOV     #RMDS1,$BDADR   ;FORM REGISTER ADDRESS OF ERROR MESSAGE
005236  060037  001122                          ADD     R0,$BDADR       ;ADD RH11 BASE ADDRESS
005242  012737  011700  001124                  MOV     #MOL!PGM!DPR!DRY!VV,$GDDAT  ;WHAT REGISTER SHOULD BE
005250  013737  001126  001164                  MOV     $BDDAT,$TMP0    ;MOVE REGISTER CONTENTS TO '$TMP0'
005256  042737  106077  001164                  BIC     #^C71700,$TMP0  ;SAVE SPECIFIED BITS
005264  023737  001124  001164                  CMP     $GDDAT,$TMP0    ;COMPARE THE BITS
005272  001414                                  BEQ     64$             ;BR IF OK
005274  013737  001126  001174                  MOV     $BDDAT,$TMP4    ;COPY 'BAD DATA'
005302  042737  071700  001174                  BIC     #71700,$TMP4    ;CLEAR THE MASKED BITS
005310  053737  001174  001124                  BIS     $TMP4,$GDDAT    ;'OR' WITH GOOD DATA FOR TYPEOUT
005316  104010                                  ERROR   10              ;TYPE MESSAGE 10
005320  005137  001244                          COM     CKERR           ;SET THE REGISTER COMPARE ERROR INDICATOR
005324  000240                        64$:      NOP

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

                                        ;RELEASE THE DRIVE FROM PORT B

005326  113760  001226  000010                  MOVB    PORTB,RMCS2(R0)  ;SELECT PORT B
005334  013737  001226  001234                  MOV     PORTB,PTNBR  ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
```

```
1818  005342  012760  000013  000000          MOV     #13,RMCS1(R0)   ;ISSUE RELEASE THROUGH PORT B
1819
1820                                          ;VERIFY THAT THE DRIVE IS IN NEUTRAL
1821
1822  005350  005037  001250                  CLR     RELERR          ;CLEAR THE 'RELEASE ERROR ' INDICATOR
1823  005354  012737  000012  001122          MOV     #RMDS1,$BDADR   ;FORM THE ADDRESS OF RMDS1 FOR TYPEOUT
1824  005362  060037  001122                  ADD     R0,$BDADR       ;ADD THE I/O BASE ADDRESS
1825  005366  012737  011600  001124          MOV     #MOL!PGM!DPR!DRY,$GDDAT  ;COMPARISON CONSTANT
1826  005374  113760  001224  000010          MOVB    PORTA,RMCS2(R0) ;SELECT PORT A.
1827  005402  016037  000012  001170          MOV     RMDS1(R0),$TMP2 ;GET THE DRIVE STATUS REGISTER FROM PORT A.
1828  005410  013737  001170  001164          MOV     $TMP2,$TMP0     ;COPY IT INTO '$TMP0'
1829  005416  042737  100100  001164          BIC     #ATA!VV,$TMP0   ;CLEAR PORT DEPENDENT BITS FROM THE COPY
1830  005424  113760  001226  000010          MOVB    PORTB,RMCS2(R0) ;SELECT PORT B.
1831  005432  016037  000012  001172          MOV     RMDS1(R0),$TMP3 ;GET THE DRIVE STATUS REGISTER FROM PORT B.
1832  005440  013737  001172  001166          MOV     $TMP3,$TMP1     ;COPY IT INTO '$TMP1'
1833  005446  042737  100100  001166          BIC     #ATA!VV,$TMP1   ;CLEAR PORT DEPENDENT BITS FROM THE COPY
1834  005454  023737  001164  001166          CMP     $TMP0,$TMP1     ;IS THE STATUS REGISTER THE SAME FROM BOTH PORTS ?
1835  005462  001006                          BNE     66$             ;BR IF NOT
1836  005464  005737  001164                  TST     $TMP0           ;REGISTERS ARE THE SAME: ARE THEY ZERO ?
1837  005470  001037                          BNE     68$             ;BR IF NOT
1838  005472  104034                          ERROR   34              ;REPORT DRIVE NOT IN NEUTRAL OR NOT SEIZED
1839  005474  000137  005660                  JMP     70$             ;BYPASS THE REST OF THE CHECKS
1840  005500  013737  001170  001126  66$:    MOV     $TMP2,$BDDAT    ;SET UP POSSIBLE BAD DATA FOR ERROR MESSAGE
1841  005506  013737  001226  001234          MOV     PORTB,PTNBR     ;SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
1842  005514  113760  001226  000010          MOVB    PORTB,RMCS2(R0) ;SELECT PORT B.
1843  005522  005737  001164                  TST     $TMP0           ;SEE IF STATUS EQ 0 FROM PORT A.
1844  005526  001414                          BEQ     67$             ;BR IF ZERO
1845  005530  013737  001224  001234          MOV     PORTA,PTNBR     ;SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
1846  005536  013737  001172  001126          MOV     $TMP3,$BDDAT    ;'BAD DATA' FOR ERROR TYPE OUT
1847  005544  113760  001224  000010          MOVB    PORTA,RMCS2(R0) ;SELECT PORT A.
1848  005552  005737  001166                  TST     $TMP1           ;SEE IF STATUS EQ ZERO FROM PORT B.
1849  005556  001004                          BNE     68$             ;BR IF NOT
1850  005560  012737  177777  001250  67$:    MOV     #-1,RELERR      ;SET 'RELEASE ERROR' INDICATOR
1851  005566  104036                          ERROR   36              ;TYPE ERROR MESSAGE 36
1852  005570  013737  001170  001126  68$:    MOV     $TMP2,$BDDAT    ;LOOK FOR BIT FAILURES WHEN RMDS1 READ
1853  005576  013737  001224  001234          MOV     PORTA,PTNBR     ;CHANGE PORT NUMBER
1854  005604  042737  100100  001170          BIC     #ATA!VV,$TMP2   ;DON'T CHECK ATTN BIT OR VV BIT
1855  005612  023737  001124  001170          CMP     $GDDAT,$TMP2    ;ALL BITS OK ?
1856  005620  001401                          BEQ     69$             ;BR IF OK FROM PORT A.
1857  005622  104037                          ERROR   37              ;REPORT ERROR
1858  005624  013737  001172  001126  69$:    MOV     $TMP3,$BDDAT    ;CHECK RMDS1 FOR BIT FAILURES - FROM PORT B.
1859  005632  013737  001226  001234          MOV     PORTB,PTNBR     ;CHANGE PORT NUMBER
1860  005640  042737  100100  001172          BIC     #ATA!VV,$TMP3   ;DON'T CHECK ATTN BIT OR VV BIT
1861  005646  023737  001124  001172          CMP     $GDDAT,$TMP3    ;SEE IF READ OK FROM PORT B.
1862  005654  001401                          BEQ     70$             ;BR IF OK
1863  005656  104037                          ERROR   37              ;REPORT THE ERROR
1864  005660  000240                  70$:    NOP
1865  005662  000004                          SCOPE                   ;LOOP ?
1866
1867
1868
1869
1870                                          ;;*****************************************************************
1871                                          ;*TEST 4        MEASURE THE TIMEOUT ONE-SHOT THROUGH PORT A
1872                                          ;*
1873                                          ;*MEASURE THE TIMEOUT ONE-SHOT VALUE THROUGH PORT A
```

```
                                        ;*
                                        ;*  A.  WRITE 0'S INTO RMDS1 THROUGH PORT A AND VERIFY THAT THE
                                        ;*      DRIVE HAS BEEN SEIZED.
                                        ;*
                                        ;*  B.  WAIT FOR TIMEOUT TO OCCUR.  MEASURE THE DURATION OF THE TIMEOUT
                                        ;*      ONE-SHOT AND SAVE THE VALUE FOR LATER USE.
                                        ;*
                                        ;*  C.  VERIFY THAT THE TIMEOUT OCCURRED AND THAT THE DRIVE RETURNS
                                        ;*      TO NEUTRAL
                                        ;*
                                        ;;*****************************************************************
005664                                  TST4:
005664  005737  001266                          TST     KYBCTL          ;PERFORMING ONLY SINGLE TESTS ?
005670  001406                                  BEQ     2$              ;BR IF NOT
005672  100002                                  BPL     1$              ;BR IF JUST ENTERED TEST
005674  000137  002424                          JMP     EXEC            ;RETURN & GET NEXT TEST NUMBER
005700  012737  177777  001266    1$:           MOV     #-1,KYBCTL      ;SET SINGLE TEST INDICATOR
005706  112737  000004  001102    2$:           MOVB    #4,$TSTNM       ;TEST NUMBER
005714  012737  005736  001106                  MOV     #TEST4,$LPADR   ;LOAD LOOP ON TEST ADDRESS
005722  012737  005736  001110                  MOV     #TEST4,$LPERR   ;LOAD LOOP ON ERROR ADDRESS
005730  012737  000001  001176                  MOV     #1,$TIMES       ;;DO 1 ITERATION
005736  012706  001100                TEST4:    MOV     #STACK,SP       ;SETUP THE STACK POINTER
005742  005037  001256                          CLR     TIMEA           ;CLEAR THE TIMEOUT VALUE STORAGE LOCATION
005746  005037  001260                          CLR     TIMEAP  ;CLEAR THE + 25% TOLERANCE LOCATION

                                        ;;*****************************************************************
                                        ;START THE TIMER

005752  005037  001252                          CLR     TIME            ;CLEAR THE ELAPSED TIME COUNTER
005756  012737  003720  001254                  MOV     #2000.,WATCH    ;SET WATCH TO 2000 MS

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

                                                ;SEIZE THE DRIVE THROUGH PORT A

005764  113760  001224  000010                  MOVB    PORTA,RMCS2(R0) ;SELECT PORT A
005772  013737  001224  001236                  MOV     PORTA,SEIZPT  ;STORE SEIZING PORT'S ADDRESS
006000  005060  000012                          CLR     RMDS1(R0)       ;WRITE RMDS1
006004  113760  001226  000010                  MOVB    PORTB,RMCS2(R0)  ;SELECT PORT B
006012  013737  001226  001234                  MOV     PORTB,PTNBR  ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
006020  013737  001226  001240                  MOV     PORTB,OPPRT     ;'OPPOSITE' PORT ADDRESS
006026  016037  000012  001126                  MOV     RMDS1(R0),$BDDAT  ;SEE IF DRIVE SEIZED BY PORT A
006034  010037  001122                          MOV     R0,$BDADR       ;RH11 BASE ADDRESS
006040  062737  000012  001122                  ADD     #RMDS1,$BDADR   ;GENERATE BAD REGISTER ADDRESS
006046  005037  001124                          CLR     $GDDAT          ;REGISTER SHOULD BE ZERO
006052  023737  001124  001126                  CMP     $GDDAT,$BDDAT   ;IS THE REGISTER ZERO
006060  001403                                  BEQ     64$             ;BR IF IT IS
006062  104030                                  ERROR   30              ;REPORT THE ERROR
006064  000137  006600                          JMP     4$              ;BYPASS REST OF THE SUBTEST
006070                                  64$:
006070  113760  001224  000010                  MOVB    PORTA,RMCS2(R0)  ;SELECT PORT A
006076  013737  001224  001234                  MOV     PORTA,PTNBR  ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
006104  016037  000012  001126                  MOV     RMDS1(R0),$BDDAT  ;SEE IF SEIZING PORT SEES CORRECT STATUS
006112  012737  011700  001124                  MOV     #MOL!PGM!DPR!DRY!VV,$GDDAT  ;EXPECTED STATUS
006120  013737  001124  001166                  MOV     $GDDAT,$TMP1  ;USE GOOD DATA AS A MASK
006126  005137  001166                          COM     $TMP1           ;COMPLEMENT THE EXPECTED STATUS
```

```
1930  006132  013737  001126  001164          MOV     $BDDAT,$TMP0    ;SAVE THE ACTUAL STATUS
1931  006140  043737  001166  001164          BIC     $TMP1,$TMP0     ;CLEAR UNWANTED BITS
1932  006146  023737  001124  001164          CMP     $GDDAT,$TMP0    ;ARE THE EXPECTED STATUS BITS SET ?
1933  006154  001401                          BEQ     65$             ;BR IF THEY ARE
1934  006156  104031                          ERROR   31              ;REPORT THE ERROR
1935  006160  000240                  65$:    NOP
1936
1937                                          ;;*****************************************************************
1938                                          ;WAIT FOR PORT A TO TIMEOUT
1939
1940  006162  113760  001226  000010          MOVB    PORTB,RMCS2(R0) ;SELECT PORT B
1941  006170  013737  001226  001234          MOV     PORTB,PTNBR     ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
1942  006176  005760  000012          1$:     TST     RMDS1(R0)       ;WAIT FOR THE DRIVE TO TIMEOUT
1943  006202  001006                          BNE     2$              ;BR WHEN TIMEOUT OCCURS
1944  006204  005737  001254                  TST     WATCH           ;CHECK WATCH
1945  006210  001372                          BNE     1$              ;BR IF NOT ZERO
1946  006212  104006                          ERROR   6               ;NO TIMEOUT WITHIN 2 SECONDS
1947  006214  000137  006252                  JMP     3$              ;BYPASS THE REST OF THE TEST
1948  006220  013737  001252  001256  2$:     MOV     TIME,TIMEA      ;SAVE THE ELAPSED TIME FOR PORT A
1949  006226  004537  013412                  JSR     R5,TOLER        ;CALCULATE THE TOLERANCE
1950  006232  001256                          .WORD   TIMEA           ;TIMEOUT VALUE FOR PORT A
1951  006234  012637  001260                  MOV     (SP)+,TIMEAP    ;+25% TOLERANCE
1952
1953                                          ;;*****************************************************************
1954                                          ;VERIFY THAT THE TIMEOUT ONE-SHOT VALUE IS AT LEAST 500 MS
1955
1956  006240  023727  001256  000764          CMP     TIMEA,#500.     ;IS TIMEOUT VALUE AT LEAST 500 MS ?
1957  006246  103001                          BHIS    3$              ;BR IF IT IS
1958  006250  104007                          ERROR   7               ;TIMEOUT LESS THAN 500 MS
1959
1960                                          ;;*****************************************************************
1961                                          ;VERIFY THAT THE DRIVE RETURNED TO NEUTRAL AFTER PORT A TIMED OUT
1962
1963  006252                          3$:
1964
1965                                          ;VERIFY THAT THE DRIVE IS IN NEUTRAL
1966
1967  006252  005037  001250                  CLR     RELERR          ;CLEAR THE 'RELEASE ERROR ' INDICATOR
1968  006256  012737  000012  001122          MOV     #RMDS1,$BDADR   ;FORM THE ADDRESS OF RMDS1 FOR TYPEOUT
1969  006264  060037  001122                  ADD     R0,$BDADR       ;ADD THE I/O BASE ADDRESS
1970  006270  012737  011700  001124          MOV     #MOL!PGM!DPR!DRY!VV,$GDDAT  ;COMPARISON CONSTANT
1971  006276  113760  001224  000010          MOVB    PORTA,RMCS2(R0) ;SELECT PORT A.
1972  006304  016037  000012  001170          MOV     RMDS1(R0),$TMP2 ;GET THE DRIVE STATUS REGISTER FROM PORT A.
1973  006312  013737  001170  001164          MOV     $TMP2,$TMP0     ;COPY IT INTO '$TMP0'
1974  006320  042737  100100  001164          BIC     #ATA!VV,$TMP0   ;CLEAR PORT DEPENDENT BITS FROM THE COPY
1975  006326  113760  001226  000010          MOVB    PORTB,RMCS2(R0) ;SELECT PORT B.
1976  006334  016037  000012  001172          MOV     RMDS1(R0),$TMP3 ;GET THE DRIVE STATUS REGISTER FROM PORT B.
1977  006342  013737  001172  001166          MOV     $TMP3,$TMP1     ;COPY IT INTO '$TMP1'
1978  006350  042737  100100  001166          BIC     #ATA!VV,$TMP1   ;CLEAR PORT DEPENDENT BITS FROM THE COPY
1979  006356  023737  001164  001166          CMP     $TMP0,$TMP1     ;IS THE STATUS REGISTER THE SAME FROM BOTH PORTS ?
1980  006364  001006                          BNE     66$             ;BR IF NOT
1981  006366  005737  001164                  TST     $TMP0           ;REGISTERS ARE THE SAME: ARE THEY ZERO ?
1982  006372  001045                          BNE     68$             ;BR IF NOT
1983  006374  104034                          ERROR   34              ;REPORT DRIVE NOT IN NEUTRAL OR NOT SEIZED
1984  006376  000137  006576                  JMP     70$             ;BYPASS THE REST OF THE CHECKS
1985  006402  013737  001170  001126  66$:    MOV     $TMP2,$BDDAT    ;SET UP POSSIBLE BAD DATA FOR ERROR MESSAGE
```

```
 1986  006410  013737  001226  001234          MOV     PORTB,PTNBR     ;SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
 1987  006416  113760  001226  000010          MOVB    PORTB,RMCS2(R0) ;SELECT PORT B.
 1988  006424  005737  001164                  TST     $TMP0           ;SEE IF STATUS EQ 0 FROM PORT A.
 1989  006430  001414                          BEQ     67$             ;BR IF ZERO
 1990  006432  013737  001224  001234          MOV     PORTA,PTNBR     ;SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
 1991  006440  013737  001172  001126          MOV     $TMP3,$BDDAT    ;'BAD DATA' FOR ERROR TYPE OUT
 1992  006446  113760  001224  000010          MOVB    PORTA,RMCS2(R0) ;SELECT PORT A.
 1993  006454  005737  001166                  TST     $TMP1           ;SEE IF STATUS EQ ZERO FROM PORT B.
 1994  006460  001012                          BNE     68$             ;BR IF NOT
 1995  006462  012737  177777  001250  67$:    MOV     #-1,RELERR      ;SET 'RELEASE ERROR' INDICATOR
 1996  006470  012760  000011  000000          MOV     #11,RMCS1(R0)   ;CLEAR THE DRIVE
 1997  006476  012760  000013  000000          MOV     #13,RMCS1(R0)   ;RELEASE THE DRIVE
 1998  006504  104035                          ERROR   35              ;TYPE ERROR MESSAGE 35
 1999  006506  013737  001170  001126  68$:    MOV     $TMP2,$BDDAT    ;LOOK FOR BIT FAILURES WHEN RMDS1 READ
 2000  006514  013737  001224  001234          MOV     PORTA,PTNBR     ;CHANGE PORT NUMBER
 2001  006522  042737  100000  001170          BIC     #ATA,$TMP2      ;DON'T CHECK THE ATTN BIT
 2002  006530  023737  001124  001170          CMP     $GDDAT,$TMP2    ;ALL BITS OK ?
 2003  006536  001401                          BEQ     69$             ;BR IF OK FROM PORT A.
 2004  006540  104037                          ERROR   37              ;REPORT ERROR
 2005  006542  013737  001172  001126  69$:    MOV     $TMP3,$BDDAT    ;CHECK RMDS1 FOR BIT FAILURES - FROM PORT B.
 2006  006550  013737  001226  001234          MOV     PORTB,PTNBR     ;CHANGE PORT NUMBER
 2007  006556  042737  100000  001172          BIC     #ATA,$TMP3      ;DON'T CHECK THE ATTN BIT
 2008  006564  023737  001124  001172          CMP     $GDDAT,$TMP3    ;SEE IF READ OK FROM PORT B.
 2009  006572  001401                          BEQ     70$             ;BR IF OK
 2010  006574  104037                          ERROR   37              ;REPORT THE ERROR
 2011  006576  000240                  70$:    NOP
 2012  006600  000004                  4$:     SCOPE                   ;LOOP ?
 2013
 2014
 2015                                          ;;*****************************************************************
 2016                                          ;*TEST 5        MEASURE THE TIMEOUT ONE-SHOT THROUGH PORT B
 2017                                          ;*
 2018                                          ;*MEASURE THE TIMEOUT ONE-SHOT VALUE THROUGH PORT B
 2019                                          ;*
 2020                                          ;*  A.  WRITE 0'S INTO RMDS1 THROUGH PORT B AND VERIFY THAT THE
 2021                                          ;*      DRIVE HAS BEEN SEIZED.
 2022                                          ;*
 2023                                          ;*  B.  WAIT FOR TIMEOUT TO OCCUR.  MEASURE THE DURATION OF THE TIMEOUT
 2024                                          ;*      ONE-SHOT AND SAVE THE VALUE FOR LATER USE.
 2025                                          ;*
 2026                                          ;*  C.  VERIFY THAT THE TIMEOUT OCCURRED AND THAT THE DRIVE RETURNS
 2027                                          ;*      TO NEUTRAL
 2028                                          ;*
 2029                                          ;;*****************************************************************
 2030  006602                          TST5:
 2031  006602  005737  001266                  TST     KYBCTL          ;PERFORMING ONLY SINGLE TESTS ?
 2032  006606  001406                          BEQ     2$              ;BR IF NOT
 2033  006610  100002                          BPL     1$              ;BR IF JUST ENTERED TEST
 2034  006612  000137  002424                  JMP     EXEC            ;RETURN & GET NEXT TEST NUMBER
 2035  006616  012737  177777  001266  1$:     MOV     #-1,KYBCTL      ;SET SINGLE TEST INDICATOR
 2036  006624  112737  000005  001102  2$:     MOVB    #5,$TSTNM       ;TEST NUMBER
 2037  006632  012737  006654  001106          MOV     #TEST5,$LPADR   ;LOAD LOOP ON TEST ADDRESS
 2038  006640  012737  006654  001110          MOV     #TEST5,$LPERR   ;LOAD LOOP ON ERROR ADDRESS
 2039  006646  012737  000001  001176          MOV     #1,$TIMES       ;;DO 1 ITERATION
 2040  006654  012706  001100          TEST5:  MOV     #STACK,SP       ;SETUP THE STACK POINTER
 2041  006660  005037  001262                  CLR     TIMEB           ;CLEAR THE TIMEOUT VALUE STORAGE LOCATION
```

```
002042  006664  005037  001264                  CLR     TIMEBP  ;CLEAR THE + 25% TOLERANCE LOCATION
002043
002044                                  ;;*****************************************************************
002045                                  ;START THE TIMER
002046
002047  006670  005037  001252                  CLR     TIME            ;CLEAR THE ELAPSED TIME COUNTER
002048  006674  012737  003720  001254          MOV     #2000.,WATCH    ;SET WATCH TO 2000 MS
002049
002050                                  ;;*****************************************************************
002051
002052                                  ;SEIZE THE DRIVE THROUGH PORT B
002053
002054  006702  113760  001226  000010          MOVB    PORTB,RMCS2(R0) ;SELECT PORT B
002055  006710  013737  001226  001236          MOV     PORTB,SEIZPT    ;STORE SEIZING PORT'S ADDRESS
002056  006716  005060  000012                  CLR     RMDS1(R0)       ;WRITE RMDS1
002057  006722  113760  001224  000010          MOVB    PORTA,RMCS2(R0) ;SELECT PORT A
002058  006730  013737  001224  001234          MOV     PORTA,PTNBR     ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
002059  006736  013737  001224  001240          MOV     PORTA,OPPRT     ;'OPPOSITE' PORT ADDRESS
002060  006744  016037  000012  001126          MOV     RMDS1(R0),$BDDAT ;SEE IF DRIVE SEIZED BY PORT B
002061  006752  010037  001122                  MOV     R0,$BDADR       ;RH11 BASE ADDRESS
002062  006756  062737  000012  001122          ADD     #RMDS1,$BDADR   ;GENERATE BAD REGISTER ADDRESS
002063  006764  005037  001124                  CLR     $GDDAT          ;REGISTER SHOULD BE ZERO
002064  006770  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;IS THE REGISTER ZERO
002065  006776  001403                          BEQ     64$             ;BR IF IT IS
002066  007000  104030                          ERROR   30              ;REPORT THE ERROR
002067  007002  000137  007516                  JMP     4$              ;BYPASS REST OF THE SUBTEST
002068  007006                          64$:
002069  007006  113760  001226  000010          MOVB    PORTB,RMCS2(R0) ;SELECT PORT B
002070  007014  013737  001226  001234          MOV     PORTB,PTNBR     ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
002071  007022  016037  000012  001126          MOV     RMDS1(R0),$BDDAT ;SEE IF SEIZING PORT SEES CORRECT STATUS
002072  007030  012737  011700  001124          MOV     #MOL!PGM!DPR!DRY!VV,$GDDAT ;EXPECTED STATUS
002073  007036  013737  001124  001166          MOV     $GDDAT,$TMP1    ;USE GOOD DATA AS A MASK
002074  007044  005137  001166                  COM     $TMP1           ;COMPLEMENT THE EXPECTED STATUS
002075  007050  013737  001126  001164          MOV     $BDDAT,$TMP0    ;SAVE THE ACTUAL STATUS
002076  007056  043737  001166  001164          BIC     $TMP1,$TMP0     ;CLEAR UNWANTED BITS
002077  007064  023737  001124  001164          CMP     $GDDAT,$TMP0    ;ARE THE EXPECTED STATUS BITS SET ?
002078  007072  001401                          BEQ     65$             ;BR IF THEY ARE
002079  007074  104031                          ERROR   31              ;REPORT THE ERROR
002080  007076  000240                  65$:    NOP
002081
002082                                  ;;*****************************************************************
002083                                  ;WAIT FOR PORT B TO TIMEOUT
002084
002085  007100  113760  001224  000010          MOVB    PORTA,RMCS2(R0) ;SELECT PORT A
002086  007106  013737  001224  001234          MOV     PORTA,PTNBR     ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
002087  007114  005760  000012          1$:     TST     RMDS1(R0)       ;WAIT FOR THE DRIVE TO TIMEOUT
002088  007120  001006                          BNE     2$              ;BR WHEN TIMEOUT OCCURS
002089  007122  005737  001254                  TST     WATCH           ;CHECK WATCH
002090  007126  001372                          BNE     1$              ;BR IF NOT ZERO
002091  007130  104006                          ERROR   6               ;NO TIMEOUT WITHIN 2 SECONDS
002092  007132  000137  007170                  JMP     3$              ;BYPASS THE REST OF THE TEST
002093  007136  013737  001252  001262  2$:     MOV     TIME,TIMEB      ;SAVE THE ELAPSED TIME FOR PORT B
002094  007144  004537  013412                  JSR     R5,TOLER        ;CALCULATE THE TOLERANCE
002095  007150  001262                          .WORD   TIMEB           ;TIMEOUT VALUE FOR PORT B
002096  007152  012637  001264                  MOV     (SP)+,TIMEBP    ;+25% TOLERANCE
002097
```

```
                                ;;*****************************************************************
                                ;VERIFY THAT THE TIMEOUT ONE-SHOT VALUE IS AT LEAST 500 MS

007156  023727  001262  000764          CMP     TIMEB,#500.     ;IS TIMEOUT VALUE AT LEAST 500 MS ?
007164  103001                          BHIS    3$              ;BR IF IT IS
007166  104007                          ERROR   7               ;TIMEOUT LESS THAN 500 MS

                                ;;*****************************************************************
                                ;VERIFY THAT THE DRIVE RETURNED TO NEUTRAL AFTER PORT B TIMED OUT

007170                          3$:

                                        ;VERIFY THAT THE DRIVE IS IN NEUTRAL

007170  005037  001250                  CLR     RELERR          ;CLEAR THE 'RELEASE ERROR ' INDICATOR
007174  012737  000012  001122          MOV     #RMDS1,$BDADR   ;FORM THE ADDRESS OF RMDS1 FOR TYPEOUT
007202  060037  001122                  ADD     R0,$BDADR       ;ADD THE I/O BASE ADDRESS
007206  012737  011700  001124          MOV     #MOL!PGM!DPR!DRY!VV,$GDDAT  ;COMPARISON CONSTANT
007214  113760  001224  000010          MOVB    PORTA,RMCS2(R0) ;SELECT PORT A.
007222  016037  000012  001170          MOV     RMDS1(R0),$TMP2 ;GET THE DRIVE STATUS REGISTER FROM PORT A.
007230  013737  001170  001164          MOV     $TMP2,$TMP0     ;COPY IT INTO '$TMP0'
007236  042737  100100  001164          BIC     #ATA!VV,$TMP0   ;CLEAR PORT DEPENDENT BITS FROM THE COPY
007244  113760  001226  000010          MOVB    PORTB,RMCS2(R0) ;SELECT PORT B.
007252  016037  000012  001172          MOV     RMDS1(R0),$TMP3 ;GET THE DRIVE STATUS REGISTER FROM PORT B.
007260  013737  001172  001166          MOV     $TMP3,$TMP1     ;COPY IT INTO '$TMP1'
007266  042737  100100  001166          BIC     #ATA!VV,$TMP1   ;CLEAR PORT DEPENDENT BITS FROM THE COPY
007274  023737  001164  001166          CMP     $TMP0,$TMP1     ;IS THE STATUS REGISTER THE SAME FROM BOTH PORTS ?
007302  001006                          BNE     66$             ;BR IF NOT
007304  005737  001164                  TST     $TMP0           ;REGISTERS ARE THE SAME: ARE THEY ZERO ?
007310  001045                          BNE     68$             ;BR IF NOT
007312  104034                          ERROR   34              ;REPORT DRIVE NOT IN NEUTRAL OR NOT SEIZED
007314  000137  007514                  JMP     70$             ;BYPASS THE REST OF THE CHECKS
007320  013737  001170  001126  66$:    MOV     $TMP2,$BDDAT    ;SET UP POSSIBLE BAD DATA FOR ERROR MESSAGE
007326  013737  001226  001234          MOV     PORTB,PTNBR     ;SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
007334  113760  001226  000010          MOVB    PORTB,RMCS2(R0) ;SELECT PORT B.
007342  005737  001164                  TST     $TMP0           ;SEE IF STATUS EQ 0 FROM PORT A.
007346  001414                          BEQ     67$             ;BR IF ZERO
007350  013737  001224  001234          MOV     PORTA,PTNBR     ;SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
007356  013737  001172  001126          MOV     $TMP3,$BDDAT    ;'BAD DATA' FOR ERROR TYPE OUT
007364  113760  001224  000010          MOVB    PORTA,RMCS2(R0) ;SELECT PORT A.
007372  005737  001166                  TST     $TMP1           ;SEE IF STATUS EQ ZERO FROM PORT B.
007376  001012                          BNE     68$             ;BR IF NOT
007400  012737  177777  001250  67$:    MOV     #-1,RELERR      ;SET 'RELEASE ERROR' INDICATOR
007406  012760  000011  000000          MOV     #11,RMCS1(R0)   ;CLEAR THE DRIVE
007414  012760  000013  000000          MOV     #13,RMCS1(R0)   ;RELEASE THE DRIVE
007422  104035                          ERROR   35              ;TYPE ERROR MESSAGE 35
007424  013737  001170  001126  68$:    MOV     $TMP2,$BDDAT    ;LOOK FOR BIT FAILURES WHEN RMDS1 READ
007432  013737  001224  001234          MOV     PORTA,PTNBR     ;CHANGE PORT NUMBER
007440  042737  100000  001170          BIC     #ATA,$TMP2      ;DON'T CHECK THE ATTN BIT
007446  023737  001124  001170          CMP     $GDDAT,$TMP2    ;ALL BITS OK ?
007454  001401                          BEQ     69$             ;BR IF OK FROM PORT A.
007456  104037                          ERROR   37              ;REPORT ERROR
007460  013737  001172  001126  69$:    MOV     $TMP3,$BDDAT    ;CHECK RMDS1 FOR BIT FAILURES - FROM PORT B.
007466  013737  001226  001234          MOV     PORTB,PTNBR     ;CHANGE PORT NUMBER
007474  042737  100000  001172          BIC     #ATA,$TMP3      ;DON'T CHECK THE ATTN BIT
007502  023737  001124  001172          CMP     $GDDAT,$TMP3    ;SEE IF READ OK FROM PORT B.
```

```
                                        BEQ     70$             ;BR IF OK
                                        ERROR   37              ;REPORT THE ERROR
                                70$:    NOP
                                4$:     SCOPE                   ;LOOP ?


                                ;;*****************************************************************
                                ;*TEST 6        TEST 'CONTROLLER SELECT' SWITCH, DRIVE CYCLED UP
                                ;*
                                ;*TEST THE OPERATION OF THE 'CONTROLLER SELECT' SWITCH (DRIVE CYCLED UP).
                                ;*
                                ;*  A.  SWITCH TO CONTROLLER 'A' POSITION.  VERIFY THAT THE DRIVE IS IN
                                ;*      NEUTRAL AND THAT THE STATUS BITS IN RMDS1, AS READ THROUGH BOTH
                                ;*      PORTS, ARE CORRECT.
                                ;*
                                ;*  B.  SWITCH TO CONTROLLER 'B' POSITION. VERIFY THAT THE DRIVE IS IN
                                ;*      NEUTRAL AND THAT THE STATUS BITS IN RMDS1, AS READ THROUGH BOTH
                                ;*      PORTS, ARE CORRECT.
                                ;*
                                ;*  C.  RETURN THE 'CONTROLLER SELECT' SWITCH TO THE 'A/B' POSITION.  VERIFY
                                ;*      THE DRIVE STATE.
                                ;*
                                ;;*****************************************************************
007520                          TST6:
007520  005737  001266                  TST     KYBCTL          ;PERFORMING ONLY SINGLE TESTS ?
007524  001406                          BEQ     2$              ;BR IF NOT
007526  100002                          BPL     1$              ;BR IF JUST ENTERED TEST
007530  000137  002424                  JMP     EXEC            ;RETURN & GET NEXT TEST NUMBER
007534  012737  177777  001266  1$:     MOV     #-1,KYBCTL      ;SET SINGLE TEST INDICATOR
007542  112737  000006  001102  2$:     MOVB    #6,$TSTNM       ;TEST NUMBER
007550  012737  007572  001106          MOV     #TEST6,$LPADR   ;LOAD LOOP ON TEST ADDRESS
007556  012737  007572  001110          MOV     #TEST6,$LPERR   ;LOAD LOOP ON ERROR ADDRESS
007564  012737  000001  001176          MOV     #1,$TIMES       ;DO 1 ITERATION
007572  012706  001100          TEST6:  MOV     #STACK,SP       ;SETUP THE STACK POINTER

                                ;CLEAR ATTENTION BITS FOR BOTH PORTS

007576  113760  001224  000010          MOVB    PORTA,RMCS2(R0) ;SELECT PORT #A
007604  005060  000012                  CLR     RMDS1(R0)       ;SEIZE THE DRIVE
007610  012760  000011  000000          MOV     #11,RMCS1(R0)   ;ISSUE DRIVE CLEAR
007616  012760  000013  000000          MOV     #13,RMCS1(R0)   ;RELEASE THE DRIVE
007624  113760  001226  000010          MOVB    PORTB,RMCS2(R0) ;SELECT PORT #B
007632  005060  000012                  CLR     RMDS1(R0)       ;SEIZE THE DRIVE THROUGH PORT 'B'
007636  012760  000011  000000          MOV     #11,RMCS1(R0)   ;ISSUE DRIVE CLEAR
007644  012760  000013  000000          MOV     #13,RMCS1(R0)   ;RELEASE THE DRIVE
007652  104401  017402                  TYPE    ,SWTCHA         ;SWITCH TO 'A'
007656  104401  017550                  TYPE    ,CONTUE         ;PRESS 'CONTINUE'
007662  000000                          HALT

                                ;VERIFY THAT THE DRIVE IS IN NEUTRAL

007664  005037  001250                  CLR     RELERR          ;CLEAR THE 'RELEASE ERROR ' INDICATOR
007670  012737  000012  001122          MOV     #RMDS1,$BDADR   ;FORM THE ADDRESS OF RMDS1 FOR TYPEOUT
007676  060037  001122                  ADD     R0,$BDADR       ;ADD THE I/O BASE ADDRESS
007702  012737  011700  001124          MOV     #MOL!PGM!DPR!DRY!VV,$GDDAT  ;COMPARISON CONSTANT
```

```
2210  007710  113760  001224  000010          MOVB    PORTA,RMCS2(R0) ;SELECT PORT A.
2211  007716  016037  000012  001170          MOV     RMDS1(R0),$TMP2 ;GET THE DRIVE STATUS REGISTER FROM PORT A.
2212  007724  013737  001170  001164          MOV     $TMP2,$TMP0     ;COPY IT INTO '$TMP0'
2213  007732  042737  100100  001164          BIC     #ATA!VV,$TMP0   ;CLEAR PORT DEPENDENT BITS FROM THE COPY
2214  007740  113760  001226  000010          MOVB    PORTB,RMCS2(R0) ;SELECT PORT B.
2215  007746  016037  000012  001172          MOV     RMDS1(R0),$TMP3 ;GET THE DRIVE STATUS REGISTER FROM PORT B.
2216  007754  013737  001172  001166          MOV     $TMP3,$TMP1     ;COPY IT INTO '$TMP1'
2217  007762  042737  100100  001166          BIC     #ATA!VV,$TMP1   ;CLEAR PORT DEPENDENT BITS FROM THE COPY
2218  007770  023737  001164  001166          CMP     $TMP0,$TMP1     ;IS THE STATUS REGISTER THE SAME FROM BOTH PORTS ?
2219  007776  001006                          BNE     64$             ;BR IF NOT
2220  010000  005737  001164                  TST     $TMP0           ;REGISTERS ARE THE SAME: ARE THEY ZERO ?
2221  010004  001045                          BNE     66$             ;BR IF NOT
2222  010006  104034                          ERROR   34              ;REPORT DRIVE NOT IN NEUTRAL OR NOT SEIZED
2223  010010  000137  010210                  JMP     68$             ;BYPASS THE REST OF THE CHECKS
2224  010014  013737  001170  001126  64$:    MOV     $TMP2,$BDDAT    ;SET UP POSSIBLE BAD DATA FOR ERROR MESSAGE
2225  010022  013737  001226  001234          MOV     PORTB,PTNBR     ;SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
2226  010030  113760  001226  000010          MOVB    PORTB,RMCS2(R0) ;SELECT PORT B.
2227  010036  005737  001164                  TST     $TMP0           ;SEE IF STATUS EQ 0 FROM PORT A.
2228  010042  001414                          BEQ     65$             ;BR IF ZERO
2229  010044  013737  001224  001234          MOV     PORTA,PTNBR     ;SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
2230  010052  013737  001172  001126          MOV     $TMP3,$BDDAT    ;'BAD DATA' FOR ERROR TYPE OUT
2231  010060  113760  001224  000010          MOVB    PORTA,RMCS2(R0) ;SELECT PORT A.
2232  010066  005737  001166                  TST     $TMP1           ;SEE IF STATUS EQ ZERO FROM PORT B.
2233  010072  001012                          BNE     66$             ;BR IF NOT
2234  010074  012737  177777  001250  65$:    MOV     #-1,RELERR      ;SET 'RELEASE ERROR' INDICATOR
2235  010102  012760  000011  000000          MOV     #11,RMCS1(R0)   ;CLEAR THE DRIVE
2236  010110  012760  000013  000000          MOV     #13,RMCS1(R0)   ;RELEASE THE DRIVE
2237  010116  104017                          ERROR   17              ;TYPE ERROR MESSAGE 17
2238  010120  013737  001170  001126  66$:    MOV     $TMP2,$BDDAT    ;LOOK FOR BIT FAILURES WHEN RMDS1 READ
2239  010126  013737  001224  001234          MOV     PORTA,PTNBR     ;CHANGE PORT NUMBER
2240  010134  042737  100000  001170          BIC     #ATA,$TMP2      ;DON'T CHECK THE ATTN BIT
2241  010142  023737  001124  001170          CMP     $GDDAT,$TMP2    ;ALL BITS OK ?
2242  010150  001401                          BEQ     67$             ;BR IF OK FROM PORT A.
2243  010152  104037                          ERROR   37              ;REPORT ERROR
2244  010154  013737  001172  001126  67$:    MOV     $TMP3,$BDDAT    ;CHECK RMDS1 FOR BIT FAILURES - FROM PORT B.
2245  010162  013737  001226  001234          MOV     PORTB,PTNBR     ;CHANGE PORT NUMBER
2246  010170  042737  100000  001172          BIC     #ATA,$TMP3      ;DON'T CHECK THE ATTN BIT
2247  010176  023737  001124  001172          CMP     $GDDAT,$TMP3    ;SEE IF READ OK FROM PORT B.
2248  010204  001401                          BEQ     68$             ;BR IF OK
2249  010206  104037                          ERROR   37              ;REPORT THE ERROR
2250  010210  000240                  68$:    NOP
2251  010212  104401  017465                  TYPE    ,SWTCHB         ;SWITCH TO 'B'
2252  010216  104401  017550                  TYPE    ,CONTUE         ;PRESS 'CONTINUE'
2253  010222  000000                          HALT
2254
2255                                          ;VERIFY THAT THE DRIVE IS IN NEUTRAL
2256
2257  010224  005037  001250                  CLR     RELERR          ;CLEAR THE 'RELEASE ERROR ' INDICATOR
2258  010230  012737  000012  001122          MOV     #RMDS1,$BDADR   ;FORM THE ADDRESS OF RMDS1 FOR TYPEOUT
2259  010236  060037  001122                  ADD     R0,$BDADR       ;ADD THE I/O BASE ADDRESS
2260  010242  012737  011700  001124          MOV     #MOL!PGM!DPR!DRY!VV,$GDDAT  ;COMPARISON CONSTANT
2261  010250  113760  001224  000010          MOVB    PORTA,RMCS2(R0) ;SELECT PORT A.
2262  010256  016037  000012  001170          MOV     RMDS1(R0),$TMP2 ;GET THE DRIVE STATUS REGISTER FROM PORT A.
2263  010264  013737  001170  001164          MOV     $TMP2,$TMP0     ;COPY IT INTO '$TMP0'
2264  010272  042737  100100  001164          BIC     #ATA!VV,$TMP0   ;CLEAR PORT DEPENDENT BITS FROM THE COPY
2265  010300  113760  001226  000010          MOVB    PORTB,RMCS2(R0) ;SELECT PORT B.
```

```
2266  010306  016037  000012  001172          MOV     RMDS1(R0),$TMP3 ;GET THE DRIVE STATUS REGISTER FROM PORT B.
2267  010314  013737  001172  001166          MOV     $TMP3,$TMP1     ;COPY IT INTO '$TMP1'
2268  010322  042737  100100  001166          BIC     #ATA!VV,$TMP1   ;CLEAR PORT DEPENDENT BITS FROM THE COPY
2269  010330  023737  001164  001166          CMP     $TMP0,$TMP1     ;IS THE STATUS REGISTER THE SAME FROM BOTH PORTS ?
2270  010336  001006                          BNE     69$             ;BR IF NOT
2271  010340  005737  001164                  TST     $TMP0           ;REGISTERS ARE THE SAME: ARE THEY ZERO ?
2272  010344  001045                          BNE     71$             ;BR IF NOT
2273  010346  104034                          ERROR   34              ;REPORT DRIVE NOT IN NEUTRAL OR NOT SEIZED
2274  010350  000137  010550                  JMP     73$             ;BYPASS THE REST OF THE CHECKS
2275  010354  013737  001170  001126  69$:    MOV     $TMP2,$BDDAT    ;SET UP POSSIBLE BAD DATA FOR ERROR MESSAGE
2276  010362  013737  001226  001234          MOV     PORTB,PTNBR     ;SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
2277  010370  113760  001226  000010          MOVB    PORTB,RMCS2(R0) ;SELECT PORT B.
2278  010376  005737  001164                  TST     $TMP0           ;SEE IF STATUS EQ 0 FROM PORT A.
2279  010402  001414                          BEQ     70$             ;BR IF ZERO
2280  010404  013737  001224  001234          MOV     PORTA,PTNBR     ;SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
2281  010412  013737  001172  001126          MOV     $TMP3,$BDDAT    ;'BAD DATA' FOR ERROR TYPE OUT
2282  010420  113760  001224  000010          MOVB    PORTA,RMCS2(R0) ;SELECT PORT A.
2283  010426  005737  001166                  TST     $TMP1           ;SEE IF STATUS EQ ZERO FROM PORT B.
2284  010432  001012                          BNE     71$             ;BR IF NOT
2285  010434  012737  177777  001250  70$:    MOV     #-1,RELERR      ;SET 'RELEASE ERROR' INDICATOR
2286  010442  012760  000011  000000          MOV     #11,RMCS1(R0)   ;CLEAR THE DRIVE
2287  010450  012760  000013  000000          MOV     #13,RMCS1(R0)   ;RELEASE THE DRIVE
2288  010456  104020                          ERROR   20              ;TYPE ERROR MESSAGE 20
2289  010460  013737  001170  001126  71$:    MOV     $TMP2,$BDDAT    ;LOOK FOR BIT FAILURES WHEN RMDS1 READ
2290  010466  013737  001224  001234          MOV     PORTA,PTNBR     ;CHANGE PORT NUMBER
2291  010474  042737  100000  001170          BIC     #ATA,$TMP2      ;DON'T CHECK THE ATTN BIT
2292  010502  023737  001124  001170          CMP     $GDDAT,$TMP2    ;ALL BITS OK ?
2293  010510  001401                          BEQ     72$             ;BR IF OK FROM PORT A.
2294  010512  104037                          ERROR   37              ;REPORT ERROR
2295  010514  013737  001172  001126  72$:    MOV     $TMP3,$BDDAT    ;CHECK RMDS1 FOR BIT FAILURES - FROM PORT B.
2296  010522  013737  001226  001234          MOV     PORTB,PTNBR     ;CHANGE PORT NUMBER
2297  010530  042737  100000  001172          BIC     #ATA,$TMP3      ;DON'T CHECK THE ATTN BIT
2298  010536  023737  001124  001172          CMP     $GDDAT,$TMP3    ;SEE IF READ OK FROM PORT B.
2299  010544  001401                          BEQ     73$             ;BR IF OK
2300  010546  104037                          ERROR   37              ;REPORT THE ERROR
2301  010550  000240                  73$:    NOP
2302  010552  005737  001266                  TST     KYBCTL          ;SINGLE TEST MODE ?
2303  010556  001402                          BEQ     1$              ;BR IF NOT
2304  010560  104401  017313                  TYPE    ,SWTCHN         ;RETURN SWITCH TO 'A/B'
2305  010564  000004                  1$:     SCOPE                   ;LOOP ?
2306
2307
2308
2309                                  ;;*****************************************************************
2310                                  ;*TEST 7        TEST 'CONTROLLER SELECT' SWITCH LOCKED ON PORT A
2311                                  ;*
2312                                  ;*TEST THE OPERATION OF THE 'CONTROLLER SELECT' SWITCH (DRIVE CYCLED DOWN).
2313                                  ;*
2314                                  ;*  A.  CYCLE THE DRIVE DOWN.
2315                                  ;*
2316                                  ;*  B.  SWITCH TO CONTROLLER A POSITION.  VERIFY THAT THE DRIVE IS IN
2317                                  ;*      NEUTRAL AND THAT THE STATUS BITS IN RMDS1, AS READ THROUGH BOTH
2318                                  ;*      PORTS, ARE CORRECT.
2319                                  ;*
2320                                  ;*  C.  SWITCH THE 'CONTROLLER SELECT' SWITCH TO A; CYCLE THE DRIVE UP.
2321                                  ;*
```

```
                                ;*  D.  WHEN THE DRIVE CYCLES UP, VERIFY THAT 'VV-A IS RESET, AND
                                ;*      THAT 'ATA-A IS SET.
                                ;*
                                ;*  E.  ISSUE A DRIVE CLEAR COMMAND AND A READIN PRESET COMMAND THROUGH
                                ;*      PORT A.
                                ;*
                                ;*  F.  VERIFY THAT THE DRIVE CANNOT BE ACCESSED THROUGH PORT B AND
                                ;*      'NED' SETS WHEN ATEMPTING TO ACCESS THE DRIVE THROUGH
                                ;*      PORT B.  ATTEMPT TO SET PORT REQUEST BY WRITING 0'S
                                ;*      INTO RMDS1 THROUGH PORT B.
                                ;*
                                ;*  G.  ISSUE A RELEASE COMMAND THROUGH PORT A.  VERIFY THAT THE
                                ;*      DRIVE REMAINS LOCKED ON PORT A.
                                ;*
                                ;;*****************************************************************
010566                          TST7:
010566  005737  001266                  TST     KYBCTL          ;PERFORMING ONLY SINGLE TESTS ?
010572  001406                          BEQ     2$              ;BR IF NOT
010574  100002                          BPL     1$              ;BR IF JUST ENTERED TEST
010576  000137  002424                  JMP     EXEC            ;RETURN & GET NEXT TEST NUMBER
010602  012737  177777  001266  1$:     MOV     #-1,KYBCTL      ;SET SINGLE TEST INDICATOR
010610  112737  000007  001102  2$:     MOVB    #7,$TSTNM       ;TEST NUMBER
010616  012737  010640  001106          MOV     #TEST7,$LPADR   ;LOAD LOOP ON TEST ADDRESS
010624  012737  010640  001110          MOV     #TEST7,$LPERR   ;LOAD LOOP ON ERROR ADDRESS
010632  012737  000001  001176          MOV     #1,$TIMES       ;;DO 1 ITERATION
010640  012706  001100          TEST7:  MOV     #STACK,SP       ;SETUP THE STACK POINTER
010644  113760  001224  000010          MOVB    PORTA,RMCS2(R0) ;SELECT PORT A
010652  013737  001224  001234          MOV     PORTA,PTNBR  ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
010660  104401  017617                  TYPE    ,CYCLED         ;'CYCLE DOWN THE DRIVE'

                                ;;*****************************************************************
                                ;WAIT FOR 'MOL' TO RESET

010664  032760  010000  000012  1$:     BIT     #MOL,RMDS1(R0)  ;TEST 'MOL'
010672  001374                          BNE     1$              ;BR IF IT IS STILL SET
010674  104401  017402                  TYPE    ,SWTCHA         ;SWITCH TO 'A'
010700  104401  017641                  TYPE    ,CYCLEU         ;'CYCLE UP THE DRIVE'

                                ;;*****************************************************************
                                ;WAIT FOR DRIVE TO CYCLE UP AFTER SWITCH CHANGED

010704  032760  010000  000012  2$:     BIT     #MOL,RMDS1(R0)  ;TEST 'MOL' AGAIN
010712  001774                          BEQ     2$              ;BR IF NOT SET

                                ;;*****************************************************************
                                ;DRIVE IS CYCLED UP, CHECK STATUS THROUGH PORT A

010714  005037  001244                  CLR     CKERR           ;CLEAR THE 'CHECK ERROR' INDICATOR
010720  016037  000012  001126          MOV     RMDS1(R0),$BDDAT  ;GET CONTENTS OF RMDS1
010726  012737  000012  001122          MOV     #RMDS1,$BDADR   ;FORM REGISTER ADDRESS OF ERROR MESSAGE
010734  060037  001122                  ADD     R0,$BDADR       ;ADD RH11 BASE ADDRESS
010740  012737  110600  001124          MOV     #ATA!MOL!DPR!DRY,$GDDAT  ;WHAT REGISTER SHOULD BE
010746  013737  001126  001164          MOV     $BDDAT,$TMP0    ;MOVE REGISTER CONTENTS TO '$TMP0'
010754  042737  066077  001164          BIC     #^C111700,$TMP0  ;SAVE SPECIFIED BITS
010762  023737  001124  001164          CMP     $GDDAT,$TMP0    ;COMPARE THE BITS
010770  001414                          BEQ     64$             ;BR IF OK
```

```
2378  010772  013737  001126  001174          MOV     $BDDAT,$TMP4    ;COPY 'BAD DATA'
2379  011000  042737  111700  001174          BIC     #111700,$TMP4   ;CLEAR THE MASKED BITS
2380  011006  053737  001174  001124          BIS     $TMP4,$GDDAT    ;'OR' WITH GOOD DATA FOR TYPEOUT
2381  011014  104021                          ERROR   21              ;TYPE MESSAGE 21
2382  011016  005137  001244                  COM     CKERR           ;SET THE REGISTER COMPARE ERROR INDICATOR
2383  011022  000240                  64$:    NOP
2384
2385                                          ;;******************************************************************
2386                                          ;SET VOLUME VALID FOR PORT A
2387
2388  011024  012760  000011  000000          MOV     #11,RMCS1(R0)   ;ISSUE A DRIVE CLEAR
2389  011032  012760  000021  000000          MOV     #21,RMCS1(R0)   ;ISSUE A READIN PRESET
2390  011040  012760  010000  000032          MOV     #FMT16,RMOF(R0) ;SET FMT16
2391
2392                                          ;;******************************************************************
2393                                          ;CHECK THE DRIVE STATUS THROUGH PORT B; VERIFY THAT 'NED'
2394                                          ;       SETS WHEN THE DRIVE IS ACCESSED THROUGH PORT B.
2395
2396  011046  113760  001226  000010          MOVB    PORTB,RMCS2(R0) ;SELECT PORT B
2397  011054  013737  001226  001234          MOV     PORTB,PTNBR     ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
2398  011062  005037  001244                  CLR     CKERR           ;CLEAR THE 'CHECK ERROR' INDICATOR
2399  011066  016037  000012  001126          MOV     RMDS1(R0),$BDDAT  ;GET CONTENTS OF RMDS1
2400  011074  012737  000012  001122          MOV     #RMDS1,$BDADR   ;FORM REGISTER ADDRESS OF ERROR MESSAGE
2401  011102  060037  001122                  ADD     R0,$BDADR       ;ADD RH11 BASE ADDRESS
2402  011106  005037  001124                  CLR     $GDDAT          ;WHAT REGISTER SHOULD BE
2403  011112  013737  001126  001164          MOV     $BDDAT,$TMP0    ;MOVE REGISTER CONTENTS TO '$TMP0'
2404  011120  042737  000077  001164          BIC     #^C177700,$TMP0 ;SAVE SPECIFIED BITS
2405  011126  023737  001124  001164          CMP     $GDDAT,$TMP0    ;COMPARE THE BITS
2406  011134  001414                          BEQ     66$             ;BR IF OK
2407  011136  013737  001126  001174          MOV     $BDDAT,$TMP4    ;COPY 'BAD DATA'
2408  011144  042737  177700  001174          BIC     #177700,$TMP4   ;CLEAR THE MASKED BITS
2409  011152  053737  001174  001124          BIS     $TMP4,$GDDAT    ;'OR' WITH GOOD DATA FOR TYPEOUT
2410  011160  104022                          ERROR   22              ;TYPE MESSAGE 22
2411  011162  005137  001244                  COM     CKERR           ;SET THE REGISTER COMPARE ERROR INDICATOR
2412  011166  000240                  66$:    NOP
2413  011170  005037  001244                  CLR     CKERR           ;CLEAR THE 'CHECK ERROR' INDICATOR
2414  011174  016037  000010  001126          MOV     RMCS2(R0),$BDDAT  ;GET CONTENTS OF RMCS2
2415  011202  012737  000010  001122          MOV     #RMCS2,$BDADR   ;FORM REGISTER ADDRESS OF ERROR MESSAGE
2416  011210  060037  001122                  ADD     R0,$BDADR       ;ADD RH11 BASE ADDRESS
2417  011214  012737  010000  001124          MOV     #NED,$GDDAT     ;WHAT REGISTER SHOULD BE
2418  011222  013737  001126  001164          MOV     $BDDAT,$TMP0    ;MOVE REGISTER CONTENTS TO '$TMP0'
2419  011230  042737  167777  001164          BIC     #^CNED,$TMP0    ;SAVE SPECIFIED BITS
2420  011236  023737  001124  001164          CMP     $GDDAT,$TMP0    ;COMPARE THE BITS
2421  011244  001414                          BEQ     68$             ;BR IF OK
2422  011246  013737  001126  001174          MOV     $BDDAT,$TMP4    ;COPY 'BAD DATA'
2423  011254  042737  010000  001174          BIC     #NED,$TMP4      ;CLEAR THE MASKED BITS
2424  011262  053737  001174  001124          BIS     $TMP4,$GDDAT    ;'OR' WITH GOOD DATA FOR TYPEOUT
2425  011270  104023                          ERROR   23              ;TYPE MESSAGE 23
2426  011272  005137  001244                  COM     CKERR           ;SET THE REGISTER COMPARE ERROR INDICATOR
2427  011276  000240                  68$:    NOP
2428  011300  005060  000012                  CLR     RMDS1(R0)       ;TRY TO SET REQUEST BY WRITING THROUGH
2429                                                                  ;THE LOCKED OUT PORT (PORT 'B')
2430
2431                                          ;;******************************************************************
2432                                          ;VERIFY THAT DRIVE STAYS LOCKED ON PORT A
2433
```

```
2434  011304  113760  001224  000010          MOVB    PORTA,RMCS2(R0)  ;SELECT PORT A
2435  011312  013737  001224  001234          MOV     PORTA,PTNBR  ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
2436  011320  012760  000013  000012          MOV     #13,RMDS1(R0)    ;ISSUE A RELEASE THROUGH PORT A
2437  011326  013737  001224  001236          MOV     PORTA,SEIZPT     ;ADDRESS OF 'LOCKED ON' PORT
2438  011334  113760  001226  000010          MOVB    PORTB,RMCS2(R0)  ;SELECT PORT B
2439  011342  013737  001226  001234          MOV     PORTB,PTNBR  ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
2440  011350  005037  001244                  CLR     CKERR            ;CLEAR THE 'CHECK ERROR' INDICATOR
2441  011354  016037  000012  001126          MOV     RMDS1(R0),$BDDAT  ;GET CONTENTS OF RMDS1
2442  011362  012737  000012  001122          MOV     #RMDS1,$BDADR    ;FORM REGISTER ADDRESS OF ERROR MESSAGE
2443  011370  060037  001122                  ADD     R0,$BDADR        ;ADD RH11 BASE ADDRESS
2444  011374  005037  001124                  CLR     $GDDAT           ;WHAT REGISTER SHOULD BE
2445  011400  013737  001126  001164          MOV     $BDDAT,$TMP0     ;MOVE REGISTER CONTENTS TO '$TMP0'
2446  011406  042737  000077  001164          BIC     #^C177700,$TMP0  ;SAVE SPECIFIED BITS
2447  011414  023737  001124  001164          CMP     $GDDAT,$TMP0     ;COMPARE THE BITS
2448  011422  001414                          BEQ     70$              ;BR IF OK
2449  011424  013737  001126  001174          MOV     $BDDAT,$TMP4     ;COPY 'BAD DATA'
2450  011432  042737  177700  001174          BIC     #177700,$TMP4    ;CLEAR THE MASKED BITS
2451  011440  053737  001174  001124          BIS     $TMP4,$GDDAT     ;'OR' WITH GOOD DATA FOR TYPEOUT
2452  011446  104024                          ERROR   24               ;TYPE MESSAGE 24
2453  011450  005137  001244                  COM     CKERR            ;SET THE REGISTER COMPARE ERROR INDICATOR
2454  011454  000240                  70$:    NOP
2455
2456                                          ;IF ERROR OCCURRED, CHECK FOR LOOP ON TEST
2457
2458  011456  105737  001103                  TSTB    $ERFLG           ;DID AN ERROR OCCUR
2459  011462  001412                          BEQ     3$               ;BR IF NOT
2460  011464  032777  001000  167446          BIT     #SW09,@SWR       ;SEE IF LOOP ON ERROR (SWR9=1)
2461  011472  001406                          BEQ     3$               ;BR IF NOT
2462  011474  105037  001103                  CLRB    $ERFLG           ;CLEAR THE ERROR FLAG
2463  011500  005037  001176                  CLR     $TIMES           ;CLEAR THE MAX ITERATION COUNT
2464  011504  000177  167400                  JMP     @SLPERR          ;GO TO THE LOOP ADDRESS
2465  011510  005737  001266          3$:     TST     KYBCTL           ;IN SINGLE TEST MODE ?
2466  011514  001460                          BEQ     6$               ;BR IF NOT
2467  011516  032777  040000  167414          BIT     #SW14,@SWR       ;LOOP ON TEST ?
2468  011524  001054                          BNE     6$               ;BR IF LOOPING
2469  011526  104401  017617                  TYPE    ,CYCLED          ;TYPE 'CYCLE DOWN'
2470  011532  104401  017313                  TYPE    ,SWTCHN          ;'SWITCH TO A/B'
2471  011536  104401  017641                  TYPE    ,CYCLEU          ;'CYCLE THE DRIVE UP'
2472  011542  113760  001224  000010          MOVB    PORTA,RMCS2(R0)  ;SELECT PORT A
2473  011550  013737  001224  001234          MOV     PORTA,PTNBR  ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
2474  011556  032760  010000  000012  4$:     BIT     #MOL,RMDS1(R0)   ;CHECK 'MOL'
2475  011564  001374                          BNE     4$               ;BR IF SET (DRIVE NOT CYCLED DOWN)
2476  011566  032760  010000  000012  5$:     BIT     #MOL,RMDS1(R0)   ;CHECK 'MOL' AGAIN
2477  011574  001774                          BEQ     5$               ;BR IF NOT SET (DRIVE NOT CYCLED UP)
2478
2479                                          ;;*****************************************************************
2480                                          ;SET VOLUME VALID FOR BOTH PORTS
2481
2482  011576  012760  000011  000000          MOV     #11,RMCS1(R0)    ;ISSUE A DRIVE CLEAR THROUGH PORT A
2483  011604  012760  000021  000000          MOV     #21,RMCS1(R0)    ;ISSUE A READIN PRESET THROUGH PORT A
2484  011612  012760  000013  000000          MOV     #13,RMCS1(R0)    ;RELEASE PORT A
2485  011620  113760  001226  000010          MOVB    PORTB,RMCS2(R0)  ;SELECT PORT B
2486  011626  013737  001226  001234          MOV     PORTB,PTNBR  ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
2487  011634  012760  000021  000000          MOV     #21,RMCS1(R0)    ;ISSUE A READIN PRESET THROUGH PORT B
2488  011642  012760  010000  000032          MOV     #FMT16,RMOF(R0)  ;SET FMT16
2489  011650  012760  000013  000000          MOV     #13,RMCS1(R0)    ;RELEASE PORT B
```

```
2490  011656  104401  001207          6$:     TYPE    ,$CRLF          ;CR-LF
2491  011662  012737  011610  001254          MOV     #5000.,WATCH    ;SPINDLE MOTOR 'COOL DOWN' DELAY
2492  011670  005737  001254          7$:     TST     WATCH           ;FINSIHED ?
2493  011674  001375                          BNE     7$              ;BR IF NOT
2494  011676  000004                          SCOPE                   ;LOOP ?
2495  011700  000400                          BR      TST10           ;;GO TO NEXT TEST
2496
2497                                  ;;*****************************************************************
2498                                  ;*TEST 10       TEST 'CONTROLLER SELECT' SWITCH LOCKED ON PORT B
2499                                  ;*
2500                                  ;*TEST THE OPERATION OF THE 'CONTROLLER SELECT' SWITCH (DRIVE CYCLED DOWN).
2501                                  ;*
2502                                  ;*  A.  CYCLE THE DRIVE DOWN.
2503                                  ;*
2504                                  ;*  B.  SWITCH TO CONTROLLER B POSITION.  VERIFY THAT THE DRIVE IS IN
2505                                  ;*      NEUTRAL AND THAT THE STATUS BITS IN RMDS1, AS READ THROUGH BOTH
2506                                  ;*      PORTS, ARE CORRECT.
2507                                  ;*
2508                                  ;*  C.  SWITCH THE 'CONTROLLER SELECT' SWITCH TO B; CYCLE THE DRIVE UP.
2509                                  ;*
2510                                  ;*  D.  WHEN THE DRIVE CYCLES UP, VERIFY THAT 'VV-B IS RESET, AND
2511                                  ;*      THAT 'ATA-B IS SET.
2512                                  ;*
2513                                  ;*  E.  ISSUE A DRIVE CLEAR COMMAND AND A READIN PRESET COMMAND THROUGH
2514                                  ;*      PORT B.
2515                                  ;*
2516                                  ;*  F.  VERIFY THAT THE DRIVE CANNOT BE ACCESSED THROUGH PORT A AND
2517                                  ;*      'NED' SETS WHEN ATEMPTING TO ACCESS THE DRIVE THROUGH
2518                                  ;*      PORT A.  ATTEMPT TO SET PORT REQUEST BY WRITING 0'S
2519                                  ;*      INTO RMDS1 THROUGH PORT A.
2520                                  ;*
2521                                  ;*  G.  ISSUE A RELEASE COMMAND THROUGH PORT B.  VERIFY THAT THE
2522                                  ;*      DRIVE REMAINS LOCKED ON PORT B.
2523                                  ;*
2524                                  ;*  H.  CYCLE THE DRIVE DOWN.  CHANGE THE 'CONTROLLER SELECT' SWITCH TO
2525                                  ;*      A/B; CYCLE THE DRIVE UP.
2526                                  ;*
2527                                  ;*  I.  VERIFY THAT BOTH PORTS CAN ACCESS THE DRIVE, THAT BOTH ATTENTION
2528                                  ;*      BITS ARE SET, AND THAT BOTH 'VV' BITS ARE RESET.
2529                                  ;*
2530                                  ;*****************************************************************
2531  011702                          TST10:
2532  011702  005737  001266                  TST     KYBCTL          ;PERFORMING ONLY SINGLE TESTS ?
2533  011706  001406                          BEQ     2$              ;BR IF NOT
2534  011710  100002                          BPL     1$              ;BR IF JUST ENTERED TEST
2535  011712  000137  002424                  JMP     EXEC            ;RETURN & GET NEXT TEST NUMBER
2536  011716  012737  177777  001266  1$:     MOV     #-1,KYBCTL      ;SET SINGLE TEST INDICATOR
2537  011724  112737  000010  001102  2$:     MOVB    #10,$TSTNM      ;TEST NUMBER
2538  011732  012737  011754  001106          MOV     #TEST10,$LPADR  ;LOAD LOOP ON TEST ADDRESS
2539  011740  012737  011754  001110          MOV     #TEST10,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
2540  011746  012737  000001  001176          MOV     #1,$TIMES       ;;DO 1 ITERATION
2541  011754  012706  001100          TEST10: MOV     #STACK,SP       ;SETUP THE STACK POINTER
2542  011760  113760  001226  000010          MOVB    PORTB,RMCS2(R0)  ;SELECT PORT B
2543  011766  013737  001226  001234          MOV     PORTB,PTNBR  ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
2544  011774  104401  017617                  TYPE    ,CYCLED         ;'CYCLE DOWN THE DRIVE'
2545
```

```
                                ;;*****************************************************************
                                ;WAIT FOR 'MOL' TO RESET
                                
012000  032760  010000  000012  1$:     BIT     #MOL,RMDS1(R0)  ;TEST 'MOL'
012006  001374                          BNE     1$              ;BR IF IT IS STILL SET
012010  104401  017465                  TYPE    ,SWTCHB         ;SWITCH TO 'B'
012014  104401  017641                  TYPE    ,CYCLEU         ;'CYCLE UP THE DRIVE'
                                
                                ;;*****************************************************************
                                ;WAIT FOR DRIVE TO CYCLE UP AFTER SWITCH CHANGED
                                
012020  032760  010000  000012  2$:     BIT     #MOL,RMDS1(R0)  ;TEST 'MOL' AGAIN
012026  001774                          BEQ     2$              ;BR IF NOT SET
                                
                                ;;*****************************************************************
                                ;DRIVE IS CYCLED UP, CHECK STATUS THROUGH PORT B
                                
012030  005037  001244                  CLR     CKERR           ;CLEAR THE 'CHECK ERROR' INDICATOR
012034  016037  000012  001126          MOV     RMDS1(R0),$BDDAT  ;GET CONTENTS OF RMDS1
012042  012737  000012  001122          MOV     #RMDS1,$BDADR   ;FORM REGISTER ADDRESS OF ERROR MESSAGE
012050  060037  001122                  ADD     R0,$BDADR       ;ADD RH11 BASE ADDRESS
012054  012737  110600  001124          MOV     #ATA!MOL!DPR!DRY,$GDDAT  ;WHAT REGISTER SHOULD BE
012062  013737  001126  001164          MOV     $BDDAT,$TMP0    ;MOVE REGISTER CONTENTS TO '$TMP0'
012070  042737  066077  001164          BIC     #^C111700,$TMP0  ;SAVE SPECIFIED BITS
012076  023737  001124  001164          CMP     $GDDAT,$TMP0    ;COMPARE THE BITS
012104  001414                          BEQ     64$             ;BR IF OK
012106  013737  001126  001174          MOV     $BDDAT,$TMP4    ;COPY 'BAD DATA'
012114  042737  111700  001174          BIC     #111700,$TMP4   ;CLEAR THE MASKED BITS
012122  053737  001174  001124          BIS     $TMP4,$GDDAT    ;'OR' WITH GOOD DATA FOR TYPEOUT
012130  104021                          ERROR   21              ;TYPE MESSAGE 21
012132  005137  001244                  COM     CKERR           ;SET THE REGISTER COMPARE ERROR INDICATOR
012136  000240                  64$:     NOP
                                
                                ;;*****************************************************************
                                ;SET VOLUME VALID FOR PORT B
                                
012140  012760  000011  000000          MOV     #11,RMCS1(R0)   ;ISSUE A DRIVE CLEAR
012146  012760  000021  000000          MOV     #21,RMCS1(R0)   ;ISSUE A READIN PRESET
012154  012760  010000  000032          MOV     #FMT16,RMOF(R0) ;SET FMT16
                                
                                ;;*****************************************************************
                                ;CHECK THE DRIVE STATUS THROUGH PORT A; VERIFY THAT 'NED'
                                ;       SETS WHEN THE DRIVE IS ACCESSED THROUGH PORT A.
                                
012162  113760  001224  000010          MOVB    PORTA,RMCS2(R0)  ;SELECT PORT A
012170  013737  001224  001234          MOV     PORTA,PTNBR     ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
012176  005037  001244                  CLR     CKERR           ;CLEAR THE 'CHECK ERROR' INDICATOR
012202  016037  000012  001126          MOV     RMDS1(R0),$BDDAT  ;GET CONTENTS OF RMDS1
012210  012737  000012  001122          MOV     #RMDS1,$BDADR   ;FORM REGISTER ADDRESS OF ERROR MESSAGE
012216  060037  001122                  ADD     R0,$BDADR       ;ADD RH11 BASE ADDRESS
012222  005037  001124                  CLR     $GDDAT          ;WHAT REGISTER SHOULD BE
012226  013737  001126  001164          MOV     $BDDAT,$TMP0    ;MOVE REGISTER CONTENTS TO '$TMP0'
012234  042737  000077  001164          BIC     #^C177700,$TMP0  ;SAVE SPECIFIED BITS
012242  023737  001124  001164          CMP     $GDDAT,$TMP0    ;COMPARE THE BITS
012250  001414                          BEQ     66$             ;BR IF OK
012252  013737  001126  001174          MOV     $BDDAT,$TMP4    ;COPY 'BAD DATA'
```

```
2602  012260  042737  177700  001174          BIC     #177700,$TMP4   ;CLEAR THE MASKED BITS
2603  012266  053737  001174  001124          BIS     $TMP4,$GDDAT    ;'OR' WITH GOOD DATA FOR TYPEOUT
2604  012274  104022                          ERROR   22              ;TYPE MESSAGE 22
2605  012276  005137  001244                  COM     CKERR           ;SET THE REGISTER COMPARE ERROR INDICATOR
2606  012302  000240                  66$:    NOP
2607  012304  005037  001244                  CLR     CKERR           ;CLEAR THE 'CHECK ERROR' INDICATOR
2608  012310  016037  000010  001126          MOV     RMCS2(R0),$BDDAT  ;GET CONTENTS OF RMCS2
2609  012316  012737  000010  001122          MOV     #RMCS2,$BDADR   ;FORM REGISTER ADDRESS OF ERROR MESSAGE
2610  012324  060037  001122                  ADD     R0,$BDADR       ;ADD RH11 BASE ADDRESS
2611  012330  012737  010000  001124          MOV     #NED,$GDDAT     ;WHAT REGISTER SHOULD BE
2612  012336  013737  001126  001164          MOV     $BDDAT,$TMP0    ;MOVE REGISTER CONTENTS TO '$TMP0'
2613  012344  042737  167777  001164          BIC     #^CNED,$TMP0    ;SAVE SPECIFIED BITS
2614  012352  023737  001124  001164          CMP     $GDDAT,$TMP0    ;COMPARE THE BITS
2615  012360  001414                          BEQ     68$             ;BR IF OK
2616  012362  013737  001126  001174          MOV     $BDDAT,$TMP4    ;COPY 'BAD DATA'
2617  012370  042737  010000  001174          BIC     #NED,$TMP4      ;CLEAR THE MASKED BITS
2618  012376  053737  001174  001124          BIS     $TMP4,$GDDAT    ;'OR' WITH GOOD DATA FOR TYPEOUT
2619  012404  104023                          ERROR   23              ;TYPE MESSAGE 23
2620  012406  005137  001244                  COM     CKERR           ;SET THE REGISTER COMPARE ERROR INDICATOR
2621  012412  000240                  68$:    NOP
2622  012414  005060  000012                  CLR     RMDS1(R0)       ;TRY TO SET REQUEST BY WRITING THROUGH
2623                                                                  ;THE LOCKED OUT PORT (PORT 'A')
2624
2625                                  ;;*****************************************************************
2626                                  ;VERIFY THAT DRIVE STAYS LOCKED ON PORT B
2627
2628  012420  113760  001226  000010          MOVB    PORTB,RMCS2(R0)  ;SELECT PORT B
2629  012426  013737  001226  001234          MOV     PORTB,PTNBR     ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
2630  012434  012760  000013  000012          MOV     #13,RMDS1(R0)   ;ISSUE A RELEASE THROUGH PORT B
2631  012442  013737  001226  001236          MOV     PORTB,SEIZPT    ;ADDRESS OF 'LOCKED ON' PORT
2632  012450  113760  001224  000010          MOVB    PORTA,RMCS2(R0)  ;SELECT PORT A
2633  012456  013737  001224  001234          MOV     PORTA,PTNBR     ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
2634  012464  005037  001244                  CLR     CKERR           ;CLEAR THE 'CHECK ERROR' INDICATOR
2635  012470  016037  000012  001126          MOV     RMDS1(R0),$BDDAT  ;GET CONTENTS OF RMDS1
2636  012476  012737  000012  001122          MOV     #RMDS1,$BDADR   ;FORM REGISTER ADDRESS OF ERROR MESSAGE
2637  012504  060037  001122                  ADD     R0,$BDADR       ;ADD RH11 BASE ADDRESS
2638  012510  005037  001124                  CLR     $GDDAT          ;WHAT REGISTER SHOULD BE
2639  012514  013737  001126  001164          MOV     $BDDAT,$TMP0    ;MOVE REGISTER CONTENTS TO '$TMP0'
2640  012522  042737  000077  001164          BIC     #^C177700,$TMP0  ;SAVE SPECIFIED BITS
2641  012530  023737  001124  001164          CMP     $GDDAT,$TMP0    ;COMPARE THE BITS
2642  012536  001414                          BEQ     70$             ;BR IF OK
2643  012540  013737  001126  001174          MOV     $BDDAT,$TMP4    ;COPY 'BAD DATA'
2644  012546  042737  177700  001174          BIC     #177700,$TMP4   ;CLEAR THE MASKED BITS
2645  012554  053737  001174  001124          BIS     $TMP4,$GDDAT    ;'OR' WITH GOOD DATA FOR TYPEOUT
2646  012562  104024                          ERROR   24              ;TYPE MESSAGE 24
2647  012564  005137  001244                  COM     CKERR           ;SET THE REGISTER COMPARE ERROR INDICATOR
2648  012570  000240                  70$:    NOP
2649
2650                                  ;IF ERROR OCCURRED, CHECK FOR LOOP ON TEST
2651
2652  012572  105737  001103                  TSTB    $ERFLG          ;DID AN ERROR OCCUR
2653  012576  001412                          BEQ     3$              ;BR IF NOT
2654  012600  032777  001000  166332          BIT     #SW09,@SWR      ;SEE IF LOOP ON ERROR (SWR9=1)
2655  012606  001406                          BEQ     3$              ;BR IF NOT
2656  012610  105037  001103                  CLRB    $ERFLG          ;CLEAR THE ERROR FLAG
2657  012614  005037  001176                  CLR     $TIMES          ;CLEAR THE MAX ITERATION COUNT
```

```
2658  012620  000177  166264                  JMP     @SLPERR         ;GO TO THE LOOP ADDRESS
2659  012624  032777  040000  166306  3$:     BIT     #SW14,@SWR      ;LOOP ON TEST ?
2660  012632  001054                          BNE     6$              ;BR IF LOOPING
2661  012634  104401  017617                  TYPE    ,CYCLED         ;TYPE 'CYCLE DOWN'
2662  012640  104401  017313                  TYPE    ,SWTCHN         ;'SWITCH TO A/B'
2663  012644  104401  017641                  TYPE    ,CYCLEU         ;'CYCLE THE DRIVE UP'
2664  012650  113760  001226  000010          MOVB    PORTB,RMCS2(R0)  ;SELECT PORT B
2665  012656  013737  001226  001234          MOV     PORTB,PTNBR  ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
2666  012664  032760  010000  000012  4$:     BIT     #MOL,RMDS1(R0)  ;CHECK 'MOL'
2667  012672  001374                          BNE     4$              ;BR IF SET (DRIVE NOT CYCLED DOWN)
2668  012674  032760  010000  000012  5$:     BIT     #MOL,RMDS1(R0)  ;CHECK 'MOL' AGAIN
2669  012702  001774                          BEQ     5$              ;BR IF NOT SET (DRIVE NOT CYCLED UP)
2670
2671                                  ;;*****************************************************************
2672                                  ;SET VOLUME VALID FOR BOTH PORTS
2673
2674  012704  012760  000011  000000          MOV     #11,RMCS1(R0)   ;ISSUE A DRIVE CLEAR THROUGH PORT B
2675  012712  012760  000021  000000          MOV     #21,RMCS1(R0)   ;ISSUE A READIN PRESET THROUGH PORT B
2676  012720  012760  000013  000000          MOV     #13,RMCS1(R0)   ;RELEASE PORT B
2677  012726  113760  001224  000010          MOVB    PORTA,RMCS2(R0)  ;SELECT PORT A
2678  012734  013737  001224  001234          MOV     PORTA,PTNBR  ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
2679  012742  012760  000021  000000          MOV     #21,RMCS1(R0)   ;ISSUE A READIN PRESET THROUGH PORT A
2680  012750  012760  010000  000032          MOV     #FMT16,RMOF(R0) ;SET FMT16
2681  012756  012760  000013  000000          MOV     #13,RMCS1(R0)   ;RELEASE PORT A
2682  012764  012737  011610  001254  6$:     MOV     #5000.,WATCH    ;SPINDLE MOTOR 'COOL DOWN' DELAY
2683  012772  005737  001254          7$:     TST     WATCH           ;FINISHED ?
2684  012776  001375                          BNE     7$              ;BR IF NOT
2685  013000  000004                          SCOPE                   ;LOOP ?
2686  013002  000137  013006                  JMP     $EOP            ;GO TO THE END OF PASS ROUTINE
2687
2688                                  .SBTTL  END OF PASS ROUTINE
2689
2690                                  ;;*****************************************************************
2691                                  ;*INCREMENT THE PASS NUMBER ($PASS)
2692                                  ;*INDICATE END-OF-PROGRAM AFTER 1 PASSES THRU THE PROGRAM
2693                                  ;*TYPE "END PASS #XXXXX TOTAL NUMBER OF ERRORS SINCE LAST REPORT YYYYY"
2694                                  ;*WHERE XXXXX AND YYYYY ARE DECIMAL NUMBERS
2695                                  ;*IF THERES A MONITOR GO TO IT
2696                                  ;*IF THERE ISN'T JUMP TO TST1AA
2697
2698  013006                          $EOP:
2699  013006  005737  001266                  TST     KYBCTL          ;ENTERED TEST VIA KEYBOARD COMMAND ?
2700  013012  001402                          BEQ     .+6             ;BR IF NOT
2701  013014  000137  002424                  JMP     EXEC            ;RETURN TO KEYBOARD CONTROL
2702  013020  005037  001102                  CLR     $TSTNM          ;;ZERO THE TEST NUMBER
2703  013024  005037  001176                  CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
2704  013030  005237  001100                  INC     $PASS           ;;INCREMENT THE PASS NUMBER
2705  013034  042737  100000  001100          BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
2706  013042  005327                          DEC     (PC)+           ;;LOOP?
2707  013044  000001          $EOPCT: .WORD   1
2708  013046  003063                          BGT     $DOAGN          ;;YES
2709  013050  012737                          MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
2710  013052  000001          $ENDCT: .WORD   1
2711  013054  013044                          $EOPCT
2712  013056  104401  013064                  TYPE    ,65$            ;;TYPE ASCIZ STRING
2713  013062  000407                          BR      64$             ;;GET OVER THE ASCIZ
```

```
                                        ;;65$:  .ASCIZ  <12><15>/END PASS #/
                                        64$:
013102  013746  001100                          MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
                                                                        ;;TYPE PASS NUMBER
013106  104405                                  TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
013110  104401  013116                          TYPE    ,67$            ;;TYPE ASCIZ STRING
013114  000421                                  BR      66$             ;;GET OVER THE ASCIZ
                                        ;;67$:  .ASCIZ  /  TOTAL ERRORS SINCE LAST REPORT /
                                        66$:
013160  013746  001112                          MOV     $ERTTL,-(SP)    ;;SAVE $ERTTL FOR TYPEOUT
                                                                        ;;TOTAL NUMBER OF ERRORS
013164  104405                                  TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
013166  104401  001207                          TYPE    ,$CRLF          ;;TYPE CARRIAGE RETURN, LINE FEED
013172  005037  001112                          CLR     $ERTTL          ;;CLEAR ERROR TOTAL
013176  013700  000042                  $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
013202  001405                                  BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
013204  000005                                  RESET                   ;;CLEAR THE WORLD
013206  004710                          $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
013210  000240                                  NOP                     ;;SAVE ROOM
013212  000240                                  NOP                     ;;FOR
013214  000240                                  NOP                     ;;ACT11
013216                                  $DOAGN:
013216  000137                                  JMP     @(PC)+          ;;RETURN
013220  002706                          $RTNAD: .WORD   TST1AA
013222     377     377     000          $ENULL: .BYTE   -1,-1,0         ;;NULL CHARACTER STRING
        013226                                  .EVEN

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

                                        .SBTTL  *** SUBROUTINES ***

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

                                        ;ROUTINE TO CHECK FOR KW11-L OR KW11-P CLOCKS

013226  012737  013276  000004  CKCLK:  MOV     #CKCLK1,@#ERRVEC  ;SET UP VECTOR FOR CLOCK CHECK
013234  005037  000006                  CLR     @#ERRVEC+2        ;NEW PSW
013240  005777  165746                  TST     @$LKCSR         ;CHECK FOR KW11-P
013244  013701  001216                  MOV     $LPVEC,R1       ;KW11-P VECTOR ADDRESS
013250  012721  013360                  MOV     #CLOCK,(R1)+    ;SET UP KW11-P VECTOR
013254  012711  000300                  MOV     #300,(R1)       ;PSW - PRI 6
013260  012777  177777  165726          MOV     #-1,@$LKCSB     ;LOAD COUNTER BUFFER WITH 1'S
013266  012777  000135  165716          MOV     #135,@$LKCSR    ;SET CLOCK - CNT UP, 16MS, CONT INT
013274  000425                          BR      CKCLK3
013276  062706  000004          CKCLK1: ADD     #4,SP           ;RESTORE THE STACK POINTER
013302  012737  013340  000004          MOV     #CKCLK2,@#ERRVEC  ;CHANGE ERROR VECTOR TO CHECK FOR KW11-L
013310  005777  165704                  TST     @$LKS           ;LOOK FOR KW11-L
013314  013701  001222                  MOV     $LLVEC,R1       ;KW11-L VECTOR ADDRESS
013320  012721  013360                  MOV     #CLOCK,(R1)+    ;SET UP KW11-L VECTOR
013324  012711  000300                  MOV     #300,(R1)       ;PSW - PRI 6
013330  012777  000100  165662          MOV     #100,@$LKS      ;SET KW11-L INTERRUPT
013336  000404                          BR      CKCLK3
013340  062706  000004          CKCLK2: ADD     #4,SP           ;RESTORE THE STACK POINTER
013344  062716  000002                  ADD     #2,(SP)         ;INCREMENT RETURN, NO CLOCK
013350  012737  000006  000004  CKCLK3: MOV     #6,@#ERRVEC     ;RESTORE THE ERROR VECTOR
```

```
                        *** SUBROUTINES ***

2770  013356  000207                RTS     PC
2771
2772                    ;ROUTINE TO COUNT CLOCK TICKS
2773
2774  013360  062737  000021  001252  CLOCK:  ADD     #17.,TIME       ;ADD 17 MS TO ELAPSED TIME COUNTER
2775  013366  005737  001254          TST     WATCH           ;IS WATCH ALREADY ZERO ?
2776  013372  001406                  BEQ     1$              ;BR IF IT IS
2777  013374  162737  000021  001254          SUB     #17.,WATCH      ;SUBTRACT 17 MS FROM WATCH DOG COUNTER
2778  013402  100002                  BPL     1$              ;BR IF NOT MINUS
2779  013404  005037  001254          CLR     WATCH           ;CLEAR WATCH DOG COUNTER
2780  013410  000002          1$:     RTI                     ;RETURN
2781
2782                    ;ROUTINE TO CALCULATE + 25% TIME TOLERANCE VALUES
2783
2784  013412  005746          TOLER:  TST     -(SP)           ;MAKE ROOM ON THE STACK
2785  013414  016616  000002          MOV     2(SP),(SP)      ;SAVE STACK
2786  013420  013546                  MOV     @(R5)+,-(SP)    ;GET TIME VALUE
2787  013422  011666  000004          MOV     (SP),4(SP)      ;MOVE TIME VALUE
2788  013426  006216                  ASR     (SP)            ;DIVIDE BY 2
2789  013430  006216                  ASR     (SP)            ;DIVIDE BY 2 AGAIN (FOR A TOTAL OF 4)
2790  013432  062666  000002          ADD     (SP)+,2(SP)     ;CALCULATE UPPER LIMIT FOR TIMEOUT
2791  013436  000205                  RTS     R5              ;RETURN WITH TOLERANCES ON THE STACK
2792
2793                    ;;*****************************************************************
2794
2795                    .SBTTL  'SYSMAC' UTILITY ROUTINES
2796
2797                    .SBTTL  SCOPE HANDLER ROUTINE
2798
2799                    ;;*****************************************************************
2800                    ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
2801                    ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
2802                    ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
2803                    ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
2804                    ;*SW14=1         LOOP ON TEST
2805                    ;*SW11=1         INHIBIT ITERATIONS
2806                    ;*CALL
2807                    ;*      SCOPE           ;;SCOPE=IOT
2808
2809  013440          $SCOPE:
2810  013440  104407          CKSWR                           ;;TEST  FOR CHANGE IN SOFT-SWR
2811  013442  032777  040000  165470  1$:     BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
2812  013450  001055                  BNE     $OVER           ;;YES IF SW14=1
2813                    ;#####START OF CODE FOR THE XOR TESTER#####
2814  013452  000416          $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
2815                                                          ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
2816  013454  013746  000004          MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
2817  013460  012737  013500  000004  MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
2818  013466  005737  177060          TST     @#177060        ;;TIME OUT ON XOR?
2819  013472  012637  000004          MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
2820  013476  000436                  BR      $SVLAD          ;;GO TO THE NEXT TEST
2821  013500  022626          5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
2822  013502  012637  000004          MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
2823  013506  000436                  BR      $OVER           ;;LOOP ON THE PRESENT TEST
2824  013510          6$:;#####END OF CODE FOR THE XOR TESTER#####
2825  013510  105737  001103  2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
```

```
2826  013514  001404                        BEQ     3$              ;;BR IF NO
2827  013516  105037  001103          4$:   CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
2828  013522  005037  001176                CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
2829  013526  032777  004000  165404  3$:   BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
2830  013534  001011                        BNE     1$              ;;BR IF YES
2831  013536  005737  001100                TST     $PASS           ;;IF FIRST PASS OF PROGRAM
2832  013542  001406                        BEQ     1$              ;;      INHIBIT ITERATIONS
2833  013544  005237  001104                INC     $ICNT           ;;INCREMENT ITERATION COUNT
2834  013550  023737  001176  001104        CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
2835  013556  002012                        BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
2836  013560  012737  000001  001104  1$:   MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
2837  013566  013737  013620  001176        MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
2838  013574  105237  001102          $SVLAD: INCB  $TSTNM          ;;COUNT TEST NUMBERS
2839  013600  011637  001106                MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
2840  013604  013777  001102  165330  $OVER: MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
2841  013612  013716  001106                MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
2842  013616  000002                        RTI                     ;;FIXES PS
2843  013620  000004                $MXCNT: 4                       ;;MAX. NUMBER OF ITERATIONS
2844                                 .SBTTL  ERROR HANDLER ROUTINE
2845
2846                                 ;;**************************************************************
2847                                 ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
2848                                 ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
2849                                 ;*AND GO TO $ERRTYP ON ERROR
2850                                 ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
2851                                 ;*SW15=1         HALT ON ERROR
2852                                 ;*SW13=1         INHIBIT ERROR TYPEOUTS
2853                                 ;*SW10=1         BELL ON ERROR
2854                                 ;*CALL
2855                                 ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER
2856
2857  013622                         $ERROR:
2858  013622  104407                        CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
2859  013624  113737  001102  001242        MOVB    $TSTNM,TSTNUM
2860  013632  105237  001103          7$:   INCB    $ERFLG          ;;SET THE ERROR FLAG
2861  013636  001775                        BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
2862  013640  013777  001102  165274        MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
2863  013646  032777  002000  165264        BIT     #BIT10,@SWR     ;;BELL ON ERROR?
2864  013654  001402                        BEQ     1$              ;;NO - SKIP
2865  013656  104401  001202                TYPE    ,$BELL          ;;RING BELL
2866  013662  005237  001112          1$:   INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
2867  013666  011637  001116                MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
2868  013672  162737  000002  001116        SUB     #2,$ERRPC
2869  013700  117737  165212  001114        MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
2870  013706  032777  020000  165224        BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
2871  013714  001004                        BNE     20$             ;;SKIP TYPEOUTS
2872  013716  004737  013754                JSR     PC,$ERRTYP      ;;GO TO USER ERROR ROUTINE
2873  013722  104401  001207                TYPE    ,$CRLF
2874  013726                         20$:
2875  013726  005777  165206          2$:   TST     @SWR            ;;HALT ON ERROR
2876  013732  100002                        BPL     3$              ;;SKIP IF CONTINUE
2877  013734  000000                        HALT                    ;;HALT ON ERROR!
2878  013736  104407                        CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
2879  013740                         3$:
2880  013740  022737  013206  000042        CMP     #$ENDAD,@#42    ;;ACT-11 AUTO-ACCEPT?
2881  013746  001001                        BNE     6$              ;;BRANCH IF NO
```

```
                                        HALT                    ;;YES
                                6$:
                                        RTI                     ;;RETURN
                                .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE

                                ;;**********************************************************************
                                ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
                                ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
                                ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.

                                $ERRTYP:
                                        TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
                                        MOV     R0,-(SP)        ;;SAVE R0
                                        CLR     R0              ;;PICKUP THE ITEM INDEX
                                        BISB    @#$ITEMB,R0
                                        BNE     1$              ;;IF ITEM NUMBER IS ZERO, JUST
                                                                ;;TYPE THE PC OF THE ERROR
                                        MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
                                                                ;;ERROR ADDRESS
                                        TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
                                        BR      10$             ;;GET OUT
                                1$:     DEC     R0              ;;ADJUST THE INDEX SO THAT IT WILL
                                        ASL     R0              ;;      WORK FOR THE ERROR TABLE
                                        ASL     R0
                                        ASL     R0
                                        ADD     #$ERRTB,R0      ;;FORM TABLE POINTER
                                        MOV     (R0)+,2$        ;;PICKUP "ERROR MESSAGE" POINTER
                                        BEQ     3$              ;;SKIP TYPEOUT IF NO POINTER
                                        TYPE                    ;;TYPE THE "ERROR MESSAGE"
                                2$:     .WORD   0               ;;"ERROR MESSAGE" POINTER GOES HERE
                                        TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
                                3$:     MOV     (R0)+,4$        ;;PICKUP "DATA HEADER" POINTER
                                        BEQ     5$              ;;SKIP TYPEOUT IF 0
                                        TYPE                    ;;TYPE THE "DATA HEADER"
                                4$:     .WORD   0               ;;"DATA HEADER" POINTER GOES HERE
                                        TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
                                5$:     MOV     R1,-(SP)        ;;SAVE R1
                                        MOV     (R0)+,R1        ;;PICKUP "DATA TABLE" POINTER
                                        BEQ     9$              ;;BR IF NO DATA TO BE TYPED
                                        MOV     (R0)+,R0        ;;PICKUP "DATA FORMAT" POINTER
                                6$:     TSTB    (R0)+           ;;"OCTAL" OR "DECIMAL"
                                        BNE     7$              ;;BR IF DECIMAL
                                        MOV     @(R1)+,-(SP)    ;;SAVE @(R1)+ FOR TYPEOUT
                                        TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
                                        BR      8$
                                7$:
                                        MOV     @(R1)+,-(SP)    ;;SAVE @(R1)+ FOR TYPEOUT
                                        TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
                                8$:     TST     (R1)            ;;IS THERE ANOTHER NUMBER?
                                        BEQ     9$              ;;BR IF NO
                                        TYPE    ,11$            ;;TYPE TWO(2) SPACES
                                        BR      6$              ;;LOOP

                                9$:     MOV     (SP)+,R1        ;;RESTORE R1
                                10$:    MOV     (SP)+,R0        ;;RESTORE R0
                                        TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
```

```
2938  014122  000207                    RTS     PC              ;;RETURN
2939  014124  020040  000       11$:    .ASCIZ  / /             ;;TWO(2) SPACES
2940          014130                    .EVEN
2941                            .SBTTL  TYPE ROUTINE
2942
2943                            ;;*****************************************************************
2944                            ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
2945                            ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
2946                            ;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
2947                            ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
2948                            ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
2949                            ;*
2950                            ;*CALL:
2951                            ;*1) USING A TRAP INSTRUCTION
2952                            ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
2953                            ;*OR
2954                            ;*      TYPE
2955                            ;*      MESADR
2956                            ;*
2957
2958  014130  105737  001157    $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
2959  014134  100002                    BPL     1$              ;;BR IF YES
2960  014136  000000                    HALT                    ;;HALT HERE IF NO TERMINAL
2961  014140  000407                    BR      3$              ;;LEAVE
2962  014142  010046            1$:     MOV     R0,-(SP)        ;;SAVE R0
2963  014144  017600  000002            MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
2964  014150  112046            2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
2965  014152  001005                    BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
2966  014154  005726                    TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
2967  014156  012600            60$:    MOV     (SP)+,R0        ;;RESTORE R0
2968  014160  062716  000002    3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
2969  014164  000002                    RTI                     ;;RETURN
2970  014166  122716  000011    4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
2971  014172  001430                    BEQ     8$
2972  014174  122716  000200            CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
2973  014200  001006                    BNE     5$
2974  014202  005726                    TST     (SP)+           ;;POP  <CR><LF> EQUIV
2975  014204  104401                    TYPE                    ;;TYPE A CR AND LF
2976  014206  001207                    $CRLF
2977  014210  105037  014344            CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
2978  014214  000755                    BR      2$              ;;GET NEXT CHARACTER
2979  014216  004737  014300    5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
2980  014222  123726  001156    6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
2981  014226  001350                    BNE     2$              ;;IF NO GO GET NEXT CHAR.
2982  014230  013746  001154            MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
2983                                                            ;;AND THE NULL CHAR.
2984  014234  105366  000001    7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
2985  014240  002770                    BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
2986  014242  004737  014300            JSR     PC,$TYPEC       ;;GO TYPE A NULL
2987  014246  105337  014344            DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
2988  014252  000770                    BR      7$              ;;LOOP
2989
2990                            ;HORIZONTAL TAB PROCESSOR
2991
2992  014254  112716  000040    8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
2993  014260  004737  014300    9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
```

```
 2994  014264  132737  000007  014344          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
 2995  014272  001372                          BNE     9$              ;;TAB STOP
 2996  014274  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
 2997  014276  000724                          BR      2$              ;;GET NEXT CHARACTER
 2998  014300  105777  164644          $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
 2999  014304  100375                          BPL     $TYPEC
 3000  014306  116677  000002  164636        MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
 3001  014314  122766  000015  000002        CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
 3002  014322  001003                          BNE     1$              ;;BRANCH IF NO
 3003  014324  105037  014344                CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
 3004  014330  000406                          BR      $TYPEX          ;;EXIT
 3005  014332  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
 3006  014340  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
 3007  014342  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
 3008  014344  000000          $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
 3009  014346  000207          $TYPEX: RTS     PC
 3010
 3011                          .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
 3012
 3013                  ;;**************************************************************
 3014                  ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
 3015                  ;*OCTAL (ASCII) NUMBER AND TYPE IT.
 3016                  ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
 3017                  ;*CALL:
 3018                  ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
 3019                  ;*      TYPOS                   ;;CALL FOR TYPEOUT
 3020                  ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
 3021                  ;*      .BYTE   M               ;;M=1 OR 0
 3022                  ;*                                      ;;1=TYPE LEADING ZEROS
 3023                  ;*                                      ;;0=SUPPRESS LEADING ZEROS
 3024                  ;*
 3025                  ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
 3026                  ;*$TYPOS OR $TYPOC
 3027                  ;*CALL:
 3028                  ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
 3029                  ;*      TYPON                   ;;CALL FOR TYPEOUT
 3030                  ;*
 3031                  ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
 3032                  ;*CALL:
 3033                  ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
 3034                  ;*      TYPOC                   ;;CALL FOR TYPEOUT
 3035
 3036  014350  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
 3037  014354  116637  000001  014573          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
 3038  014362  112637  014575                MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
 3039  014366  062716  000002                ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
 3040  014372  000406                        BR      $TYPON
 3041  014374  112737  000001  014573  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
 3042  014402  112737  000006  014575          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
 3043  014410  112737  000005  014572  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
 3044  014416  010346                        MOV     R3,-(SP)        ;;SAVE R3
 3045  014420  010446                        MOV     R4,-(SP)        ;;SAVE R4
 3046  014422  010546                        MOV     R5,-(SP)        ;;SAVE R5
 3047  014424  113704  014575                MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
 3048  014430  005404                        NEG     R4
 3049  014432  062704  000006                ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
```

```
3050  014436  110437  014574          MOVB    R4,$OMODE       ;;SAVE IT FOR USE
3051  014442  113704  014573          MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
3052  014446  016605  000012          MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
3053  014452  005003                  CLR     R3              ;;CLEAR THE OUTPUT WORD
3054  014454  006105          1$:     ROL     R5              ;;ROTATE MSB INTO "C"
3055  014456  000404                  BR      3$              ;;GO DO MSB
3056  014460  006105          2$:     ROL     R5              ;;FORM THIS DIGIT
3057  014462  006105                  ROL     R5
3058  014464  006105                  ROL     R5
3059  014466  010503                  MOV     R5,R3
3060  014470  006103          3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
3061  014472  105337  014574          DECB    $OMODE          ;;TYPE THIS DIGIT?
3062  014476  100016                  BPL     7$              ;;BR IF NO
3063  014500  042703  177770          BIC     #177770,R3      ;;GET RID OF JUNK
3064  014504  001002                  BNE     4$              ;;TEST FOR 0
3065  014506  005704                  TST     R4              ;;SUPPRESS THIS 0?
3066  014510  001403                  BEQ     5$              ;;BR IF YES
3067  014512  005204          4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
3068  014514  052703  000060          BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
3069  014520  052703  000040          BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
3070  014524  110337  014570          MOVB    R3,8$           ;;SAVE FOR TYPING
3071  014530  104401  014570          TYPE    ,8$             ;;GO TYPE THIS DIGIT
3072  014534  105337  014572  7$:     DECB    $OCNT           ;;COUNT BY 1
3073  014540  003347                  BGT     2$              ;;BR IF MORE TO DO
3074  014542  002402                  BLT     6$              ;;BR IF DONE
3075  014544  005204                  INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
3076  014546  000744                  BR      2$              ;;GO DO THE LAST DIGIT
3077  014550  012605          6$:     MOV     (SP)+,R5        ;;RESTORE R5
3078  014552  012604                  MOV     (SP)+,R4        ;;RESTORE R4
3079  014554  012603                  MOV     (SP)+,R3        ;;RESTORE R3
3080  014556  016666  000002  000004  MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
3081  014564  012616                  MOV     (SP)+,(SP)
3082  014566  000002                  RTI                     ;;RETURN
3083  014570  000             8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
3084  014571  000                     .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
3085  014572  000             $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
3086  014573  000             $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
3087  014574  000000          $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
3088                          .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
3089
3090                          ;;**************************************************************
3091                          ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
3092                          ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
3093                          ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
3094                          ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
3095                          ;*REPLACED WITH SPACES.
3096                          ;*CALL:
3097                          ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
3098                          ;*      TYPDS                   ;;GO TO THE ROUTINE
3099
3100  014576                  $TYPDS:
3101  014576  010046                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
3102  014600  010146                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
3103  014602  010246                  MOV     R2,-(SP)        ;;PUSH R2 ON STACK
3104  014604  010346                  MOV     R3,-(SP)        ;;PUSH R3 ON STACK
3105  014606  010546                  MOV     R5,-(SP)        ;;PUSH R5 ON STACK
```

```
3106  014610  012746  020200                    MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
3107  014614  016605  000020                    MOV     20(SP),R5       ;;GET THE INPUT NUMBER
3108  014620  100004                            BPL     1$              ;;BR IF INPUT IS POS.
3109  014622  005405                            NEG     R5              ;;MAKE THE BINARY NUMBER POS.
3110  014624  112766  000055  000001            MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
3111  014632  005000                    1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
3112  014634  012703  015012                    MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
3113  014640  112723  000040                    MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
3114  014644  005002                    2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
3115  014646  016001  015002                    MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
3116  014652  160105                    3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
3117  014654  002402                            BLT     4$              ;;BR IF DONE
3118  014656  005202                            INC     R2              ;;INCREASE THE BCD DIGIT BY 1
3119  014660  000774                            BR      3$
3120  014662  060105                    4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
3121  014664  005702                            TST     R2              ;;CHECK IF BCD DIGIT=0
3122  014666  001002                            BNE     5$              ;;FALL THROUGH IF 0
3123  014670  105716                            TSTB    (SP)            ;;STILL DOING LEADING 0'S?
3124  014672  100407                            BMI     7$              ;;BR IF YES
3125  014674  106316                    5$:     ASLB    (SP)            ;;MSD?
3126  014676  103003                            BCC     6$              ;;BR IF NO
3127  014700  116663  000001  177777            MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
3128  014706  052702  000060            6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
3129  014712  052702  000040            7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
3130  014716  110223                            MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
3131  014720  005720                            TST     (R0)+           ;;JUST INCREMENTING
3132  014722  020027  000010                    CMP     R0,#10          ;;CHECK THE TABLE INDEX
3133  014726  002746                            BLT     2$              ;;GO DO THE NEXT DIGIT
3134  014730  003002                            BGT     8$              ;;GO TO EXIT
3135  014732  010502                            MOV     R5,R2           ;;GET THE LSD
3136  014734  000764                            BR      6$              ;;GO CHANGE TO ASCII
3137  014736  105726                    8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
3138  014740  100003                            BPL     9$              ;;BR IF NO
3139  014742  116663  177777  177776            MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
3140  014750  105013                    9$:     CLRB    (R3)            ;;SET THE TERMINATOR
3141  014752  012605                            MOV     (SP)+,R5        ;;POP STACK INTO R5
3142  014754  012603                            MOV     (SP)+,R3        ;;POP STACK INTO R3
3143  014756  012602                            MOV     (SP)+,R2        ;;POP STACK INTO R2
3144  014760  012601                            MOV     (SP)+,R1        ;;POP STACK INTO R1
3145  014762  012600                            MOV     (SP)+,R0        ;;POP STACK INTO R0
3146  014764  104401  015012                    TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
3147  014770  016666  000002  000004            MOV     2(SP),4(SP)     ;;ADJUST THE STACK
3148  014776  012616                            MOV     (SP)+,(SP)
3149  015000  000002                            RTI                     ;;RETURN TO USER
3150  015002  023420                    $DTBL:  10000.
3151  015004  001750                            1000.
3152  015006  000144                            100.
3153  015010  000012                            10.
3154  015012  000004                    $DBLK:  .BLKW   4
3155                                    .SBTTL  TTY INPUT ROUTINE
3156
3157                                    ;;**************************************************************************
3158                                    .ENABL  LSB
3159  015022  000000                    $TKCNT: .WORD   0               ;;NUMBER OF ITEMS IN QUEUE
3160  015024  000000                    $TKQIN: .WORD   0               ;;INPUT POINTER
3161  015026  000000                    $TKQOUT: .WORD  0               ;;OUTPUT POINTER
```

```
3162  015030  000001                  $TKQSRT: .BLKB  1               ;;TTY KEYBOARD QUEUE
3163          015031                  $TKQEND=.
3164          015032                  .EVEN
3165
3166                                  ;*TK INITIALIZE ROUTINE
3167                                  ;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
3168                                  ;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
3169                                  ;
3170                                  ;*CALL:
3171                                  ;*      JSR     PC,$TKINT
3172                                  ;*      RETURN
3173                                  ;
3174  015032  005037  015022          $TKINT: CLR     $TKCNT          ;;CLEAR COUNT OF ITEMS IN QUEUE
3175  015036  012737  015030  015024          MOV     #$TKQSRT,$TKQIN ;;MOVE THE STARTING ADDRESS OF THE
3176  015044  013737  015024  015026          MOV     $TKQIN,$TKQOUT  ;;QUEUE INTO THE INPUT & OUTPUT POINTERS.
3177  015052  012737  015102  000060          MOV     #$TKSRV,@#TKVEC ;;INITIALIZE THE KEYBOARD VECTOR
3178  015060  012737  000200  000062          MOV     #200,@#TKVEC+2  ;;"BR" LEVEL 4
3179  015066  005777  164054                  TST     @$TKB           ;;CLEAR DONE FLAG
3180  015072  012777  000100  164044          MOV     #100,@$TKS      ;;ENABLE TTY KEYBOARD INTERRUPT
3181  015100  000207                          RTS     PC              ;;RETURN TO CALLER
3182
3183                                  ;*TK SERVICE ROUTINE
3184                                  ;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
3185                                  ;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
3186                                  ;*IT IN THE QUEUE.
3187                                  ;
3188  015102  117746  164040          $TKSRV: MOVB    @$TKB,-(SP)     ;;PICKUP THE CHARACTER
3189  015106  042716  177600                  BIC     #^C177,(SP)     ;;STRIP THE JUNK
3190  015112  021627  000007          1$:     CMP     (SP),#7         ;;IS IT A CONTROL G?
3191  015116  001004                          BNE     2$              ;;BRANCH IF NO
3192  015120  022737  000176  001140          CMP     #SWREG,SWR      ;;IS SOFT-SWR SELECTED?
3193  015126  001500                          BEQ     6$              ;;GO TO SWR CHANGE
3194
3195  015130                          2$:
3196  015130  022737  000001  015022          CMP     #1,$TKCNT       ;;IS THE QUEUE FULL?
3197  015136  001004                          BNE     3$              ;;BRANCH IF NO
3198  015140  104401  001202                  TYPE    ,$BELL          ;;RING THE TTY BELL
3199  015144  005726                          TST     (SP)+           ;;CLEAN CHARACTER OFF OF STACK
3200  015146  000451                          BR      5$              ;;EXIT
3201  015150  021627  000023          3$:     CMP     (SP),#23        ;;IS IT A CONTROL-S?
3202  015154  001021                          BNE     32$             ;;BRANCH IF NO
3203  015156  005077  163762                  CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
3204  015162  005726                          TST     (SP)+           ;;CLEAN CHAR OFF STACK
3205  015164  105777  163754          31$:    TSTB    @$TKS           ;;WAIT FOR A CHAR
3206  015170  100375                          BPL     31$             ;;LOOP UNTIL ITS THERE
3207  015172  117746  163750                  MOVB    @$TKB,-(SP)     ;;GET THE CHARACTER
3208  015176  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
3209  015202  022627  000021                  CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
3210  015206  001366                          BNE     31$             ;;BRANCH IF NO
3211  015210  012777  000100  163726          MOV     #100,@$TKS      ;;REENABLE TTY KEYBOARD INTERRUPTS
3212  015216  000002                          RTI                     ;;RETURN
3213  015220  005237  015022          32$:    INC     $TKCNT          ;;COUNT THIS CHARACTER
3214  015224  021627  000140                  CMP     (SP),#140       ;;IS IT UPPER CASE?
3215  015230  002405                          BLT     4$              ;;BRANCH IF YES
3216  015232  021627  000175                  CMP     (SP),#175       ;;IS IT A SPECIAL CHAR?
3217  015236  003002                          BGT     4$              ;;BRANCH IF YES
```

```
3218  015240  042716  000040          4$:     BIC     #40,(SP)        ;;MAKE IT UPPER CASE
3219  015244  112677  177554                  MOVB    (SP)+,@STKQIN   ;;AND PUT IT IN QUEUE
3220  015250  005237  015024                  INC     STKQIN          ;;UPDATE THE POINTER
3221  015254  023727  015024  015031          CMP     STKQIN,#STKQEND ;;GO OFF THE END?
3222  015262  001003                          BNE     5$              ;;BRANCH IF NO
3223  015264  012737  015030  015024          MOV     #STKQSRT,STKQIN ;;RESET THE POINTER
3224  015272  000002                  5$:     RTI                     ;;RETURN
3225
3226                                  ;;**********************************************************************
3227                                  ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
3228                                  ;*ROUTINE IS ENTERED  FROM THE TRAP HANDLER, AND WILL
3229                                  ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP
3230                                  ;*CALL WHEN OPERATING IN TTY INTERRUPT MODE.
3231  015274  022737  000176  001140  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED
3232  015302  001104                          BNE     15$             ;;EXIT IF NOT
3233  015304  105777  163634                  TSTB    @STKS           ;;IS A CHAR WAITING?
3234  015310  100101                          BPL     15$             ;;IF NOT, EXIT
3235  015312  117746  163630                  MOVB    @STKB,-(SP)     ;;YES
3236  015316  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT  7-BIT ASCII
3237  015322  021627  000007                  CMP     (SP),#7         ;;IS IT A CONTROL-G?
3238  015326  001300                          BNE     2$              ;;IF NOT, PUT IT IN THE TTY QUEUE
3239                                                                  ;;AND EXIT
3240
3241                                  ;;**********************************************************************
3242                                  ;*CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
3243                                  ;*ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
3244                                  ;*CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.
3245  015330  123727  001134  000001  6$:     CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
3246  015336  001674                          BEQ     2$              ;;BRANCH IF YES
3247  015340  005726                          TST     (SP)+           ;;CLEAR CONTROL-G OFF STACK
3248  015342  004737  015032                  JSR     PC,$TKINT       ;;FLUSH THE TTY INPUT QUEUE
3249  015346  005077  163572                  CLR     @#TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
3250  015352  112737  000001  001135          MOVB    #1,$INTAG       ;;SET INTERRUPT MODE INDICATOR
3251
3252  015360  104401  016136                  TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
3253  015364  104401  016143          $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
3254  015370  013746  000176                  MOV     $WREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
3255  015374  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
3256  015376  104401  016154                  TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
3257  015402  005046                  19$:    CLR     -(SP)           ;;CLEAR COUNTER
3258  015404  005046                          CLR     -(SP)           ;;THE NEW SWR
3259  015406  105777  163532          7$:     TSTB    @STKS           ;;CHAR THERE?
3260  015412  100375                          BPL     7$              ;;IF NOT TRY AGAIN
3261
3262  015414  117746  163526                  MOVB    @STKB,-(SP)     ;;PICK UP CHAR
3263  015420  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
3264
3265
3266
3267  015424  021627  000025          9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
3268  015430  001005                          BNE     10$             ;;BRANCH IF NOT
3269  015432  104401  016131                  TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (^U)
3270  015436  062706  000006          20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
3271  015442  000757                          BR      19$             ;;LET'S TRY IT AGAIN
3272
3273
```

```
3274  015444  021627  000015          10$:    CMP     (SP),#15        ;;IS IT A <CR>?
3275  015450  001022                          BNE     16$             ;;BRANCH IF NO
3276  015452  005766  000004                  TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
3277  015456  001403                          BEQ     11$             ;;BRANCH IF YES
3278  015460  016677  000002  163452          MOV     2(SP),@SWR      ;;SAVE NEW SWR
3279  015466  062706  000006          11$:    ADD     #6,SP           ;;CLEAR UP STACK
3280  015472  104401  001207          14$:    TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
3281  015476  123727  001135  000001          CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
3282  015504  001003                          BNE     15$             ;;BRANCH IF NOT
3283  015506  012777  000100  163430          MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
3284  015514  000002          15$:    RTI                             ;;RETURN
3285  015516  004737  014300          16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
3286  015522  021627  000060                  CMP     (SP),#60        ;;CHAR < 0?
3287  015526  002420                          BLT     18$             ;;BRANCH IF YES
3288  015530  021627  000067                  CMP     (SP),#67        ;;CHAR > 7?
3289  015534  003015                          BGT     18$             ;;BRANCH IF YES
3290  015536  042726  000060                  BIC     #60,(SP)+       ;;STRIP-OFF ASCII
3291  015542  005766  000002                  TST     2(SP)           ;;IS THIS THE FIRST CHAR
3292  015546  001403                          BEQ     17$             ;;BRANCH IF YES
3293  015550  006316                          ASL     (SP)            ;;NO, SHIFT PRESENT
3294  015552  006316                          ASL     (SP)            ;;   CHAR OVER TO MAKE
3295  015554  006316                          ASL     (SP)            ;;   ROOM FOR NEW ONE.
3296  015556  005266  000002          17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
3297  015562  056616  177776                  BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
3298  015566  000707                          BR      7$              ;;GET THE NEXT ONE
3299  015570  104401  001206          18$:    TYPE    ,$QUES          ;;TYPE ?<CR><LF>
3300  015574  000720                          BR      20$             ;;SIMULATE CONTROL-U
3301                                  .DSABL  LSB
3302
3303
3304                                  ;;*****************************************************************
3305                                  ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
3306                                  ;*CALL:
3307                                  ;*      RDCHR                   ;;GET A CHARACTER FROM THE QUEUE
3308                                  ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
3309                                  ;*                              ;;WITH PARITY BIT STRIPPED OFF
3310                                  ;
3311
3312  015576  011646          $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC AND
3313  015600  016666  000004  000002          MOV     4(SP),2(SP)     ;;THE PS
3314  015606  005066  000004                  CLR     4(SP)           ;;GET READY FOR A CHARACTER
3315  015612  005046                          CLR     -(SP)           ;;PUT NEW PS ON STACK
3316  015614  012746  015622                  MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
3317  015620  000002                          RTI                     ;;POP NEW PC AND PS
3318  015622                          64$:
3319  015622  005737  015022          1$:     TST     $TKCNT          ;;WAIT ON A CHARACTER
3320  015626  001775                          BEQ     1$
3321  015630  005337  015022                  DEC     $TKCNT          ;;DECREMENT THE COUNTER
3322  015634  117766  177166  000004          MOVB    @$TKQOUT,4(SP)  ;;GET ONE CHARACTER
3323  015642  005237  015026                  INC     $TKQOUT         ;;UPDATE THE POINTER
3324  015646  023727  015026  015031          CMP     $TKQOUT,#$TKQEND ;;DID IT GO OFF OF THE END?
3325  015654  001003                          BNE     2$              ;;BRANCH IF NO
3326  015656  012737  015030  015026          MOV     #$TKQ$RT,$TKQOUT ;;RESET THE POINTER
3327  015664  000002          2$:     RTI                             ;;RETURN
3328                                  ;;*****************************************************************
3329                                  ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
```

```
                                        ;*CALL:
                                        ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
                                        ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
                                        ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S

015666  010346                  $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
015670  005046                          CLR     -(SP)           ;;CLEAR THE RUBOUT KEY
015672  012703  016122            1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
015676  022703  016131            2$:     CMP     #$TTYIN+7,R3    ;;BUFFER FULL?
015702  101456                          BLOS    4$              ;;BR IF YES
015704  104410                          RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
015706  112613                          MOVB    (SP)+,(R3)      ;;GET CHARACTER
015710  122713  000177            10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
015714  001022                          BNE     5$              ;;BR IF NO
015716  005716                          TST     (SP)            ;;IS THIS THE FIRST RUBOUT?
015720  001007                          BNE     6$              ;;BR IF NO
015722  112737  000134  016120          MOVB    #'\,9$          ;;TYPE A BACK SLASH
015730  104401  016120                  TYPE    ,9$
015734  012716  177777                  MOV     #-1,(SP)        ;;SET THE RUBOUT KEY
015740  005303                  6$:     DEC     R3              ;;BACKUP BY ONE
015742  020327  016122                  CMP     R3,#$TTYIN      ;;STACK EMPTY?
015746  103434                          BLO     4$              ;;BR IF YES
015750  111337  016120                  MOVB    (R3),9$         ;;SETUP TO TYPEOUT THE DELETED CHAR.
015754  104401  016120                  TYPE    ,9$             ;;GO TYPE
015760  000746                          BR      2$              ;;GO READ ANOTHER CHAR.
015762  005716                  5$:     TST     (SP)            ;;RUBOUT KEY SET?
015764  001406                          BEQ     7$              ;;BR IF NO
015766  112737  000134  016120          MOVB    #'\,9$          ;;TYPE A BACK SLASH
015774  104401  016120                  TYPE    ,9$
016000  005016                          CLR     (SP)            ;;CLEAR THE RUBOUT KEY
016002  122713  000025            7$:     CMPB    #25,(R3)        ;;IS CHARACTER A CTRL U?
016006  001003                          BNE     8$              ;;BR IF NO
016010  104401  016131                  TYPE    ,$CNTLU         ;;TYPE A CONTROL "U"
016014  000726                          BR      1$              ;;GO START OVER
016016  122713  000022            8$:     CMPB    #22,(R3)        ;;IS CHARACTER A "↑R"?
016022  001011                          BNE     3$              ;;BRANCH IF NO
016024  105013                          CLRB    (R3)            ;;CLEAR THE CHARACTER
016026  104401  001207                  TYPE    ,$CRLF          ;;TYPE A "CR" & "LF"
016032  104401  016122                  TYPE    ,$TTYIN         ;;TYPE THE INPUT STRING
016036  000717                          BR      2$              ;;GO PICKUP ANOTHER CHACTER
016040  104401  001206            4$:     TYPE    ,$QUES          ;;TYPE A '?'
016044  000712                          BR      1$              ;;CLEAR THE BUFFER AND LOOP
016046  111337  016120            3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
016052  104401  016120                  TYPE    ,9$
016056  122723  000015                  CMPB    #15,(R3)+       ;;CHECK FOR RETURN
016062  001305                          BNE     2$              ;;LOOP IF NOT RETURN
016064  105063  177777                  CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
016070  104401  001210                  TYPE    ,$LF            ;;TYPE A LINE FEED
016074  005726                          TST     (SP)+           ;;CLEAN RUBOUT KEY FROM THE STACK
016076  012603                          MOV     (SP)+,R3        ;;RESTORE R3
016100  011646                          MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
016102  016666  000004  000002          MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
016110  012766  016122  000004          MOV     #$TTYIN,4(SP)
016116  000002                          RTI                     ;;RETURN
016120      000                 9$:     .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
016121      000                         .BYTE   0               ;;TERMINATOR
```

```
                                        $TTYIN: .BLKB   7               ;;RESERVE 7 BYTES FOR TTY INPUT
                                        $CNTLU: .ASCIZ  /↑U/<15><12>    ;;CONTROL "U"
                                        $CNTLG: .ASCIZ  /↑G/<15><12>    ;;CONTROL "G"
                                        $MSWR:  .ASCIZ  <15><12>/SWR = /
                                        $MNEW:  .ASCIZ  /  NEW = /

                                        .EVEN
                                        .SBTTL  READ AN OCTAL NUMBER FROM THE TTY

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

016166  011646          $RDOCT: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR THE
016170  016666  000004  000002  MOV     4(SP),2(SP)     ;;INPUT NUMBER
016176  010046          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
016200  010146          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
016202  010246          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
016204  104411          1$:     RDLIN           ;;READ AN ASCIZ LINE
016206  012600          MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
016210  010037  016314  MOV     R0,5$           ;;AND SAVE IT
016214  005001          CLR     R1              ;;CLEAR DATA WORD
016216  005002          CLR     R2
016220  112046          2$:     MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
016222  001420          BEQ     3$              ;;IF ZERO GET OUT
016224  122716  000060  CMPB    #'0,(SP)        ;;MAKE SURE THIS CHARACTER
016230  003026          BGT     4$              ;;IS AN OCTAL DIGIT
016232  122716  000067  CMPB    #'7,(SP)
016236  002423          BLT     4$
016240  006301          ASL     R1              ;;*2
016242  006102          ROL     R2
016244  006301          ASL     R1              ;;*4
016246  006102          ROL     R2
016250  006301          ASL     R1              ;;*8
016252  006102          ROL     R2
016254  042716  177770  BIC     #↑C7,(SP)       ;;STRIP THE ASCII JUNK
016260  062601          ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
016262  000756          BR      2$              ;;LOOP
016264  005726          3$:     TST     (SP)+   ;;CLEAN TERMINATOR FROM STACK
016266  010166  000012  MOV     R1,12(SP)       ;;SAVE THE RESULT
016272  010237  016324  MOV     R2,$HIOCT
016276  012602          MOV     (SP)+,R2        ;;POP STACK INTO R2
016300  012601          MOV     (SP)+,R1        ;;POP STACK INTO R1
016302  012600          MOV     (SP)+,R0        ;;POP STACK INTO R0
016304  000002          RTI                     ;;RETURN
016306  005726          4$:     TST     (SP)+   ;;CLEAN PARTIAL FROM STACK
016310  105010          CLRB    (R0)            ;;SET A TERMINATOR
```

```
                        TYPE                    ;;TYPE UP THRU THE BAD CHAR.
5$:     .WORD   0
                        TYPE    $QUES           ;;"?" "CR" & "LF"
                        BR      1$              ;;TRY AGAIN
$HIOCT: .WORD   0               ;;HIGH ORDER BITS GO HERE
.SBTTL  SAVE AND RESTORE R0-R5 ROUTINES

;;*****************************************************************
;*SAVE R0-R5
;*CALL:
;*      SAVREG
;*UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
;*
;*TOP---(+16)
;* +2---(+18)
;* +4---R5
;* +6---R4
;* +8---R3
;*+10---R2
;*+12---R1
;*+14---R0

$SAVREG:
                MOV     R0,-(SP)        ;;PUSH R0 ON STACK
                MOV     R1,-(SP)        ;;PUSH R1 ON STACK
                MOV     R2,-(SP)        ;;PUSH R2 ON STACK
                MOV     R3,-(SP)        ;;PUSH R3 ON STACK
                MOV     R4,-(SP)        ;;PUSH R4 ON STACK
                MOV     R5,-(SP)        ;;PUSH R5 ON STACK
                MOV     22(SP),-(SP)    ;;SAVE PS OF MAIN FLOW
                MOV     22(SP),-(SP)    ;;SAVE PC OF MAIN FLOW
                MOV     22(SP),-(SP)    ;;SAVE PS OF CALL
                MOV     22(SP),-(SP)    ;;SAVE PC OF CALL
                RTI

;*RESTORE R0-R5
;*CALL:
;*      RESREG
$RESREG:
                MOV     (SP)+,22(SP)    ;;RESTORE PC OF CALL
                MOV     (SP)+,22(SP)    ;;RESTORE PS OF CALL
                MOV     (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
                MOV     (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
                MOV     (SP)+,R5        ;;POP STACK INTO R5
                MOV     (SP)+,R4        ;;POP STACK INTO R4
                MOV     (SP)+,R3        ;;POP STACK INTO R3
                MOV     (SP)+,R2        ;;POP STACK INTO R2
                MOV     (SP)+,R1        ;;POP STACK INTO R1
                MOV     (SP)+,R0        ;;POP STACK INTO R0
                RTI
.SBTTL  TRAP DECODER

;;*****************************************************************
;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
```

```
3498                                    ;*GO TO THAT ROUTINE.
3499
3500  016422  010046                    $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
3501  016424  016600  000002                    MOV     2(SP),R0        ;;GET TRAP ADDRESS
3502  016430  005740                            TST     -(R0)           ;;BACKUP BY 2
3503  016432  111000                            MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
3504  016434  006300                            ASL     R0              ;;POSITION FOR INDEXING
3505  016436  016000  016456                    MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
3506  016442  000200                            RTS     R0              ;;GO TO ROUTINE
3507
3508
3509                                    ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
3510
3511  016444  011646                    $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
3512  016446  016666  000004  000002            MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
3513  016454  000002                            RTI                     ;;RESTORE THE PSW
3514
3515                                    .SBTTL  TRAP TABLE
3516
3517                                    ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
3518                                    ;*BY THE "TRAP" INSTRUCTION.
3519
3520                                    ;       ROUTINE
3521                                    ;       -------
3522  016456  016444                    $TRPAD: .WORD   $TRAP2
3523  016460  014130                            $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
3524  016462  014374                            $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
3525  016464  014350                            $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
3526  016466  014410                            $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
3527  016470  014576                            $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
3528
3529  016472  015364                            $GTSWR  ;;CALL=GTSWR    TRAP+6(104406)  GET SOFT-SWR SETTING
3530
3531  016474  015274                            $CKSWR  ;;CALL=CKSWR    TRAP+7(104407)  TEST FOR CHANGE IN SOFT-SWR
3532  016476  015576                            $RDCHR  ;;CALL=RDCHR    TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
3533  016500  015666                            $RDLIN  ;;CALL=RDLIN    TRAP+11(104411) TTY TYPEIN STRING ROUTINE
3534  016502  016166                            $RDOCT  ;;CALL=RDOCT    TRAP+12(104412) READ AN OCTAL NUMBER FROM TTY
3535  016504  016326                            $SAVREG ;;CALL=SAVREG   TRAP+13(104413) SAVE R0-R5 ROUTINE
3536  016506  016364                            $RESREG ;;CALL=RESREG   TRAP+14(104414) RESTORE R0-R5 ROUTINE
3537
3538                                    ;;**************************************************************
3539
3540                                    .SBTTL  TELETYPE MESSAGES
3541
3542                                    ;;**************************************************************
3543
3544  016510  005015  046412  044501    TITLE:  .ASCII  <CR><LF><LF>/MAINDEC-11-DZRJF-A/<CR><LF>
3545  016516  042116  041505  030455
3546  016524  026461  055104  045122
3547  016532  026506  006501    012
3548  016537    122   030115  020063            .ASCIZ  @RM03 DUAL CONTROLLER LOGIC TEST - PART 2@<CR><LF><LF>
3549  016544  052504  046101  041440
3550  016552  047117  051124  046117
3551  016560  042514  020122  047514
3552  016566  044507  020103  042524
3553  016574  052123  026440  050040
```

```
3554  016602  051101  020124  006462
3555  016610  005012     000
3556  016613     015  042412  052116  ENTERA: .ASCIZ  <CR><LF>/ENTER DRIVE ADDRESS: /
3557  016620  051105  042040  044522
3558  016626  042526  040440  042104
3559  016634  042522  051523  020072
3560  016642     000
3561  016643     111  053116  046101  ADRERR: .ASCIZ  /INVALID ADDRESS/<CR><LF>
3562  016650  042111  040440  042104
3563  016656  042522  051523  005015
3564  016664     000
3565  016665     015  050012  051117  PORTAIS: .ASCIZ <CR><LF>/PORT A ADDRESS IS: /
3566  016672  020124  020101  042101
3567  016700  051104  051505  020123
3568  016706  051511  020072     000
3569  016713     015  050012  051117  PORTBIS: .ASCIZ <CR><LF>/PORT B ADDRESS IS: /
3570  016720  020124  020102  042101
3571  016726  051104  051505  020123
3572  016734  051511  020072     000
3573  016741     015  051412  051531  NOCLOCK: .ASCIZ <CR><LF>/SYSTEM MUST HAVE 'L' OR 'P' CLOCK/<CR><LF><LF>
3574  016746  042524  020115  052515
3575  016754  052123  044040  053101
3576  016762  020105  046047  020047
3577  016770  051117  023440  023520
3578  016776  041440  047514  045503
3579  017004  005015  000012
3580  017010  042412  052116  051105  TESTNO: .ASCIZ  <LF>/ENTER TEST #: /
3581  017016  052040  051505  020124
3582  017024  035043  000040
3583  017030  047111  040526  044514  BADNO:  .ASCIZ  /INVALID TEST NUMBER/<CR><LF>
3584  017036  020104  042524  052123
3585  017044  047040  046525  042502
3586  017052  006522  000012
3587  017056  005015  052012  042510  ADDRIS: .ASCIZ  <CR><LF><LF>/THE PRESENT ADDRESS OF THE RH11 (RMCS1) IS: /
3588  017064  050040  042522  042523
3589  017072  052116  040440  042104
3590  017100  042522  051523  047440
3591  017106  020106  044124  020105
3592  017114  044122  030461  024040
3593  017122  046522  051503  024461
3594  017130  044440  035123  000040
3595  017136  042412  052116  051105  NTRH11: .ASCIZ  <LF>/ENTER NEW RH11 ADDRESS: /
3596  017144  047040  053505  051040
3597  017152  030510  020061  042101
3598  017160  051104  051505  035123
3599  017166  000040
3600  017170  005015  050012  042522  STANDBY: .ASCII <CR><LF><LF>/PRESS 'STANDBY' ON DRIVE/
3601  017176  051523  023440  052123
3602  017204  047101  041104  023531
3603  017212  047440  020116  051104
3604  017220  053111     105
3605  017223     015  050012  047522          .ASCIZ  <CR><LF>/PROGRAM WILL LOOP WAITING FOR 'MOL' TO SET FROM PORT /
3606  017230  051107  046501  053440
3607  017236  046111  020114  047514
3608  017244  050117  053440  044501
3609  017252  044524  043516  043040
```

```
3610  017260  051117  023440  047515
3611  017266  023514  052040  020117
3612  017274  042523  020124  051106
3613  017302  046517  050040  051117
3614  017310  020124     000
3615  017313     015  005012  042522  SWTCHN: .ASCIZ  <CR><LF><LF>@RETURN 'CONTROLLER SELECT' SWITCH ON DRIVE TO 'A/B'@
3616  017320  052524  047122  023440
3617  017326  047503  052116  047522
3618  017334  046114  051105  051440
3619  017342  046105  041505  023524
3620  017350  051440  044527  041524
3621  017356  020110  047117  042040
3622  017364  044522  042526  052040
3623  017372  020117  040447  041057
3624  017400  000047
3625  017402  005015  052012  051125  SWTCHA: .ASCIZ  <CR><LF><LF>/TURN 'CONTROLLER SELECT' SWITCH ON DRIVE TO 'A'/
3626  017410  020116  041447  047117
3627  017416  051124  046117  042514
3628  017424  020122  042523  042514
3629  017432  052103  020047  053523
3630  017440  052111  044103  047440
3631  017446  020116  051104  053111
3632  017454  020105  047524  023440
3633  017462  023501     000
3634  017465     015  005012  052524  SWTCHB: .ASCIZ  <CR><LF><LF>/TURN 'CONTROLLER SELECT' SWITCH ON DRIVE TO 'B'/
3635  017472  047122  023440  047503
3636  017500  052116  047522  046114
3637  017506  051105  051440  046105
3638  017514  041505  023524  051440
3639  017522  044527  041524  020110
3640  017530  047117  042040  044522
3641  017536  042526  052040  020117
3642  017544  041047  000047
3643  017550  005015  044124  047105  CONTUE: .ASCIZ  <CR><LF>/THEN PRESS 'CONT' ON THE PROCESSOR/<CR><LF>
3644  017556  050040  042522  051523
3645  017564  023440  047503  052116
3646  017572  020047  047117  052040
3647  017600  042510  050040  047522
3648  017606  042503  051523  051117
3649  017614  005015     000
3650  017617     015  005012  052123  CYCLED: .ASCIZ  <CR><LF><LF>/STOP THE DRIVE/
3651  017624  050117  052040  042510
3652  017632  042040  044522  042526
3653  017640     000
3654  017641     015  005012  052123  CYCLEU: .ASCIZ  <CR><LF><LF>/START THE DRIVE - THE PROGRAM WILL WAIT FOR 'MOL' TO SET/
3655  017646  051101  020124  044124
3656  017654  020105  051104  053111
3657  017662  020105  020055  044124
3658  017670  020105  051120  043517
3659  017676  040522  020115  044527
3660  017704  046114  053440  044501
3661  017712  020124  047506  020122
3662  017720  046447  046117  020047
3663  017726  047524  051440  052105
3664  017734     000
3665
```

```
                                ;;*****************************************************************
                                ;
                                .SBTTL  TEST ERROR MESSAGES
                                ;
                                ;;*****************************************************************

017735  104  044522  042526  EM1:    .ASCIZ  /DRIVE IS NON-EXISTENT ('NED' BIT SET)/
017742  044440  020123  047516
017750  026516  054105  051511
017756  042524  052116  024040
017764  047047  042105  020047
017772  044502  020124  042523
020000  024524  000
020003  127  047522  043516  EM2:    .ASCIZ  /WRONG DRIVE TYPE/
020010  042040  044522  042526
020016  052040  050131  000105
020024  047503  052116  047522  EM3:    .ASCIZ  @CONTROLLER SELECT SWITCH ON DRIVE NOT IN 'A/B'@
020032  046114  051105  051440
020040  046105  041505  020124
020046  053523  052111  044103
020054  047440  020116  051104
020062  053111  020105  047516
020070  020124  047111  023440
020076  027501  023502  000
020103  104  044522  042526  EM4:    .ASCIZ  /DRIVE NOT ON LINE/
020110  047040  052117  047440
020116  020116  044514  042516
020124  000
020125  123  051105  040511  EM5:    .ASCIZ  /SERIAL NUMBER READ THROUGH EACH PORT NOT THE SAME/
020132  020114  052516  041115
020140  051105  051040  040505
020146  020104  044124  047522
020154  043525  020110  040505
020162  044103  050040  051117
020170  020124  047516  020124
020176  044124  020105  040523
020204  042515  000
020207  124  046511  047505  EM6:    .ASCIZ  /TIMEOUT HAS NOT OCCURRED WITHIN 2 SECONDS/
020214  052125  044040  051501
020222  047040  052117  047440
020230  041503  051125  042522
020236  020104  044527  044124
020244  047111  031040  051440
020252  041505  047117  051504
020260  000
020261  124  046511  047505  EM7:    .ASCIZ  /TIMEOUT ONE-SHOT IS LESS THAN 500 MS/
020266  052125  047440  042516
020274  051455  047510  020124
020302  051511  046040  051505
020310  020123  044124  047101
```

```
3722  020316  032440  030060  046440
3723  020324  000123
3724
3725  020326  042522  042101  047111  EM10:   .ASCIZ  /READIN PRESET DOES NOT SET VOLUME VALID FOR THE PORT/
3726  020334  050040  042522  042523
3727  020342  020124  047504  051505
3728  020350  047040  052117  051440
3729  020356  052105  053040  046117
3730  020364  046525  020105  040526
3731  020372  044514  020104  047506
3732  020400  020122  044124  020105
3733  020406  047520  052122     000
3734
3735  020413     047  047507  020047  EM11:   .ASCIZ  /'GO' BIT RESET DURING UNLOAD COMMAND/
3736  020420  044502  020124  042522
3737  020426  042523  020124  052504
3738  020434  044522  043516  052440
3739  020442  046116  040517  020104
3740  020450  047503  046515  047101
3741  020456  000104
3742
3743  020460  047111  047503  051122  EM12:   .ASCIZ  /INCORRECT STATUS DURING UNLOAD COMMAND/
3744  020466  041505  020124  052123
3745  020474  052101  051525  042040
3746  020502  051125  047111  020107
3747  020510  047125  047514  042101
3748  020516  041440  046517  040515
3749  020524  042116     000
3750
3751  020527     104  044522  042526  EM13:   .ASCIZ  /DRIVE DID NOT RETURN TO NEUTRAL AFTER UNLOAD COMMAND/
3752  020534  042040  042111  047040
3753  020542  052117  051040  052105
3754  020550  051125  020116  047524
3755  020556  047040  052505  051124
3756  020564  046101  040440  052106
3757  020572  051105  052440  046116
3758  020600  040517  020104  047503
3759  020606  046515  047101  000104
3760
3761  020614  052101  042524  052116  EM14:   .ASCIZ  /ATTENTION BIT SET ON 'OPPOSITE PORT' AFTER UNLOAD/
3762  020622  047511  020116  044502
3763  020630  020124  042523  020124
3764  020636  047117  023440  050117
3765  020644  047520  044523  042524
3766  020652  050040  051117  023524
3767  020660  040440  052106  051105
3768  020666  052440  046116  040517
3769  020674  000104
3770
3771  020676  052101  042524  052116  EM15:   .ASCIZ  /ATTENTION BIT NOT SET ON PORT WHICH ISSUED 'UNLOAD'/
3772  020704  047511  020116  044502
3773  020712  020124  047516  020124
3774  020720  042523  020124  047117
3775  020726  050040  051117  020124
3776  020734  044127  041511  020110
3777  020742  051511  052523  042105
```

```
3778  020750  023440  047125  047514
3779  020756  042101  000047
3780
3781  020762  051104  053111  020105  EM16:   .ASCII  /DRIVE NOT IN NEUTRAL AFTER UNLOAD WITH 'CONTROLLER/<CR><LF>
3782  020770  047516  020124  047111
3783  020776  047040  052505  051124
3784  021004  046101  040440  052106
3785  021012  051105  052440  046116
3786  021020  040517  020104  044527
3787  021026  044124  023440  047503
3788  021034  052116  047522  046114
3789  021042  051105  005015
3790  021046  042523  042514  052103          .ASCIZ  @SELECT' SWITCH MOVED FROM 'A/B'@
3791  021054  020047  053523  052111
3792  021062  044103  046440  053117
3793  021070  042105  043040  047522
3794  021076  020115  040447  041057
3795  021104  000047
3796
3797  021106  051104  053111  020105  EM17:   .ASCIZ  /DRIVE LOCKED ON PORT 'A' BY SWITCH WHILE CYCLED UP/
3798  021114  047514  045503  042105
3799  021122  047440  020116  047520
3800  021130  052122  023440  023501
3801  021136  041040  020131  053523
3802  021144  052111  044103  053440
3803  021152  044510  042514  041440
3804  021160  041531  042514  020104
3805  021166  050125     000
3806
3807  021171     104  044522  042526  EM20:   .ASCIZ  /DRIVE LOCKED ON PORT 'B' BY SWITCH WHILE CYCLED UP/
3808  021176  046040  041517  042513
3809  021204  020104  047117  050040
3810  021212  051117  020124  041047
3811  021220  020047  054502  051440
3812  021226  044527  041524  020110
3813  021234  044127  046111  020105
3814  021242  054503  046103  042105
3815  021250  052440  000120
3816
3817  021254  052123  052101  051525  EM21:   .ASCIZ  /STATUS INCORRECT FOR PORT AFTER CYCLE UP/
3818  021262  044440  041516  051117
3819  021270  042522  052103  043040
3820  021276  051117  050040  051117
3821  021304  020124  043101  042524
3822  021312  020122  054503  046103
3823  021320  020105  050125     000
3824
3825  021325     122  043505  051511  EM22:   .ASCIZ  /REGISTER CONTENTS SEEN WHEN DRIVE SWITCHED ON 'OPPOSITE' PORT/
3826  021332  042524  020122  047503
3827  021340  052116  047105  051524
3828  021346  051440  042505  020116
3829  021354  044127  047105  042040
3830  021362  044522  042526  051440
3831  021370  044527  041524  042510
3832  021376  020104  047117  023440
3833  021404  050117  047520  044523
```

```
3834  021412  042524  020047  047520
3835  021420  052122     000
3836
3837  021423     047  042516  023504  EM23:   .ASCIZ  /'NED' NOT SET WHEN RMDS1 ACCESSED THROUGH PORT NOT SWITCHED/
3838  021430  047040  052117  051440
3839  021436  052105  053440  042510
3840  021444  020116  046522  051504
3841  021452  020061  041501  042503
3842  021460  051523  042105  052040
3843  021466  051110  052517  044107
3844  021474  050040  051117  020124
3845  021502  047516  020124  053523
3846  021510  052111  044103  042105
3847  021516     000
3848
3849  021517     104  044522  042526  EM24:   .ASCIZ  /DRIVE SWITCHED TO LOCKED OUT PORT WHEN RELEASED/
3850  021524  051440  044527  041524
3851  021532  042510  020104  047524
3852  021540  046040  041517  042513
3853  021546  020104  052517  020124
3854  021554  047520  052122  053440
3855  021562  042510  020116  042522
3856  021570  042514  051501  042105
3857  021576     000
3858
3859  021577     122  030510  020061  EM25:   .ASCIZ  /RH11 DIDN'T RESPOND TO ADDRESSING/
3860  021604  044504  047104  052047
3861  021612  051040  051505  047520
3862  021620  042116  052040  020117
3863  021626  042101  051104  051505
3864  021634  044523  043516     000
3865
3866  021641     104  044522  042526  EM30:   .ASCIZ  /DRIVE NOT SEIZED BY PORT/
3867  021646  047040  052117  051440
3868  021654  044505  042532  020104
3869  021662  054502  050040  051117
3870  021670  000124
3871
3872  021672  051127  047117  020107  EM31:   .ASCIZ  /WRONG STATUS SEEN BY THE SEIZING PORT/
3873  021700  052123  052101  051525
3874  021706  051440  042505  020116
3875  021714  054502  052040  042510
3876  021722  051440  044505  044532
3877  021730  043516  050040  051117
3878  021736  000124
3879
3880  021740  042522  044507  052123  EM32:   .ASCIZ  /REGISTER CONTENTS WRONG/
3881  021746  051105  041440  047117
3882  021754  042524  052116  020123
3883  021762  051127  047117  000107
3884
3885  021770  047503  052116  047522  EM33:   .ASCIZ  /CONTROL BUS PARITY ERROR READING INDICATED REGISTER/
3886  021776  020114  052502  020123
3887  022004  040520  044522  054524
3888  022012  042440  051122  051117
3889  022020  051040  040505  044504
```

```
 3890  022026  043516  044440  042116
 3891  022034  041511  052101  042105
 3892  022042  051040  043505  051511
 3893  022050  042524  000122
 3894
 3895  022054  040503  023516  020124  EM34:   .ASCIZ  /CAN'T ACCESS DRIVE THROUGH EITHER PORT/
 3896  022062  041501  042503  051523
 3897  022070  042040  044522  042526
 3898  022076  052040  051110  052517
 3899  022104  044107  042440  052111
 3900  022112  042510  020122  047520
 3901  022120  052122     000
 3902
 3903  022123     104  044522  042526  EM35:   .ASCIZ  /DRIVE NOT IN NEUTRAL AFTER RELEASE - REQUEST NOT SET/
 3904  022130  047040  052117  044440
 3905  022136  020116  042516  052125
 3906  022144  040522  020114  043101
 3907  022152  042524  020122  042522
 3908  022160  042514  051501  020105
 3909  022166  020055  042522  052521
 3910  022174  051505  020124  047516
 3911  022202  020124  042523  000124
 3912
 3913  022210  051104  053111  020105  EM36:   .ASCIZ  /DRIVE NOT IN NEUTRAL AFTER TIMEOUT - REQUEST NOT SET/
 3914  022216  047516  020124  047111
 3915  022224  047040  052505  051124
 3916  022232  046101  040440  052106
 3917  022240  051105  052040  046511
 3918  022246  047505  052125  026440
 3919  022254  051040  050505  042525
 3920  022262  052123  047040  052117
 3921  022270  051440  052105     000
 3922
 3923  022275     122  043505  051511  EM37:   .ASCIZ  /REGISTER CONTENTS WRONG AFTER RELEASE OR TIMEOUT/
 3924  022302  042524  020122  047503
 3925  022310  052116  047105  051524
 3926  022316  053440  047522  043516
 3927  022324  040440  052106  051105
 3928  022332  051040  046105  040505
 3929  022340  042523  047440  020122
 3930  022346  044524  042515  052517
 3931  022354  000124
 3932
 3933  022356  051104  053111  020105  EM40:   .ASCIZ  /DRIVE IN NEUTRAL AFTER RELEASE - REQUEST SET/
 3934  022364  047111  047040  052505
 3935  022372  051124  046101  040440
 3936  022400  052106  051105  051040
 3937  022406  046105  040505  042523
 3938  022414  026440  051040  050505
 3939  022422  042525  052123  051440
 3940  022430  052105     000
 3941
 3942  022433     122  043505  051511  EM41:   .ASCIZ  /REGISTER WRONG AFTER RELEASE WITH REQUEST SET/
 3943  022440  042524  020122  051127
 3944  022446  047117  020107  043101
 3945  022454  042524  020122  042522
```

```
022462  042514  051501  020105
022470  044527  044124  051040
022476  050505  042525  052123
022504  051440  052105     000


022511     124  051505  020124  DH1:   .ASCIZ  /TEST #  ERR PC  PORT #  REG ADR CONTENTS/
022516  020043  042440  051122
022524  050040  020103  050040
022532  051117  020124  020043
022540  051040  043505  040440
022546  051104  041440  047117
022554  042524  052116  000123
022562  042524  052123  021440  DH2:   .ASCIZ  /TEST #  ERR PC  PORT #  REG ADR GOOD    BAD/
022570  020040  051105  020122
022576  041520  020040  047520
022604  052122  021440  020040
022612  042522  020107  042101
022620  020122  047507  042117
022626  020040  020040  040502
022634  000104
022636  042524  052123  021440  DH5:   .ASCIZ  /TEST #  ERR PC  REG ADR PORT A  PORT B/
022644  020040  051105  020122
022652  041520  020040  042522
022660  020107  042101  020122
022666  047520  052122  040440
022674  020040  047520  052122
022702  041040     000
022705     124  051505  020124  DH6:   .ASCIZ  /TEST #  ERR PC  PORT #/
022712  020043  042440  051122
022720  050040  020103  050040
022726  051117  020124  000043
022734  042524  052123  021440  DH7:   .ASCIZ  /TEST #  ERR PC  PORT #  TIME (IN MS)/
022742  020040  051105  020122
022750  041520  020040  047520
022756  052122  021440  020040
022764  044524  042515  024040
022772  047111  046440  024523
023000     000
023001     040  020040  020040  DH13:  .ASCII  /                SEIZE/<CR><LF>
023006  020040  020040  020040
023014  020040  020040  051440
023022  044505  042532  005015
023030  042524  052123  021440         .ASCIZ  /TEST #  ERR PC  PORT #/
023036  020040  051105  020122
023044  041520  020040  047520
023052  052122  021440     000
023057     040  020040  020040  DH14:  .ASCII  /                SEIZE   ERROR/<CR><LF>
023064  020040  020040  020040
023072  020040  020040  051440
023100  044505  042532  020040
023106  042440  051122  051117
023114  005015
023116  042524  052123  021440         .ASCIZ  /TEST #  ERR PC  PORT #  PORT #  REG ADR CONTENTS/
023124  020040  051105  020122
023132  041520  020040  047520
```

```
4002  023140  052122  021440  020040
4003  023146  047520  052122  021440
4004  023154  020040  042522  020107
4005  023162  042101  020122  047503
4006  023170  052116  047105  051524
4007  023176     000
4008  023177     124  051505  020124  DH17:   .ASCIZ  /TEST #  ERR PC/
4009  023204  020043  042440  051122
4010  023212  050040  000103
4011  023216  020040  020040  020040  DH24:   .ASCII  /               LOCKED  SWITCHED TO/<CR><LF>
4012  023224  020040  020040  020040
4013  023232  020040  020040  047514
4014  023240  045503  042105  020040
4015  023246  053523  052111  044103
4016  023254  042105  052040  006517
4017  023262     012
4018  023263     124  051505  020124          .ASCIZ  /TEST #  ERR PC  PORT #  PORT #/
4019  023270  020043  042440  051122
4020  023276  050040  020103  050040
4021  023304  051117  020124  020043
4022  023312  050040  051117  020124
4023  023320  000043
4024  023322  051044  040515  051104  DH25:   .ASCIZ  /$RMADR/
4025  023330     000
4026  023331     040  020040  020040  DH30:   .ASCII  /               SEIZE   ERROR/<CR><LF>
4027  023336  020040  020040  020040
4028  023344  020040  020040  051440
4029  023352  044505  042532  020040
4030  023360  042440  051122  051117
4031  023366  005015
4032  023370  042524  052123  021440          .ASCIZ  /TEST #  ERR PC  PORT #  PORT #  REG ADR GOOD    BAD/
4033  023376  020040  051105  020122
4034  023404  041520  020040  047520
4035  023412  052122  021440  020040
4036  023420  047520  052122  021440
4037  023426  020040  042522  020107
4038  023434  042101  020122  047507
4039  023442  042117  020040  020040
4040  023450  040502  000104
4041  023454  020040  020040  020040  DH34:   .ASCII  /               PORT A  PORT B/<CR><LF>
4042  023462  020040  020040  020040
4043  023470  020040  020040  047520
4044  023476  052122  040440  020040
4045  023504  047520  052122  041040
4046  023512  005015
4047  023514  042524  052123  021440          .ASCIZ  /TEST #  ERR PC  RMDS1   RMDS1/
4048  023522  020040  051105  020122
4049  023530  041520  020040  046522
4050  023536  051504  020061  020040
4051  023544  046522  051504  000061
4052  023552  020040  020040  020040  DH35:   .ASCII  /               RELSNG  ERROR/<CR><LF>
4053  023560  020040  020040  020040
4054  023566  020040  020040  042522
4055  023574  051514  043516  020040
4056  023602  051105  047522  006522
4057  023610     012
```

```
4058  023611    124  051505  020124          .ASCIZ  /TEST #  ERR PC  PORT #  PORT #/
4059  023616 020043  042440  051122
4060  023624 050040  020103  050040
4061  023632 051117  020124  020043
4062  023640 050040  051117  020124
4063  023646 000043
4064  023650 020040  020040  020040  DH37:   .ASCII  /                RELSNG  ERROR/<CR><LF>
4065  023656 020040  020040  020040
4066  023664 020040  020040  042522
4067  023672 051514  043516  020040
4068  023700 051105  047522  006522
4069  023706    012
4070  023707    124  051505  020124          .ASCIZ  /TEST #  ERR PC  PORT #  PORT #  REG ADR GOOD    BAD/
4071  023714 020043  042440  051122
4072  023722 050040  020103  050040
4073  023730 051117  020124  020043
4074  023736 050040  051117  020124
4075  023744 020043  051040  043505
4076  023752 040440  051104  043440
4077  023760 047517  020104  020040
4078  023766 041040  042101     000
4079  023773    040  020040  020040  DH40:   .ASCII  /                RELSNG  RQSTNG/<CR><LF>
4080  024000 020040  020040  020040
4081  024006 020040  020040  051040
4082  024014 046105  047123  020107
4083  024022 051040  051521  047124
4084  024030 006507     012
4085  024033    124  051505  020124          .ASCIZ  /TEST #  ERR PC  PORT #  PORT #/
4086  024040 020043  042440  051122
4087  024046 050040  020103  050040
4088  024054 051117  020124  020043
4089  024062 050040  051117  020124
4090  024070 000043
4091
4092                                  .EVEN
4093
4094  024072 001242  001116  001234  DT1:    .WORD   TSTNUM,SERRPC,PTNBR,SBDADR,SBDDAT,0
4095  024100 001122  001126  000000
4096  024106 001242  001116  001234  DT2:    .WORD   TSTNUM,SERRPC,PTNBR,SBDADR,SGDDAT,SBDDAT,0
4097  024114 001122  001124  001126
4098  024122 000000
4099  024124 001242  001116  001122  DT5:    .WORD   TSTNUM,SERRPC,SBDADR,SGDDAT,SBDDAT,0
4100  024132 001124  001126  000000
4101  024140 001242  001116  001234  DT6:    .WORD   TSTNUM,SERRPC,PTNBR,0
4102  024146 000000
4103  024150 001242  001116  001234  DT7:    .WORD   TSTNUM,SERRPC,PTNBR,TIME,0
4104  024156 001252  000000
4105  024162 001242  001116  001236  DT13:   .WORD   TSTNUM,SERRPC,SEIZPT,0
4106  024170 000000
4107  024172 001242  001116  001236  DT14:   .WORD   TSTNUM,SERRPC,SEIZPT,PTNBR,SBDADR,SBDDAT,0
4108  024200 001234  001122  001126
4109  024206 000000
4110  024210 001242  001116  000000  DT17:   .WORD   TSTNUM,SERRPC,0
4111  024216 001242  001116  001236  DT24:   .WORD   TSTNUM,SERRPC,SEIZPT,PTNBR,0
4112  024224 001234  000000
4113  024230 001272  000000          DT25:   .WORD   $RMADR,0
```

```
024234  001242  001116  001236  DT30:   .WORD   TSTNUM,$ERRPC,SEIZPT,PTNBR,$BDADR,$GDDAT,$BDDAT,0
024242  001234  001122  001124
024250  001126  000000
024254  001242  001116  001170  DT34:   .WORD   TSTNUM,$ERRPC,$TMP2,$TMP3,0
024262  001172  000000
024266  001242  001116  001236  DT35:   .WORD   TSTNUM,$ERRPC,SEIZPT,PTNBR,0
024274  001234  000000
024300  001242  001116  001236  DT40:   .WORD   TSTNUM,$ERRPC,SEIZPT,OPPRT,0
024306  001240  000000

024312     000     000     001  DF1:    .BYTE   0,0,1,0,0
024315     000     000
024317     000     000     001  DF2:    .BYTE   0,0,1,0,0,0
024322     000     000     000
024325     000     000     000  DF5:    .BYTE   0,0,0,0,0
024330     000     000
024332     000     000     001  DF6:    .BYTE   0,0,1
024335     000     000     001  DF7:    .BYTE   0,0,1,1
024340     001
024341     000     000     001  DF14:   .BYTE   0,0,1,1,0,0
024344     001     000     000
024347     000     000          DF17:   .BYTE   0,0
024351     000                  DF25:   .BYTE   0
024352     000     000     001  DF30:   .BYTE   0,0,1,1,0,0,0
024355     001     000     000
024360     000
024361     000     000     000  DF34:   .BYTE   0,0,0,0
024364     000

        024366                          .EVEN



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

                                .SBTTL  CONSTANTS, TABLES, ETC

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

                                ;TABLE OF TEST STARTING ADDRESSES

024366  002722                  TSTADR: .WORD   TST1+2          ;STARTING ADDRESS OF TEST 1
024370  004322                          .WORD   TST2+2          ;STARTING ADDRESS OF TEST 2
024372  005104                          .WORD   TST3+2          ;STARTING ADDRESS OF TEST 3
024374  005666                          .WORD   TST4+2          ;STARTING ADDRESS OF TEST 4
024376  006604                          .WORD   TST5+2          ;STARTING ADDRESS OF TEST 5
024400  007522                          .WORD   TST6+2          ;STARTING ADDRESS OF TEST 6
024402  010570                          .WORD   TST7+2          ;STARTING ADDRESS OF TEST 7
024404  011704                          .WORD   TST10+2         ;STARTING ADDRESS OF TEST 10

                                ;ATTENTION BIT TABLE

024406     001                  ATABIT: .BYTE   1               ;ATTENTION BIT FOR DRIVE 0
024407     002                          .BYTE   2               ;ATTENTION BIT FOR DRIVE 1
024410     004                          .BYTE   4               ;ATTENTION BIT FOR DRIVE 2
024411     010                          .BYTE   10              ;ATTENTION BIT FOR DRIVE 3
024412     020                          .BYTE   20              ;ATTENTION BIT FOR DRIVE 4
```

M06


```
4170  024413  040                       .BYTE   40          ;ATTENTION BIT FOR DRIVE 5
4171  024414  100                       .BYTE   100         ;ATTENTION BIT FOR DRIVE 6
4172  024415  200                       .BYTE   200         ;ATTENTION BIT FOR DRIVE 7
4173
4174  024416  000010             MAXTN: .WORD   $TN-1       ;MAXIMUM TEST NUMBER
4175
4176          000001                    .END
```

```
ADDRIS  017056          1423    3587#
ADREAR  016643          1350    3561#
AOE   = 001000           811#
ASR1    001232          1008#   1375*
ATA   = 100000           798#   1719    1723    1744    1750    1829    1833    1854    1860    1974    1978    2001    2007
                        2119    2123    2146    2152    2213    2217    2240    2246    2264    2268    2291    2297    2373
                        2567
ATABIT  024406          1375    4165#
ATO   = 000001           825#
AT1   = 000002           826#
AT2   = 000004           827#
AT3   = 000010           828#
AT4   = 000020           829#
AT5   = 000040           830#
AT6   = 000100           831#
AT7   = 000200           832#
A16   = 000400           729#
A17   = 001000           730#
BADNO   017030          1408    3583#
BAI   = 000010           747#
BIT0  = 000001           701#
BIT00 = 000001           691#    701
BIT01 = 000002           690#    700
BIT02 = 000004           689#    699
BIT03 = 000010           688#    698
BIT04 = 000020           687#    697
BIT05 = 000040           686#    696
BIT06 = 000100           685#    695
BIT07 = 000200           684#    694
BIT08 = 000400           683#    693
BIT09 = 001000           682#    692
BIT1  = 000002           700#
BIT10 = 002000           681#   2863
BIT11 = 004000           680#   2829
BIT12 = 010000           679#
BIT13 = 020000           678#   2870
BIT14 = 040000           677#   2811
BIT15 = 100000           676#
BIT2  = 000004           699#
BIT3  = 000010           698#
BIT4  = 000020           697#
BIT5  = 000040           696#
BIT6  = 000100           695#
BIT7  = 000200           694#
BIT8  = 000400           693#
BIT9  = 001000           692#
BPTVEC= 000014           708#
CHANGE  002576          1344    1420#
CHGADR  001270          1023#   1291*   1293*   1420    1422*
CKCLK   013226          1380    1393    2750#
CKCLK1  013276          2750    2759#
CKCLK2  013340          2760    2767#
CKCLK3  013350          2758    2766    2769#
CKERR   001244          1013#   1484*   1497*   1499    1505*   1518*   1520    1529*   1542*   1546*   1559*   1567*   1580*
                        1582*   1595*   1599*   1612*   1614*   1627*   1686*   1699*   1796*   1809*   2369*   2382*   2398*
                        2411*   2413*   2426*   2440*   2453*   2563*   2576*   2592*   2605*   2607*   2620*   2634*   2647*
```

```
CKSWR = 104407          2810    2858    2878    3531#
CLOCK   013360          2754    2763    2774#
CLR   = 000040           749#   1451    1501    1522
CONTUE  017550          2201    2252    3643#
CR    = 000015           616#    998#   3001    3011    3544    3548    3556    3561    3565    3569    3573    3583    3587
                        3600    3605    3615    3625    3634    3643    3650    3654    3781    3985    3993    4011    4026
                        4041    4052    4064    4079
CRLF  = 000200           617#   2972    3011
CYCLED  017617          2350    2469    2544    2661    3650#
CYCLEU  017641          2358    2471    2552    2663    3654#
DCK   = 100000           817#
DDISP = 177570           623#    974    1320
DE1   = 000040           788#
DFF20 = 000002           784#
DF1     024312          1056    1070    1105    1112    1119    1140    1168    1175    1182    1238    4124#
DF14    024341          1133    4133#
DF17    024347          1154    1161    4135#
DF2     024317          1063    1077    1224    1231    4126#
DF25    024351          1196    4136#
DF30    024352          1217    1266    1280    4137#
DF34    024361          1245    4140#
DF5     024325          1084    4128#
DF6     024332          1091    1126    1148    4130#
DF7     024335          1098    1189    1252    1259    1273    4131#
DH1     022511          1054    1068    1103    1110    1117    1138    1166    1173    1180    1236    3952#
DH13    023001          1124    1146    3985#
DH14    023057          1131    3993#
DH17    023177          1152    1159    4008#
DH2     022562          1061    1075    1222    1229    3959#
DH24    023216          1187    4011#
DH25    023322          1194    4024#
DH30    023331          1215    1278    4026#
DH34    023454          1243    4041#
DH35    023552          1250    1257    4052#
DH37    023650          1264    4064#
DH40    023773          1271    4079#
DH5     022636          1082    3967#
DH6     022705          1089    3974#
DH7     022734          1096    3978#
DIGB  = 000004           785#
DISPLA  001142           974#   1320*   1328*   2840*   2862*
DISPRE  000174           923#   1328
DLT   = 100000           759#
DL64  = 000020           787#
DMD   = 000001           821#
DPE   = 000010           863#
DPR   = 000400           791#   1586    1618    1690    1715    1800    1825    1927    1970    2072    2115    2209    2260
                        2373    2567
DRQ   = 004000           848#
DRY   = 000200           790#   1586    1618    1690    1715    1800    1825    1927    1970    2072    2115    2209    2260
                        2373    2567
DSWR  = 177570           622#    973    1319
DTE   = 010000           814#
DTO0  = 000001           839#
DTO1  = 000002           840#
DTO2  = 000004           841#
```

```
DT03  = 000010            842#
DT04  = 000020            843#
DT05  = 000040            844#
DT06  = 000100            845#
DT07  = 000200            846#
DT08  = 000400            847#
DT1     024072           1055    1069    1104    1111    1118    1139    1167    1174    1181    1237    4094#
DT13    024162           1125    1147    4105#
DT14    024172           1132    4107#
DT17    024210           1153    1160    4110#
DT2     024106           1062    1076    1223    1230    4096#
DT24    024216           1188    4111#
DT25    024230           1195    4113#
DT30    024234           1216    1265    1279    4114#
DT34    024254           1244    4117#
DT35    024266           1251    1258    4119#
DT40    024300           1272    4121#
DT5     024124           1083    4099#
DT6     024140           1090    4101#
DT7     024150           1097    4103#
DVA   = 004000            779#
DVC   = 000200            864#
ECH   = 000100            808#
ECI   = 004000            875#
EMTVEC= 000030            711#    1305*   1306*
EM1     017735           1053    3672#
EM10    020326           1102    3725#
EM11    020413           1109    3735#
EM12    020460           1116    3743#
EM13    020527           1123    3751#
EM14    020614           1130    3761#
EM15    020676           1137    3771#
EM16    020762           1144    3781#
EM17    021106           1151    3797#
EM2     020003           1060    3680#
EM20    021171           1158    3807#
EM21    021254           1165    3817#
EM22    021325           1172    3825#
EM23    021423           1179    3837#
EM24    021517           1186    3849#
EM25    021577           1193    3859#
EM3     020024           1067    3684#
EM30    021641           1214    3866#
EM31    021672           1221    3872#
EM32    021740           1228    3880#
EM33    021770           1235    3885#
EM34    022054           1242    3895#
EM35    022123           1249    3903#
EM36    022210           1256    3913#
EM37    022275           1263    3923#
EM4     020103           1074    3693#
EM40    022356           1270    3933#
EM41    022433           1277    3942#
EM5     020125           1081    3698#
EM6     020207           1088    3708#
EM7     020261           1095    3717#
```

```
ENTERA  016613          1345    3556#
ERR     = 040000         797#
ERRVEC= 000004           704#   1317    1318*   1329*   2750*   2751*   2760*   2769*   2816    2817*   2819*   2822*
EXEC    002424          1381    1388#   1470    1666    1776    1889    2034    2182    2341    2535    2701
FER     = 000020         806#
FMT16   = 010000         876#   1681    1791    2390    2488    2584    2680
F0      = 000002         774#
F1      = 000004         775#
F2      = 000010         776#
F3      = 000020         777#
F4      = 000040         778#
GNS     = ****** U       922    2714    2721    3523    3524    3525    3526    3527    3529    3531    3532    3533    3534
                        3535    3536
GO      = 000001         773#
GRV     = 000010         786#
GTSWR   = 104406        1340    3529#
HCE     = 000200         809#
HCI     = 002000         874#
HCRC    = 000400         810#
HT      = 000011         614#   2970    3011
IAE     = 002000         812#
IE      = 000100         727#
ILF     = 000001         802#
ILR     = 000002         803#
IOTVEC= 000020           709#   1303*   1304*
IR      = 000100         750#
IVC     = 010000         867#
KYBCTL  001266          1022#   1396*   1413*   1467    1471*   1663    1667*   1773    1777*   1886    1890*   2031    2035*
                        2179    2183*   2302    2338    2342*   2465    2532    2536*   2699
LBC     = 002000         865#
LBT     = 002000         793#
LF      = 000012         615#    999#   3005    3011    3544    3548    3556    3561    3565    3569    3573    3580    3583
                        3587    3595    3600    3605    3615    3625    3634    3643    3650    3654    3781    3985    3993
                        4011    4026    4041    4052    4064    4079
LSC     = 004000         866#
MAXTN   024416          1406    4174#
MCPE    = 020000         732#
MDPE    = 000400         752#
MOH     = 020000         849#
MOL     = 010000         795#   1586    1618    1690    1715    1800    1825    1927    1970    2072    2115    2209    2260
                        2355    2363    2373    2474    2476    2549    2557    2567    2666    2668
MXF     = 001000         753#
NBA     = 100000         851#
NED     = 010000         756#   1490    1494    1511    1515    2417    2419    2423    2611    2613    2617
NEM     = 004000         755#
NOATA   = 000000        1284#   1744    1750    1854    1860    2001    2007    2146    2152    2240    2246    2291    2297
NOCLOC  016741          1382    3573#
NOSEIZ  001246          1014#
NTRH11  017136          1427    3595#
OFD     = 000200         873#
OPE     = 020000         868#
OPI     = 020000         815#
OPPRT   001240          1011#   1914*   2059*   4121
OR      = 000200         751#
PAR     = 000010         805#
PAT     = 000020         748#
```

```
PGE    = 002000          754#
PGM    = 001000          792#  1571  1573  1577  1603  1605  1609  1690  1715  1800  1825  1927  1970
                        2072  2115  2209  2260
PIP    = 020000          796#
PIRQ   = 177772          621#
PIRQVE= 000240           715#
PORTA    001224         1005#  1347*  1348  1352  1355  1359  1371  1374  1481  1482  1527  1528  1565
                        1566  1633  1673  1674  1706  1707  1716  1735  1737  1743  1826  1845  1847
                        1853  1909  1910  1924  1925  1971  1990  1992  2000  2057  2058  2059  2085
                        2086  2116  2135  2137  2145  2192  2210  2229  2231  2239  2261  2280  2282
                        2290  2348  2349  2434  2435  2437  2472  2473  2590  2591  2632  2633  2677
                        2678
PORTAI   016665         1358  3565#
PORTB    001226         1006#  1352*  1353*  1354*  1357*  1365  1502  1503  1544  1545  1597  1598  1635
                        1720  1731  1732  1749  1783  1784  1816  1817  1830  1841  1842  1859  1912
                        1913  1914  1940  1941  1975  1986  1987  2006  2054  2055  2069  2070  2120
                        2131  2132  2151  2196  2214  2225  2226  2245  2265  2276  2277  2296  2396
                        2397  2438  2439  2485  2486  2542  2543  2628  2629  2631  2664  2665
PORTBI   016713         1364  3569#
PORTC    001230         1007#  1371*  1372*  1373*
PR0    = 000000          638#
PR1    = 000040          639#
PR2    = 000100          640#
PR3    = 000140          641#
PR4    = 000200          642#
PR5    = 000240          643#
PR6    = 000300          644#
PR7    = 000340          645#
PS     = 177776          618#  619  1389*
PSEL   = 002000          731#
PSW    = 177776          619#
PTNBR    001234         1009#  1482*  1503*  1528*  1545*  1566*  1598*  1674*  1707*  1731*  1735*  1743*  1749*
                        1784*  1817*  1841*  1845*  1853*  1859*  1913*  1925*  1941*  1986*  1990*  2000*  2006*
                        2058*  2070*  2086*  2131*  2135*  2145*  2151*  2225*  2229*  2239*  2245*  2276*  2280*
                        2290*  2296*  2349*  2397*  2435*  2439*  2473*  2486*  2543*  2591*  2629*  2633*  2665*
                        2678*  4094  4096  4101  4103  4107  4111  4114  4119
PWRVEC= 000024           710#
RDCHR  = 104410         3340  3532#
RDLIN  = 104411         3413  3533#
RDOCT  = 104412         1346  1402  1428  3534#
RDY    = 000200          728#
RELERR   001250         1015#  1712*  1740*  1822*  1850*  1967*  1995*  2112*  2140*  2206*  2234*  2257*  2285*
RELOK  = 000001         1284#  1741  1851  1867#  1996  2141  2235  2286
RESREG= 104414          3536#
RESVEC= 000010           705#
RMAS   = 000016          903#
RMBA   = 000004          898#
RMCS1  = 000000          896#  1679*  1680*  1708*  1789*  1790*  1818*  1996*  1997*  2141*  2142*  2194*  2195*
                        2198*  2199*  2235*  2236*  2286*  2287*  2388*  2389*  2482*  2483*  2484*  2487*  2489*
                        2582*  2583*  2674*  2675*  2676*  2679*  2681*
RMCS2  = 000010          900#  1451*  1481*  1485  1486  1501*  1502*  1506  1507  1522*  1527*  1544*  1565*
                        1597*  1633*  1635*  1673*  1706*  1716*  1720*  1732*  1737*  1783*  1816*  1826*  1830*
                        1842*  1847*  1909*  1912*  1924*  1940*  1971*  1975*  1987*  1992*  2054*  2057*  2069*
                        2085*  2116*  2120*  2132*  2137*  2192*  2196*  2210*  2214*  2226*  2231*  2261*  2265*
                        2277*  2282*  2348*  2396*  2414  2415  2434*  2438*  2472*  2485*  2542*  2590*  2608
                        2609  2628*  2632*  2664*  2677*
```

```
RMDA   = 000006          899#
RMDB   = 000022          905#
RMDC   = 000034          910#
RMDS1  = 000012          901#  1483  1504  1568  1569  1583  1584  1600  1601  1615  1616  1687  1688
                        1713  1717  1721  1797  1798  1823  1827  1831  1911* 1915  1917  1926  1942
                        1968  1972  1976  2056* 2060  2062  2071  2087  2113  2117  2121  2193* 2197*
                        2207  2211  2215  2258  2262  2266  2355  2363  2370  2371  2399  2400  2428*
                        2436* 2441  2442  2474  2476  2549  2557  2564  2565  2593  2594  2622* 2630*
                        2635  2636  2666  2668
RMDT   = 000026          907#  1530  1531  1547  1548
RMEC1  = 000044          913#
RMEC2  = 000046          914#
RMER1  = 000014          902#
RMER2  = 000042          912#
RMLA   = 000020          904#
RMMR1  = 000024          906#
RMMR2  = 000040          911#
RMOF   = 000032          909#  1681* 1791* 2390* 2488* 2584* 2680*
RMR    = 000004          804#
RMSN   = 000030          908#  1634  1636
RMWC   = 000002          897#  1435
SAVREG= 104413          3535#
SC     = 100000           734#
SC0    = 000100           855#
SC1    = 000200           856#
SC2    = 000400           857#
SC3    = 001000           858#
SC4    = 002000           859#
SEIZPT  001236          1010#  1910* 2055* 2437* 2631* 4105  4107  4111  4114  4119  4121
SKI    = 100000           869#
STACK  = 001100           609#  1301  1392  1476  1672  1782  1895  2040  2188  2347  2541
STAND8  017170          3600#
START   001706           939   1291#
START1  001714           943   1293#
START2  001722          1292   1294#
STKLMT= 177774           620#
SWR     001140           973#  1299  1319* 1321  1327* 1338  1640  2460  2467  2654  2659  2811  2829
                        2863  2870  2875  3192  3231  3278*
SWREG   000176           924#  1327  1338  3192  3231  3254
SWTCHA  017402          2200   2357  3625#
SWTCHB  017465          2251   2551  3634#
SWTCHN  017313          2304   2470  2662  3615#
SW0    = 000001           673#
SW00   = 000001           663#   673
SW01   = 000002           662#   672
SW02   = 000004           661#   671
SW03   = 000010           660#   670
SW04   = 000020           659#   669
SW05   = 000040           658#   668
SW06   = 000100           657#   667
SW07   = 000200           656#   666
SW08   = 000400           655#   665
SW09   = 001000           654#   664  2460  2654
SW1    = 000002           672#
SW10   = 002000           653#
SW11   = 004000           652#
```

```
SW12  = 010000          651#
SW13  = 020000          650#
SW14  = 040000          649#  2467    2659
SW15  = 100000          648#  1640
SW2   = 000004          671#
SW3   = 000010          670#
SW4   = 000020          669#
SW5   = 000040          668#
SW6   = 000100          667#
SW7   = 000200          666#
SW8   = 000400          665#
SW9   = 001000          664#
TAP   = 040000          850#
TBITVE= 000014          706#
TESTNO  017010         1401    3580#
TEST1   002772         1473    1474    1476#
TEST10  011754         2538    2539    2541#
TEST2   004372         1669    1670    1672#
TEST3   005154         1779    1780    1782#
TEST4   005736         1892    1893    1895#
TEST5   006654         2037    2038    2040#
TEST6   007572         2185    2186    2188#
TEST7   010640         2344    2345    2347#
TIME    001252         1016#   1902*   1948    2047*   2093    2774*   4103
TIMEA   001256         1018#   1376*   1896*   1948*   1950    1956
TIMEAP  001260         1019#   1377*   1897*   1951*
TIMEB   001262         1020#   1378*   2041*   2093*   2095    2101
TIMEBP  001264         1021#   1379*   2042*   2096*
TITLE   016510         1333    3544#
TKVEC = 000060          713#   3177*   3178*
TOLER   013412         1949    2094    2784#
TPVEC = 000064          714#
TRAPVE= 000034          712#   1307*   1308*
TRE   = 040000          733#
TRTVEC= 000014          707#
TSTADR  024366         1412    4154#
TSTNUM  001242         1012#   2859*   4094    4096    4099    4101    4103    4105    4107    4110    4111    4114    4117
                       4119    4121
TST1    002720         1405    1466#   4154
TST1AA  002706         1450#   2737
TST10   011702         2495    2531#   4161
TST2    004320         1662#   4155
TST3    005102         1772#   4156
TST4    005664         1885#   4157
TST5    006602         2030#   4158
TST6    007520         2178#   4159
TST7    010566         2337#   4160
TYPDS = 104405         2718    2725    2929    3527#
TYPE  = 104401         1333    1345    1350    1358    1364    1370    1382    1390    1401    1408    1423    1426    1427
                       2200    2201    2251    2252    2304    2350    2357    2358    2469    2470    2471    2490    2544
                       2551    2552    2661    2662    2663    2712    2719    2726    2865    2873    2893    2910    2912
                       2915    2917    2932    2937    2975    3071    3146    3198    3252    3253    3256    3269    3280
                       3299    3347    3353    3358    3362    3367    3368    3370    3373    3377    3442    3444    3523#
TYPOC = 104402         1425    2901    2925    3255    3524#
TYPON = 104404         3526#
TYPOS = 104403         1361    1367    3525#
```

```
UNS     = 040000          816#
UPE     = 020000          757#
U0      = 000001          744#
U1      = 000002          745#
U3      = 000004          746#
VV      = 000100          789#  1690  1719  1723  1744  1750  1800  1829  1833  1854  1860  1927  1970
                         1974  1978  2072  2115  2119  2123  2209  2213  2217  2260  2264  2268
VVSET   = 000001         1284#  1715  1741  1744  1750  1825  1851  1854  1860  1867#  1927  1970  1996
                         1998  2001  2002  2007  2008  2072  2115  2141  2143  2146  2147  2152  2153
                         2209  2235  2240  2241  2246  2247  2260  2286  2291  2292  2297  2298
WATCH   001254           1017#  1903*  1944  2048*  2089  2491*  2492  2682*  2683  2775  2777*  2779*
WCE     = 040000          758#
WCF     = 000040          807#
WLE     = 004000          813#
WRL     = 004000          794#
$AUTOB  001134           970#  1342*  3245  3393
$BDADR  001122           965#  1486*  1487*  1507*  1508*  1531*  1532*  1548*  1549*  1569*  1570*  1584*  1585*
                        1601*  1602*  1616*  1617*  1688*  1689*  1713*  1714*  1798*  1799*  1823*  1824*  1916*
                        1917*  1968*  1969*  2061*  2062*  2113*  2114*  2207*  2208*  2258*  2259*  2371*  2372*
                        2400*  2401*  2415*  2416*  2442*  2443*  2565*  2566*  2594*  2595*  2609*  2610*  2636*
                        2637*  4094  4096  4099  4107  4114
$BDDAT  001126           967#  1485*  1489  1493  1506*  1510  1514  1530*  1534  1538  1547*  1551  1555
                        1568*  1572  1576  1583*  1587  1591  1600*  1604  1608  1615*  1619  1623  1636*
                        1637  1687*  1691  1695  1730*  1736*  1742*  1748*  1797*  1801  1805  1840*  1846*
                        1852*  1858*  1915*  1919  1926*  1930  1985*  1991*  1999*  2005*  2060*  2064  2071*
                        2075  2130*  2136*  2144*  2150*  2224*  2230*  2238*  2244*  2275*  2281*  2289*  2295*
                        2370*  2374  2378  2399*  2403  2407  2414*  2418  2422  2441*  2445  2449  2564*
                        2568  2572  2593*  2597  2601  2608*  2612  2616  2635*  2639  2643  4094  4096
                        4099  4107  4114
$BELL   001202           993#  2865  2885  3198  3387
$CHARC  014344          2977*  2987*  2994  3003*  3008#
$CKSWR  015274          3231#  3531
$CMTAG  001100           953#  1296  1297  1305  1309  1310  1311
$CM1    = 000001         985#   986#
$CM2    = 000002         985#   986#
$CM3    = 000001         983#   985
$CM4    = 000005         986#   987#  988#  989#  990#  991#
$CNTLG  016136          3252  3388#
$CNTLU  016131          3269  3362  3387#
$CRLF   001207           995#  1370  1390  1426  2490  2726  2873  2885  2893  2912  2917  2937  2976
                        3011  3280  3367  3387  3447
$DBLK   015012          3112  3146  3154#
$DOAGN  013216          2708  2729  2735#
$DTBL   015002          3115  3150#
$ENDAD  013206           932  2731#  2880
$ENDCT  013052          1309  2710#
$ENULL  013222          2738#
$EOP    013006          2686  2698#
$EOPCT  013044          1309*  2707#  2711
$ERFLG  001103           956#  2458  2462*  2652  2656*  2802  2825  2827*  2844  2860*  2885
$ERMAX  001115           962#  1312*  1398*  2844
$ERROR  013622          1305  2857#
$ERRPC  001116           963#  2867*  2868*  2869  2885  2899  4094  4096  4099  4101  4103  4105  4107
                        4110  4111  4114  4117  4119  4121
$ERRTB  001276          1048#  2907
$ERRTY  013754          2872  2892#
```

```
$ERTTL  001112          960#  2723   2727*  2866*  2885
$ESCAP  001200          992#  1311*
$FILLC  001156          981#  2980   3011
$FILLS  001155          980#  3011
$GDADR  001120          964#
$GDDAT  001124          966#  1488*  1491   1495*  1509*  1512   1516*  1533*  1536   1540*  1550*  1553   1557*
                       1571*  1574   1578*  1586*  1589   1593*  1603*  1606   1610*  1618*  1621   1625*  1634*
                       1637   1690*  1693   1697*  1715*  1745   1751   1800*  1803   1807*  1825*  1855   1861
                       1918*  1919   1927*  1928   1932   1970*  2002   2008   2063*  2064   2072*  2073   2077
                       2115*  2147   2153   2209*  2241   2247   2260*  2292   2298   2373*  2376   2380*  2402*
                       2405   2409*  2417*  2420   2424*  2444*  2447   2451*  2567*  2570   2574*  2596*  2599
                       2603*  2611*  2614   2618*  2638*  2641   2645*  4096   4099   4114
$GET42  013176          2728#
$GTSWR  015364          3253#  3529
$HD   = 000000           593
$HIOCT  016324          3435*  3446#
$ICNT   001104           957#  1414*  2833*  2834   2836*  2843
$INTAG  001135           971#  3250*  3281   3393
$ITEMB  001114           961#  2869*  2885   2896
$LF     001210           996#  2885   3011   3377   3387   3447
$LKCSB  001214          1001#  2756*
$LKCSR  001212          1000#  2752   2757*
$LKS    001220          1003#  2761   2765*
$LLVEC  001222          1004#  2762
$LPADR  001106           958#  1313*  1399*  1473*  1669*  1779*  1892*  2037*  2185*  2344*  2538*  2839*  2841
                       2843
$LPERR  001110           959#  1314*  1400*  1474*  1670*  1780*  1893*  2038*  2186*  2345*  2464   2539*  2658
$LPVEC  001216          1002#  2753
$MAIL = ****** U        1331   1338   2839   2875   2964
$MNEW   016154          3256   3391#
$MSWR   016143          3253   3389#
$MXCNT  013620          2837   2843#
$NULL   001154           979#  2982   3011
$NWTST= 000001          1453#  1455   1647#  1649   1757#  1759   1870#  1872   2015#  2017   2161#  2163   2309#
                       2311   2497#  2499
$OCNT   014572          3043*  3072*  3085#
$OMODE  014574          3038*  3042*  3047   3050*  3061*  3087#
$OVER   013604          2812   2823   2835   2840#
$PASS   001100           954#  1397*  2704*  2705*  2716   2738   2831   2844
$QUES   001206           994#  2885   3011   3299   3370   3387   3444   3447
$RDCHR  015576          3312#  3532
$RDDEC= ****** U        3535
$RDLIN  015666          3335#  3533
$RDOCT  016166          3408#  3534
$RDSZ = 000007          3328#
$REGAD  001160           983#
$REGO   001162           985#
$RESRE  016364          3480#  3536
$RMADR  001272          1031#  1391   1424   1431*  1434   1450   4113
$RMVEC  001274          1032#
$RTNAD  013220          2737#
$R2A  = ****** U        3537
$SAVRE  016326          3464#  3535
$SCOPE  013440          1303   2809#
$SETUP= 000127          1284#  1302   1303   1305   1307   1309   1310   1311   1313   1335   2702   2810   2858
                       2878   2880   3190   3195   3196   3226   3393
```

```
$STUP = 177777          1284#
$SVLAD  013574          2820    2838#
$SVPC = 000200           930#    935
$SWR  = 166000           582#    593     598     599     600     601     602     603     604     991     992     993    1310
                        1311    1313    1314    1475    1671    1781    1894    2039    2187    2346    2540    2695    2703
                        2730    2736    2738    2797#   2803    2804    2805    2806    2811    2823    2825    2826    2827
                        2828    2829    2840    2843    2850    2851    2852    2853    2854    2863    2870    2875    2879
                        2885
$SWRMK= 000000          2806
$TIMES  001176           991#   1310*   1475*   1671*   1781*   1894*   2039*   2187*   2346*   2463*   2540*   2657*   2703*
                        2828*   2834    2837*   2843
$TKB    001146           976#   3158    3179    3188    3207    3235    3262
$TKCNT  015022          3159#   3174*   3196    3213*   3319    3321*
$TKINT  015032          1334    1395    3174#   3248
$TKQEM= 015031          3163#   3221    3324
$TKQIN  015024          3160#   3175*   3176    3219*   3220*   3221    3223*
$TKQOU  015026          3161#   3176*   3322    3323*   3324    3326*
$TKQSR  015030          3162#   3175    3223    3326
$TKS    001144           975#   3158    3180*   3203*   3205    3211*   3233    3249*   3259    3283*
$TKSRV  015102          3177    3188#
$TMPO   001164           986#   1489*   1490*   1491    1510*   1511*   1512    1534*   1535*   1536    1551*   1552*   1553
                        1572*   1573*   1574    1587*   1588*   1589    1604*   1605*   1606    1619*   1620*   1621    1691*
                        1692*   1693    1718*   1719*   1724    1726    1733    1801*   1802*   1803    1828*   1829*   1834
                        1836    1843    1930*   1931*   1932    1973*   1974*   1979    1981    1988    2075*   2076*   2077
                        2118*   2119*   2124    2126    2133    2212*   2213*   2218    2220    2227    2263*   2264*   2269
                        2271    2278    2374*   2375*   2376    2403*   2404*   2405    2418*   2419*   2420    2445*   2446*
                        2447    2568*   2569*   2570    2597*   2598*   2599    2612*   2613*   2614    2639*   2640*   2641
$TMP1   001166           987#   1722*   1723*   1724    1738    1832*   1833*   1834    1848    1928*   1929*   1931    1977*
                        1978*   1979    1993    2073*   2074*   2076    2122*   2123*   2124    2138    2216*   2217*   2218
                        2232    2267*   2268*   2269    2283
$TMP2   001170           988#   1717*   1718    1730    1742    1744*   1745    1827*   1828    1840    1852    1854*   1855
                        1972*   1973    1985    1999    2001*   2002    2117*   2118    2130    2144    2146*   2147    2211*
                        2212    2224    2238    2240*   2241    2262*   2263    2275    2289    2291*   2292    4117
$TMP3   001172           989#   1721*   1722    1736    1748    1750*   1751    1831*   1832    1846    1858    1860*   1861
                        1976*   1977    1991    2005    2007*   2008    2121*   2122    2136    2150    2152*   2153    2215*
                        2216    2230    2244    2246*   2247    2266*   2267    2281    2295    2297*   2298    4117
$TMP4   001174           990#   1493*   1494*   1495    1514*   1515*   1516    1538*   1539*   1540    1555*   1556*   1557
                        1576*   1577*   1578    1591*   1592*   1593    1608*   1609*   1610    1623*   1624*   1625    1695*
                        1696*   1697    1805*   1806*   1807    2378*   2379*   2380    2407*   2408*   2409    2422*   2423*
                        2424    2449*   2450*   2451    2572*   2573*   2574    2601*   2602*   2603    2616*   2617*   2618
                        2643*   2644*   2645
$TN   = 000011           582#    593    1453    1466    1475#   1476    1647    1662    1671#   1672    1757    1772    1781#
                        1782    1870    1885    1894#   1895    2015    2030    2039#   2040    2161    2178    2187#   2188
                        2309    2337    2346#   2347    2495    2497    2531    2540#   2541    4174
$TPB    001152           978#   3000*   3011
$TPFLG  001157           982#   2958    3011
$TPS    001150           977#   2998    3011
$TRAP   016422          1307    3500#
$TRAP2  016444          3511#   3522
$TRP  = 000015          3515#   3524#   3525#   3526#   3527#   3528#   3529    3530#   3531    3532#   3533#   3534#   3535#
                        3536#   3537#
$TRPAD  016456          3505    3522#
$TSTNM  001102           955#   1472*   1668*   1778*   1891*   2036*   2184*   2343*   2537*   2702*   2802    2838*   2840
                        2844    2859    2862    2885
$TTYIN  016122          3337    3338    3350    3368    3382    3386#
$TYPBN= ****** U        3528
```

```
$TYPDS  014576            3100#  3527
$TYPE   014130            2958#  3515   3523
$TYPEC  014300            2979   2986   2993   2998#  2999   3285
$TYPEX  014346            3004   3006   3009#
$TYPOC  014374            3041#  3524
$TYPON  014410            3040   3043#  3526
$TYPOS  014350            3036#  3525
$XTSTR  013452            2814#
$SGET4= 000000            2730#
$OFILL  014573            3037*  3041*  3051   3086#
$40CAT= ****** U          2811   2872
.       = 024420           918#   922#   930    931#   933#   935#   952#   997   1300  1313  1314  1399  1400
                          1500   1521   2700   2715#  2738   2739#  2843   2844  2885  2940# 3011  3154# 3158
                          3162#  3163   3164#  3386#  3387   3393#  3447   4143#
```

```
AA          1868#  2013
BB          1645#  1646  1756
CHECK        581#  1484  1505  1529  1546  1567  1582  1599  1614  1686  1796  2369  2398  2413  2440
            2563   2592  2607  2634
CLRATA       581#  2189
COMMEN       716#
ENDCOM       716#
ERROR        610#  1437  1496  1517  1541  1558  1579  1594  1611  1626  1639  1698  1728  1741  1747
            1753   1808  1838  1851  1857  1863  1921  1934  1946  1958  1983  1998  2004  2010  2066
            2079   2091  2103  2128  2143  2149  2155  2222  2237  2243  2249  2273  2288  2294  2300
            2381   2410  2425  2452  2575  2604  2619  2646
ESCAPE       716#
FF          2307#  2308  2496
GETPRI       716#
GETSWR       582#   716#  1335
HEAD         581#  1466  1662  1772  1885  2030  2178  2337  2531
MORETA       945#   998
MSG         1449#  1455  1646#  1649  1756#  1759  1868#  1872  2013#  2017  2160#  2163  2308#  2311  2496#
            2499
MULT         716#
NEUTRA       581#  1709  1819  1963  2108  2203  2254
NEWTST       716#  1453  1647  1757  1870  2015  2161  2309  2497
POP          716#  3141  3436  3485
PUSH         716#  3100  3410  3465
RELEAS       581#  1703  1813
REPORT       716#
SCOPE        611#  1643  1755  1865  2012  2157  2305  2494  2685
SEIZE        581#  1906  2051
SELECT       581#  1481  1502  1527  1544  1565  1597  1673  1706  1783  1816  1912  1923  1940  2057
            2068   2085  2348  2396  2434  2438  2472  2485  2542  2590  2628  2632  2664  2677
SETATA       581#
SETPRI       716#  3315
SETTRA      3515#  3524  3525  3526  3527  3529  3531  3532  3533  3534  3535  3536
SETUP        716#  1295
SKIP         716#  2495
SLASH        716#
SPACE        716#
STARS        716#   717   719   723   765   769   890   894   928   948   997  1025  1029  1285  1289
            1443   1447  1453  1465  1478  1524  1562  1630  1647  1661  1676  1683  1702  1757  1771
            1786   1793  1812  1870  1884  1899  1905  1937  1953  1960  2015  2029  2044  2050  2082
            2098   2105  2161  2177  2309  2336  2352  2360  2366  2385  2392  2431  2479  2497  2530
            2546   2554  2560  2579  2586  2625  2671  2690  2741  2745  2793  2799  2846  2887  2943
            3013   3090  3157  3226  3241  3304  3328  3396  3449  3494  3538  3542  3666  3670  4146
            4150
SWITCH       581#
SWRSU        716#  1315#
TESTAG       581#  1476  1672  1782  1895  2040  2188  2347  2541
TIMER        581#  1898  2043
TRMTRP      3515#
TYPBIN       716#
TYPDEC       716#  2716  2723  2927
TYPNAM       716#
TYPNUM       716#
TYPOCS       716#  1359  1365
TYPOCT       716#  2899  2924  3254
TYPTXT       716#  2712  2719
```

```
USEREQ   2688#   2699
$$CHRE    946#    985
$$CMTM    946#    986    987    988    989    990
$$ESCA    716#
$$NEMT    716#   1453   1647   1757   1870   2015   2161   2309   2497
$$SET    3515#   3524   3525   3526   3527   3529   3531   3532   3533   3534   3535   3536
$$SKIP    716#   2495
.EQUAT    582#    606
.HEADE    582#    583
.SETUP    582#   1284
.SWRHI    582#    594
.SWRLO    582#    604#
.$ACTI    582#    926
.$CATC    582#    916
.$CMTA    582#    946
.$EOP     582#   2688
.$ERRO    582#   2844
.$ERRT    582#   2885
.$RDOC    582#   3394
.$READ    582#   3155
.$SAVE    582#   3447
.$SCOP    582#   2797
.$TRAP    582#   3492
.$TYPD    582#   3088
.$TYPE    582#   2941
.$TYPO    582#   3011

. ABS.  024420     000

 ERRORS DETECTED:  0

 DSKZ:DZRMHA.BIN,DSKZ:DZRMHA.SEQ/DOC/SOL/CRF=DSKM:DZRMHA.P11
 RUN-TIME: 20 13 1 SECONDS
 RUN-TIME RATIO: 177/35=4.9
 CORE USED:  28K  (55 PAGES)

 DOCUMENT PAGES:  91

EOF1DZRMHASEQ              00010000      770804            PDP10 411
```