RP04

DUAL CONTROLLER II

MD-11-DZRPQ-A

EP-DZRPQ-A-DL-A

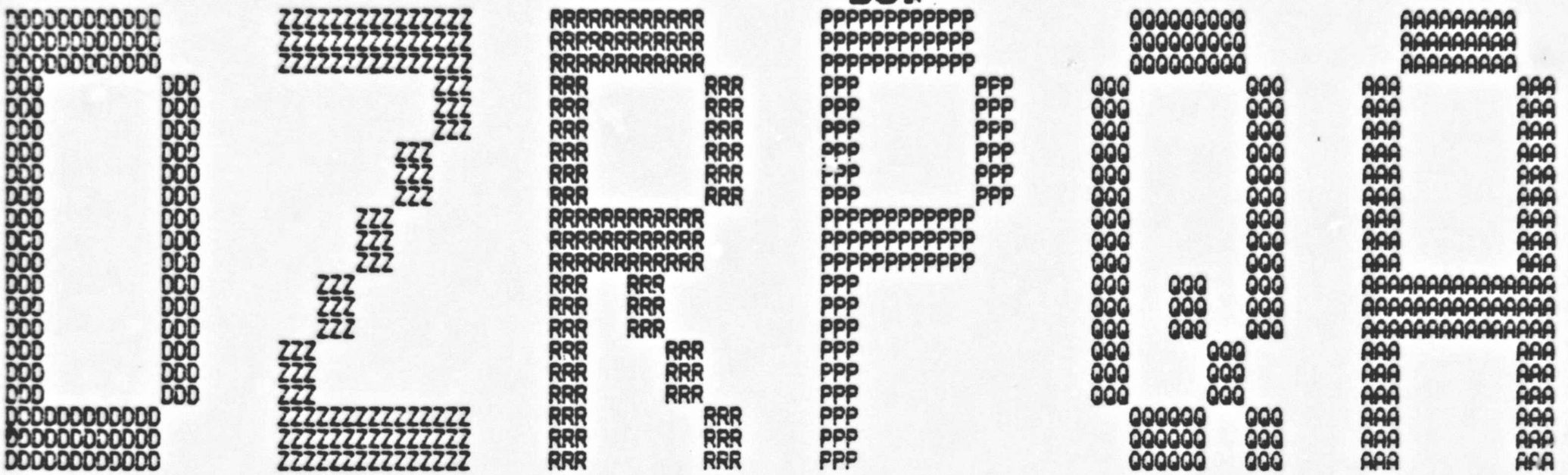
MAR 1976

FICHE 1 OF 1

digital

Made In U.S.A.

BOI




```
LPTSPL Version 101(2111) Running on MTA26D
 *START* User WELLINGHAM,A[1,2]  Job  DZRPQA Seq. 2196 Date 14-Feb-76 11:21:11 Monitor     RZ231D KL1D SYS#1026 *START*
Request created: 14-Feb-76 11:03:01
File: DSKN:DZRPQA.SEQ<055>[400,3722] Created: 13-Feb-76 18:36:23 Printed: 14-Feb-76 11:21:11
QUEUE Switches:  /FILE:ASCII /COPIES:1 /SPACING:1 /LIMIT:250 /FORMS:NORMAL
```

IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DZRPQ-A-D

PRODUCT NAME: RP04 DUAL CONTROLLER LOGIC TEST - PART 2

DATE CREATED: DECEMBER 21, 1974

MAINTAINER: DIAGNOSTIC ENGINEERING

AUTHOR: C. HESS

CONTENTS

1. ABSTRACT

THE RP04 DUAL CONTROLLER OPTION LOGIC TEST PERFORMS A SERIES OF TESTS WHICH VERIFY THAT THE RP04 DUAL CONTROLLER LOGIC IS FUNCTIONING PROPERLY. ONLY THE CONTROL LOGIC IS TESTED BY THIS PROGRAM; DATA HANDLING IN THE DUAL CONTROLLER MODE IS NOT TESTED BY THIS PROGRAM.

BOTH PORTS OF THE RP04 ARE CABLED TO THE SAME MASSBUS BY A SPECIAL ADAPTER CABLE. THIS ARRANGEMENT ALLOWS THE DUAL CONTROLLER LOGIC TO BE TESTED FROM ONE PDP-11/RH11.

THIS PROGRAM IS THE SECOND PART OF THE RP04 DUAL CONTROLLER OPTION LOGIC TEST. ONLY THE CONTROL LOGIC ASSOCIATED WITH THE UNLOAD COMMAND AND 'CONTROLLER SELECT' SWITCH IS TEST BY THIS PROGRAM.

2. REQUIREMENTS

2.1 EQUIPMENT

PDP-11 PROCESSOR
16K OF MEMORY
KW11-L OR KW11-P CLOCK
TELETYPE
RH11 WITH AN RP04
RP04 DUAL CONTROLLER OPTION TEST CABLE

2.2 PRELIMINARY PROGRAMS

A. RP04 DISKLESS CONTROLLER TEST
   PART 1 (MAINDEC-11-DZRPS)
   PART 2 (MAINDEC-11-DZRPT)

B. RP04 FUNCTIONAL CONTROLLER TEST
   PART 1 (MAINDEC-11-DZRPU)
   PART 2 (MAINDEC-11-DZRPV)

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

C. RP04 DUAL CONTROLLER LOGIC TEST
   PART 1 (MAINDEC-11-DZRPP)

2.3 OTHER PROGRAMS

A. DYNAMIC OPERATION OF THE DUAL CONTROLLER OPTION IS TESTED BY THE RP04 MULTIDRIVE EXERCISER (MAINDEC-11-DZRPN-B). NOTE THAT THE RP04 EXERCISER MUST BE PROGRAM REVISION 'B' OR LATER. REVISION 'A' OF THE RP04 EXERCISER DOES NOT SUPPORT DUAL CONTROLLER OPERATION.

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.

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 DRIVE ADDRESS OF THE RP04 TO BE TESTED.

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

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

4.2 UNIBUS & VECTOR ADDRESSES

THE PROGRAM ASSURES THE FOLLOWING UNIBUS AND VECTOR ADDRESSES. THESE ADDRESSES MAY BE CHANGED PRIOR TO INITIATING A PROGRAM START AT ANY OF THE STARTING LOCATIONS.

MEMORY LOCATION	CONTENTS	FUNCTION
1136	177560	TTY KEYBOARD STATUS REG
1140	177562	TTY KEYBOARD BUFFER REG
1142	177564	TTY PRINTER STATUS REG
1144	177566	TTY PRINTER BUFFER REG
1204	172540	KW11-P STATUS REG
1206	172542	KW11-L COUNTER BUFFER
1210	104	KW11-P VECTOR ADDRESS
1212	177546	KW11-L STATUS REGISTER
1214	100	KW11-L VECTOR ADDRESS
1262	176700	RH11/RP04 ADDRESS
1264	254	RH11 INTERRUPT VECTOR ADDRESS

4.3 OPERATOR ACTION

A. CONNECT THE DUAL CONTROLLER TEST CABLE BETWEEN BUS A & BUS B ON THE RP04 BEING TESTED. (SEE SECTION 5.3)
B. LOAD THE PROGRAM INTO MEMORY IN THE PROPER PROCESSOR.
C. SWITCH THE 'CONTROLLER SELECT' SWITCH ON THE RP04 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.
E. PRESS START.
F. ENTER THE DRIVE NUMBER. (THIS MUST BE THE NUMBER DISPLAYED IN THE DRIVE NUMBER LED ON THE CONTROL PANEL.)
G. ENTER THE NUMBER OF THE TEST TO BE RUN.
H. THE PROGRAM CAN BE STOPPED AT ANY TIME AND RESTARTED

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 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 UNTIL IT IS HALTED.

THE 'RUBOUT KEY' (RO) CAN BE USED TO DELETE THE LAST CHARACTER ENTERED. SUCCESSIVE STUCKING AT 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 THE ENTIRE ENTRY BY TYPING A 'CONTROL U' (↑U).

5.3 TEST CABLE CONNECTION

TO TEST THE RP04 DUAL CONTROLLER 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 RP04 BEING TESTED AND IS CONSTRUCTED SO THAT BIT 0 OF THE MASSBUS UNIT SELECT LINES IS COMPLEMENTED.

WITH THE TEST CABLE CONNECTED TO THE RP04 UNDER TEST, THE DRIVE APPEARS AS TWO UNITS ON THE MASSBUS: EACH PORT OF THE RP04 WILL RESPOND TO A DIFFERENT MASSBUS ADDRESS. THE ADDRESS OF EACH PORT WILL DEPEND UPON THE DRIVE'S ADDRESS (THE ADDRESS SELECTED BY THE SWITCHES ON THE 'DP' BOARD - MODULE M7775.)

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 DEVICE ON THE SYSTEM (RP04 OR NON-RP04 DEVICE) WHICH           *
* HAS ANY ADDRESS IN CONFLICT WITH EITHER OF THE TEST                *
* ADDRESSES MUST BE TURNED OFF.                                      *
**********************************************************************
```

THE TEST CABLE CONNECTION TO THE RP04 UNDER TEST WILL DEPEND ON WHICH PROCESSOR/RH11 IS TO TEST THE RP04. IF THE RP04 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 RP04 IS TO BE TESTED BY PROCESSOR ON PORT B, THE TEST CABLE IS CONNECTED FROM 'BUS B OUT' TO 'BUS A IN'.

WHEN THE DUAL PORT TEST CABLE IS CONNTECTED, THE ATTENTION BITS FOR PORTS A & B ARE ASSERTED IN THE SAME BIT POSITION WHEN 'RHAS' (ATTENTION SUMMARY REGISTER) IS READ. THE ATTENTION BIT POSITION IS DETERMINED BY THE SWITCH ON THE 'DP' BOARD (M7775). 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 'RHDS1' (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 RP04 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 CONTROLLER OPTION.

7.3 EXECUTION TIME

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

7.4 STACK POINTER

THE STACK IS INITIALLY SET TO 1100 AND EXTENDS DOWNWARD IN MEMORY.

7.5 SUBROUTINE CALLS

THE SUBROUTINE CALLS USED BY THE PROGRAM ARE:

A. 'SCOPE' (IOT INSTRUCTION). THIS CALL IS PLACED BETWEEN EACH TEST IN THE INSTRUCTION. THIS ROUTINE ESTABLISHES THE ITERATION COUNT AND THE LOOP ON TEST AND LOOP ON ERROR ADDRESSES.

B. 'ERROR' (EMT INSTRUCTION). THIS CALL IS USED TO REPORT ALL ERRORS. THE CALL IS FOLLOWED BY A NUMBER WHICH IDENTIFIES THE ERROR MESSAGE WHICH WILL BE TYPED.

THE TRAP INSTRUCTION IS USED FOR THE FOLLOWING SUBROUTINE CALLS:

TYPE - TTY TYPEOUT ROUTINE
TYPOC - TYPE OCTAL NUMBER (WITH LEADING ZERO)
TYPOS - TYPE OCTAL NUMBER (NO LEADING ZEROS)
TYPON - TYPE OCTAL NUMBER PER LAST CALL
TYPDS - TYPE DECIMAL NUMBER WITH SIGN
RDCHR - READ CHARACTER FROM TTY KEYBOARD
RDLIN - READ A LINE FROM THE TTY KEYBOARD.
RDOCT - READ AN OCTAL NUMBER FROM THE TTY KEYBOARD
SAVREG - ROUTINE TO SAVE R0-R5
RESREG - ROUTINE TO RESTORE R0-R5

7.6 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 - 14 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 - 14 ARE RUN.

7.7 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 RP04 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

7.9 TEST ITERATION

EACH TEST IS PERFORMED ONLY ONCE ON EACH PASS OF THE PROGRAM.

TESTS PERFORMED IN THE SINGLE TEST MODE CAN BE ITERATED AS MANY TIMES AS DESIRED BY SETTING SW<14>.

8. TEST DESCRIPTIONS

8.1 METHOD USED TO VERIFY THAT THE DRIVE IS IN NEUTRAL

THE PROGRAM DETERMINES IF AN RP04 IS IN NEUTRAL BY CHECKING THE CONTENTS OF THE DRIVE STATUS REGISTER (RHDS1) 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 RHDS1, 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 (RHDS1) THROUGH THE SEIZING PORT AND VERIFING THAT CORRECT STATUS IS SEEN. WHEN RHDS1 IS READ THROUGH THE OPPOSITE PORT, ZEROS SHOULD BE SEEN. IF BOTH CONDITIONS EXIST, (I.E., CORRECT STATUS THROUGH THE SEIZING PORT AND ZEROS THROUGH THE OPPOSITE PORT), THE PROGRAM CONCLUDES THAT THE DRIVE HAS BEEN SEIZED BY THE SPECIFIED PORT.

8.3 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 RP04, THAT THE DRIVE IS ONLINE (RHDS1 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.

8.4 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.

8.5 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.

8.6 TEST 4 - MEASURE THE TIMEOUT ONE-SHOT THROUGH PORT 'A'

MEASURE THE TIMEOUT ONE-SHOT VALUE THROUGH PORT 'A'

A. WRITE 0'S INTO RHDS1 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 OCCURED AND THAT THE DRIVE RETURNS TO NEUTRAL.

8.7 TEST 5 - MEASURE THE TIMEOUT ONE-SHOT THROUGH PORT 'B'

MEASURE THE TIMEOUT ONE-SHOT VALUE THROUGH PORT 'B'

A. WRITE 0'S INTO RHDS1 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 OCCURED AND THAT THE DRIVE RETURNS TO NEUTRAL.

8.8 TEST 6 - TEST UNLOAD COMMAND THROUGH PORT 'A'

VERIFY THAT THE UNLOAD COMMAND FUNCTIONS PROPERLY AND THAT A PORT TIMEOUT WILL NOT OCCUR WHILE THE 'GO' BIT IS SET.

A. ISSUE AN UNLOAD COMMAND THROUGH PORT 'A'; VERIFY THAT THE DRIVE IS SEIZED.

B. WAIT THE MEASURED TIMEOUT INTERVAL + 25%; VERIFY THAT THE DRIVE

DOES NOT TIME OUT. VERIFY THAT THE 'GO' BIT IS STILL SET AND THAT 'DRY' AND 'PIP' ARE NOT SET.

C. REQUEST THAT THE OPERATOR PRESS THE 'STANDBY' BUTTON ON THE DRIVE.

D. WHEN THE DRIVE CYCLES UP, VERIFY THAT THE DRIVE IS STILL SEIZED BY PORT A, THAT THE 'VV' BIT FOR PORT A IS RESET, THAT THE ATTENTION BIT FOR PORT 'A' IS SET, AND THAT THE ATTENTION BIT FOR PORT 'B' IS NOT SET.

E. WAIT FOR THE PORT TIMEOUT TO RELEASE THE DRIVE. WHEN THE TIMEOUT OCCURS, VERIFY THAT THE DRIVE RETURNED TO NEUTRAL, THAT THE ATTENTION BIT FOR PORT A IS STILL SET, AND THAT THE ATTENTION BIT FOR PORT 'B' IS NOT SET.

F. VERIFY THAT THE 'VV' BIT FOR PORT 'B' IS NOT SET.

G. ISSUE A PACK ACKNOWLEDGE INSTRUCTION THROUGH BOTH PORTS.

8.9 TEST 7 - TEST UNLOAD COMMAND THROUGH PORT 'B'

VERIFY THAT THE UNLOAD COMMAND FUNCTIONS PROPERLY AND THAT A PORT TIMEOUT WILL NOT OCCUR WHILE THE 'GO' BIT IS SET.

A. ISSUE AN UNLOAD COMMAND THROUGH PORT 'B'; VERIFY THAT THE DRIVE IS SEIZED.

B. WAIT THE MEASURED TIMEOUT INTERVAL + 25%; VERIFY THAT THE DRIVE DOES NOT TIME OUT. VERIFY THAT THE 'GO' BIT IS STILL SET AND THAT 'DRY' AND 'PIP' ARE NOT SET.

C. REQUEST THAT THE OPERATOR PRESS THE 'STANDBY' BUTTON ON THE DRIVE.

D. WHEN THE DRIVE CYCLES UP, VERIFY THAT THE DRIVE IS STILL SEIZED BY PORT B, THAT THE 'VV' BIT FOR PORT B IS RESET, THAT THE ATTENTION BIT FOR PORT 'B' IS SET, AND THAT THE ATTENTION BIT FOR PORT 'A' IS NOT SET.

E. WAIT FOR THE PORT TIMEOUT TO RELEASE THE DRIVE. WHEN THE TIMEOUT OCCURS, VERIFY THAT THE DRIVE RETURNED TO NEUTRAL, THAT THE ATTENTION BIT FOR PORT 'B' IS STILL SET, AND THAT THE ATTENTION BIT FOR PORT 'A' IS NOT SET.

F. VERIFY THAT THE 'VV' BIT FOR PORT 'A' IS NOT SET.

G. ISSUE A PACK ACKNOWLEDGE INSTRUCTION THROUGH BOTH PORTS.

8.10 TEST 10 - TEST THE CONTROLLER SELECT SWITCH THROUGH PORT 'A' DURING UNLOAD

VERIFY THAT THE 'CONTROLLER SELECT' SWITCH IS INHIBITED WHEN THE RP04 IS CYCLED DOWN BY AN UNLOAD COMMAND.

A. ISSUE AN UNLOAD COMMAND THROUGH PORT 'A'.

B. REQUEST THAT THE OPERATOR SWITCH THE 'CONTROLLER SELECT' SWITCH TO 'A' AND THEN PRESS THE 'STANDBY' BUTTON.

C. WAIT FOR 'MOL' TO SET BY MONITORING RHDS1 THROUGH PORT 'A'. WHEN THE DRIVE HAS CYCLED UP ('MOL' HAS SET), ISSUE A RELEASE COMMAND THROUGH PORT 'A'.

D. VERIFY THAT THE DRIVE CAN BE ACCESSED BY BOTH PORTS AND THAT THE DRIVE IS STILL IN NEUTRAL.

E. REQUEST THAT THE OPERATOR RETURN THE 'CONTROLLER SELECT' SWITCH TO 'A/B'.

8.11 TEST 11 - TEST 'CONTROLLER SELECT' SWITCH THROUGH PORT 'B' DURING UNLOAD

VERIFY THAT THE 'CONTROLLER SELECT' SWITCH IS INHIBITED WHEN THE RP04 IS CYCLED DOWN BY AN UNLOAD COMMAND.

A. ISSUE AN UNLOAD COMMAND THROUGH PORT 'B'.

B. REQUEST THAT THE OPERATOR SWITCH THE 'CONTROLLER SELECT' SWITCH TO 'B' AND THEN PRESS THE 'STANDBY' BUTTON.

C. WAIT FOR 'MOL' TO SET BY MONITORING RHDS1 THROUGH PORT 'B'. WHEN THE DRIVE HAS CYCLED UP ('MOL' HAS SET), ISSUE A RELEASE COMMAND THROUGH PORT 'B'.

D. VERIFY THAT THE DRIVE CAN BE ACCESSED BY BOTH PORTS AND THAT THE DRIVE IS STILL IN NEUTRAL.

E. REQUEST THAT THE OPERATOR RETURN THE 'CONTROLLER SELECT' SWITCH TO 'A/B'.

8.12 TEST 12 - 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 RHDS1, 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 RHDS1, AS READ THROUGH BOTH PORTS, ARE CORRECT.

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

8.13 TEST 13 - 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 RHDS1, 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' DOES NOT SET WHEN ATEMPTING TO ACCESS THE DRIVE THROUGH PORT 'B'. ATTEMPT TO SET PORT REQUEST BY WRITING 0'S INTO RHDS1 THROUGH PORT 'B'.

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

8.14 TEST 14 - 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 RHDS1, 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' DOES NOT SET WHEN ATEMPTING TO ACCESS THE DRIVE THROUGH PORT 'A'. ATTEMPT TO SET PORT REQUEST BY WRITING 0'S INTO RHDS1 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.

9. PROGRAM LISTING

```
                                .TITLE  MD-11-DZRPQ-A, RP04 DUAL CONTROLLER LOGIC TEST - PART 2
                                ;*COPYRIGHT (C) 1974
                                ;*DIGITAL EQUIPMENT CORP.
                                ;*MAYNARD, MASS. 01754
                                ;*
                                ;*PROGRAM BY C. HESS
                                ;*
                                ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                                ;*PACKAGE (MAINDEC-11-DZQAC-A2).
                                ;*


                                .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
177776                          PS=     177776          ;;PROCESSOR STATUS WORD
                                .EQUIV  PS,PSW
177774                          STKLMT= 177774          ;;STACK LIMIT REGISTER
177772                          PIRQ=   177772          ;;PROGRAM INTERRUPT REQUEST REGISTER
177570                          SWR=    177570          ;;SWITCH REGISTER
177570                          DISPLAY=SWR

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

                                ;*PRIORITY LEVEL DEFINITIONS
```

```
(1)  000000                  PR0=    0              ;;PRIORITY LEVEL 0
(1)  000040                  PR1=    40             ;;PRIORITY LEVEL 1
(1)  000100                  PR2=    100            ;;PRIORITY LEVEL 2
(1)  000140                  PR3=    140            ;;PRIORITY LEVEL 3
(1)  000200                  PR4=    200            ;;PRIORITY LEVEL 4
(1)  000240                  PR5=    240            ;;PRIORITY LEVEL 5
(1)  000300                  RP6=    300            ;;PRIORITY LEVEL 6
(1)  000340                  PR7=    340            ;;PRIORITY LEVEL 7
(1)
(1)                          ;*"SWITCH REGISTER" SWITCH DEFINITIONS
(1)  100000                  SW15=   100000
(1)  040000                  SW14=   40000
(1)  020000                  SW13=   20000
(1)  010000                  SW12=   10000
(1)  004000                  SW11=   4000
(1)  002000                  SW10=   2000
(1)  001000                  SW09=   1000
(1)  000400                  SW08=   400
(1)  000200                  SW07=   200
(1)  000100                  SW06=   100
(1)  000040                  SW05=   40
(1)  000020                  SW04=   20
(1)  000010                  SW03=   10
(1)  000004                  SW02=   4
(1)  000002                  SW01=   2
(1)  000001                  SW00=   1
(1)                          .EQUIV  SW09,SW9
(1)                          .EQUIV  SW08,SW8
(1)                          .EQUIV  SW07,SW7
(1)                          .EQUIV  SW06,SW6
(1)                          .EQUIV  SW05,SW5
(1)                          .EQUIV  SW04,SW4
(1)                          .EQUIV  SW03,SW3
(1)                          .EQUIV  SW02,SW2
(1)                          .EQUIV  SW01,SW1
(1)                          .EQUIV  SW00,SW0
(1)
(1)                          ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
(1)  100000                  BIT15=  100000
(1)  040000                  BIT14=  40000
(1)  020000                  BIT13=  20000
(1)  010000                  BIT12=  10000
(1)  004000                  BIT11=  4000
(1)  002000                  BIT10=  2000
(1)  001000                  BIT09=  1000
(1)  000400                  BIT08=  400
(1)  000200                  BIT07=  200
(1)  000100                  BIT06=  100
(1)  000040                  BIT05=  40
(1)  000020                  BIT04=  20
(1)  000010                  BIT03=  10
(1)  000004                  BIT02=  4
(1)  000002                  BIT01=  2
(1)  000001                  BIT00=  1
```

```
(1)                             .EQUIV  BIT09,BIT9
(1)                             .EQUIV  BIT08,BIT8
(1)                             .EQUIV  BIT07,BIT7
(1)                             .EQUIV  BIT06,BIT6
(1)                             .EQUIV  BIT05,BIT5
(1)                             .EQUIV  BIT04,BIT4
(1)                             .EQUIV  BIT03,BIT3
(1)                             .EQUIV  BIT02,BIT2
(1)                             .EQUIV  BIT01,BIT1
(1)                             .EQUIV  BIT00,BIT0
(1)
(1)                             ;*BASIC "CPU" TRAP VECTOR ADDRESSES
(1)     000004                  ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
(1)     000010                  RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
(1)     000014                  TBITVEC=14              ;;"T" BIT
(1)     000014                  TRTVEC= 14              ;;TRACE TRAP
(1)     000014                  BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
(1)     000020                  IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
(1)     000024                  PWRVEC= 24              ;;POWER FAIL
(1)     000030                  EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
(1)     000034                  TRAPVEC=34              ;;"TRAP" TRAP
(1)     000060                  TKVEC=  60              ;;TTY KEYBOARD VECTOR
(1)     000064                  TPVEC=  64              ;;TTY PRINTER VECTOR
(1)     000240                  PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR
4248
4249                            ;;*****************************************************************************
4250
4251                            .SBTTL  RH11 REGISTERS
4252
4253                            ;;*****************************************************************************
4254
4255                            ;WORD COUNT REGISTER (RHWC)
4256                            ;EACH BIT IS CALLED BY BIT NUMBER
4257
4258                            ;BUS ADDRESS REGISTER (RHBA)
4259                            ;EACH BIT IS CALLED BY BIT NUMBER
4260
4261                            ;CONTROL AND STATUS REGISTER 2 (RHCS2)
4262
4263     000001                 US1=    1               ;UNIT SELECT (BIT #0)
4264     000002                 US2=    2               ;UNIT SELECT (BIT #1)
4265     000004                 US4=    4               ;UNIT SELECT (BIT #2)
4266     000010                 BAI=    10              ;BUS ADDRESS INCREMENT INHIBIT (BIT #3)
4267     000020                 PAT=    20              ;MASSBUS PARITY TEST (BIT #4)
4268     000040                 CLR=    40              ;CLEAR (BIT #5)
4269     000100                 IR=     100             ;INPUT READY (BIT #6)
4270     000200                 OR=     200             ;OUTPUT READY (BIT #7)
4271     000400                 MPE=    400             ;MASS BUS PARITY ERROR (BIT #8)
4272     001000                 MXF=    1000            ;MISSED TRANSFER ERROR (BIT #9)
4273     002000                 PGE=    2000            ;PROGRAM ERROR (BIT #10)
4274     004000                 NEM=    4000            ;NON EXISTANT MEMORY (BIT #11)
4275     010000                 NED=    10000           ;NON EXISTANT DRIVE (BIT #12)
4276     020000                 UPE=    20000           ;UNIBUS PARITY ERROR (BIT #13)
4277     040000                 WCE=    40000           ;WRITE CHECK ERROR (BIT #14)
```

```
4278          100000                  DLT=    100000                  ;DATA LATE (BIT #15)
4279
4280                                  ;DATA BUFFER REGISTER (RHDB)
4281                                  ;EACH BIT IS CALLED BY BIT NUMBER
4282
4283
4284                                  ;;*************************************************************************
4285
4286                                  .SBTTL  RP04 REGISTERS
4287
4288                                  ;;*************************************************************************
4289
4290                                  ;CONTROL AND STATUS 1 REGISTER. (#00)
4291
4292          000001                  GO=     1                       ;GO (BIT #0)
4293          000100                  IE=     100                     ;INTERRUPT ENABLE (BIT #6)
4294          000200                  RDY=    200                     ;READY (BIT #7)
4295          000400                  A16=    400                     ;HIGH ORDER UNIBUS BITS (BIT #8)
4296          001000                  A17=    1000                    ;HIGH ORDER UNIBUS BITS (BIT #9)
4297          002000                  PSEL=   2000                    ;PORT SELECT (BIT #10)
4298          004000                  DVA=    4000                    ;DEVICE AVAILABLE (BIT #11)
4299          020000                  MCPE=   20000                   ;MASSBUSS PARITY ERROR (BIT #13)
4300          040000                  TRE=    40000                   ;TRANSFER ERROR (BIT #14)
4301          100000                  SC=     100000                  ;SPECIAL CONDITION (BIT #15)
4302
4303                                  ;STATUS REGISTER (RHDS1) (#01)
4304
4305                                  ;DFS=   1                       DRIVE FORWARD 5"/SEC.  (BIT #0)
4306          000002                  DFF20=  2                       ;DRIVE FORWARD 20"/SEC.  (BIT #1)
4307          000004                  DIGB=   4                       ;DRIVE TO INNER GUARD BAND (BIT #2)
4308          000010                  GRV=    10                      ;GO REVERSE (BIT #3)
4309          000020                  DL64=   20                      ;DIFFERENCE LESS THAN 64 (BIT #4)
4310          000040                  DE1=    40                      ;DIFFERENCE EQUALS 1 (BIT #5)
4311          000100                  VV=     100                     ;VOLUME VALID (BIT #6)
4312          000200                  DRY=    200                     ;DRIVE READY (BIT #7)
4313          000400                  DPR=    400                     ;DRIVE PRESENT (BIT #8)
4314          001000                  PGM=    1000                    ;PROGRAMABLE (BIT #9)
4315          002000                  LST=    2000                    ;LAST SECTOR TRANSFERRED (BIT #10)
4316          004000                  WRL=    4000                    ;WRITE LOCK (BIT #11)
4317          010000                  MOL=    10000                   ;MEDIUM ON-LINE (BIT #12)
4318          020000                  PIP=    20000                   ;POSITIONING OPERATION IN PROGRESS (BIT #13)
4319          040000                  ERR=    40000                   ;COMPOSIT ERROR. (BIT #14)
4320          100000                  ATA=    100000                  ;ATTENTION ACTIVE (BIT #15;
4321
4322                                  ;ERROR REGISTER #01 (RHER1) (#02)
4323
4324          000001                  ILF=    1                       ;ILLEGAL FUNCTION (BIT #0)
4325          000002                  ILR=    2                       ;ILLEGAL REGISTER (BIT #1)
4326          000004                  RMR=    4                       ;REGISTER MODIFICATION REFUSED (BIT #2)
4327          000010                  PAR=    10                      ;PARITY ERROR (BIT #3)
4328          000020                  FER=    20                      ;FORMAT ERROR (BIT #4)
4329          000040                  WCF=    40                      ;WRITE CLOCK FAIL (BIT #5)
4330          000100                  ECH=    100                     ;ECC HARD ERROR (BIT #6)
4331          000200                  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 (RHMR)(#03)

                                DMD=    1               ;DIAGINOSTIC MODE (BIT #0)
                                MCLK=   2               ;MAINTAINABILITY CLOCK (BIT #1)
                                MINX=   4               ;MAINTAINABILITY INDEX (BIT #2)
                                MSTCK=  10              ;MAINTAINABILITY SECTOR CLOCK (BIT #3)
                                MRD=    20              ;MAINTAINABILITY READ (BIT #4)
                                MWR=    40              ;MAINTAINABILITY WRITE (BIT #5)
                                DTSY=   200             ;MAINTAINABILITY SYNC DETECTED (BIT #7)

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

                                AT0=    1               ;DEVICE 0 (BIT #0)
                                AT1=    2               ;DEVICE 1 (BIT #1)
                                AT2=    4               ;DEVICE 2 (BIT #2)
                                AT3=    10              ;DEVICE 3 (BIT #3)
                                AT4=    20              ;DEVICE 4 (BIT #4)
                                AT5=    40              ;DEVICE 5 (BIT #5)
                                AT6=    100             ;DEVICE 6 (BIT #6)
                                AT7=    200             ;DEVICE 7 (BIT #7)

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

                        ;DRIVE TYPE REGISTER (RHDT) (#06)
                        ;EACH BIT IS CALLED BY BIT NUMBER

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

                                EXT1=   1               ;EXTENSION 1 (BIT #0)
                                EXT2=   2               ;EXTENSION 2 (BIT #1)
                                EXT4=   4               ;EXTENSION 3 (BIT #2)
                                EXT10=  10              ;EXTENSION 4 (BIT #3)
                                EXT20=  20              ;EXTENSION 5 (BIT #4)
                                EXT40=  40              ;EXTENSION 6 (BIT #5)
                                SC1=    100             ;SECTOR COUNT FIELD 0 (BIT #6)
                                SC2=    200             ;SECTOR COUNT FIELD 1 (BIT #7)
                                SC4=    400             ;SECTOR COUNT FIELD 2 (BIT #8)
                                SC10=   1000            ;SECTOR COUNT FIELD 3 (BIT #9)
                                SC20=   2000            ;SECTOR COUNT FIELD 4 (BIT #10)
                                TRK1=   4000            ;TRACK FIELD 1 (BIT #11)
                                TRK2=   10000           ;TRACK FIELD 2 (BIT #12)
                                TRK4=   20000           ;TRACK FIELD 3 (BIT #13)
                                TRK10=  40000           ;TRACK FIELD 4 (BIT #14)
                                TRK20=  100000          ;TRACK FIELD 5 (BIT #15)
```

```
4386
4387                                    ;ERROR REGISTER #2 (RHER2) (#10)
4388
4389            000001                  WCU=    1               ;WRITE CURRENT UNSAFE (BIT #0)
4390            000002                  CSF=    2               ;CURRENT SINK FAILURE (BIT #1)
4391            000004                  WSU=    4               ;WRITE SELECT UNSAFE (BIT #2)
4392            000010                  CSU=    10              ;CURRENT SWITCH UNSAFE (BIT #3)
4393            000020                  MSE=    20              ;MOTOR SEQUENCE ERROR (BIT #4)
4394            000040                  TDF=    40              ;TRANSITIONS DETECTOR FAILURE (BIT #5)
4395            000100                  TUF=    100             ;TRANSITIONS UNSAFE (BIT #6)
4396            000200                  FEN=    200             ;FAILSAFE ENABLED (BIT #7)
4397            000400                  WRU=    400             ;WRITE READY UNSAFE (BIT #8)
4398            001000                  MHS=    1000            ;MULTIPLE HEAD SELECT (BIT #9)
4399            002000                  NHS=    2000            ;NO HEAD SELECTION (BIT #10)
4400            004000                  IXE=    4000            ;INDEX ERROR (BIT #11)
4401            010000                  VU30=   10000           ;30VOLT UNSAFE (BIT #12)
4402            020000                  PLU=    20000           ;PLO UNSAFE (BIT #13)
4403            100000                  ACU=    100000          ;AC UNSAFE (BIT #15)
4404
4405                                    ;OFFSET REGISTER (RHOF) (#11)
4406
4407            000001                  OF25=   1               ;OFFSET 25 MICRO INCHES (BIT #0)
4408            000002                  OF50=   2               ;OFFSET 50 MICRO INCHES (BIT #1)
4409            000004                  OF100=  4               ;OFFSET 100 MICRO INCHES (BIT #2)
4410            000010                  OF200=  10              ;OFFSET 200 MICRO INCHES (BIT #3)
4411            000020                  OF400=  20              ;OFFSET 400 MICRO INCHES (BIT #4)
4412            000040                  OF800=  40              ;OFFSET 800 MICRO INCHES (BIT #5)
4413            000200                  OFREV=  200             ;OFFSET NEGATIVE (REVERSE) (BIT #5)
4414            002000                  HCI=    2000            ;HEADER COMPARE INHIBIT (BIT #10)
4415            004000                  ECI=    4000            ;ERROR CORRECTION CODE INHIBIT (BIT #11)
4416            010000                  FMT22=  10000           ;FORMAT BIT (BIT #12
4417
4418                                    ;DESIRED CYLINDER ADDRESS (RHCA) (#12)
4419                                    ;EACH BIT IS CALLED BY BIT NUMBER.
4420
4421                                    ;CURRENT CYLINDER ADDRESS (RHCC) (#13)
4422                                    ;EACH BIT IS CALLED BY BIT NUMBER
4423
4424                                    ;SERIAL NUMBER REGISTER (RHSN) (#14)
4425                                    ;EACH IS CALLED BY BIT NUMBER
4426
4427                                    ;ERROR REGISTER #03 (RHER3) (#15)
4428
4429            000001                  PSU=    1               ;PACK SPEED UNSAFE (BIT #0)
4430            000002                  VUF=    2               ;VELOCITY UNSAFE (BIT #1)
4431            000010                  UWR=    10              ;ANY UNSAFE EXCEPT READ/WRITE (BIT #3)
4432            000020                  PRE=    20              ;DISK PACK ROTATION ERROR (BIT #4)
4433            000040                  ACL=    40              ;AC LOW (BIT #5)
4434            000100                  DCL=    100             ;DC LOW (BIT #6)
4435            040000                  SKI=    40000           ;SEEK INCOMPLETE (BIT #14)
4436            100000                  OCYL=   100000          ;OFF CYLINDER (BIT #15)
4437
4438                                    ;ECC POSITION REGISTER (RHEC1) (#16)
4439                                    ;EACH BIT IS CALLED BY BIT NUMBER
```

```
4440
4441                                    ;ECC PATTERN REGISTER (RHEC2) (#17)
4442                                    ;EACH BIT IS CALLED BY BIT NUMBER
4443
4444                                    ;;*************************************************************************
4445
4446                                    .SBTTL  DEFINITIONS OF THE RH11/RPQ4 ADDRESS INDEXES
4447
4448                                    ;;*************************************************************************
4449
4450            000000                  RHCS1=0                 ;CONTROL AND STATUS REGISTER #1 (DRIVE REG. 00)
4451            000002                  RHWC=2                  ;WORD COUNT REGISTER (NOT A DRIVE REG)
4452            000004                  RHBA=4                  ;UNIBUS ADDRESS REGISTER (NOT A DRIVE REG)
4453            000006                  RHDA=6                  ;DESIRED SECTOR/TRACK ADDRESS REGISTER (DRIVE REG. 05)
4454            000010                  RHCS2=10                ;CONTROL AND STATUS REGISTER #2 (NOT A DRIVE REG)
4455            000012                  RHDS1=12                ;DRIVE STATUS REGISTER (DRIVE REG 01)
4456            000014                  RHER1=14                ;ERROR REGISTER #1 (DRIVE REG. 02)
4457            000016                  RHAS=16                 ;ATTENTION SUMMARY PSEUDO REGISTER (DRIVE REG. 04)
4458            000020                  RHLA=20                 ;LOOK AHEAD REGISTER (DRIVE REG. 07)
4459            000022                  RHDB=22                 ;DATA BUFFER REGISTER (NOT A DRIVE REG.)
4460            000024                  RHMR=24                 ;MAINTAINABILITY REGISTER (DRIVE REG. 03)
4461            000026                  RHDT=26                 ;DRIVE TYPE REGISTER (DRIVE REG. 06)
4462            000030                  RHSN=30                 ;SERIAL NUMBER REGISTER (DRIVE REG. 10)
4463            000032                  RHOF=32                 ;OFFSET REGISTER (DRIVE REG. 11)
4464            000034                  RHCA=34                 ;DESIRED CYLINDER ADDRESS REGISTER (DRIVE REG. 12)
4465            000036                  RHCC=36                 ;CURRENT CYLINDER ADDRESS REGISTER (DRIVE REG. 13)
4466            000040                  RHER2=40                ;ERROR REGISTER #2 (DRIVE REG. 14)
4467            000042                  RHER3=42                ;ERROR REGISTER #3 (DRIVE REG. 15)
4468            000044                  RHEC1=44                ;ECC POSITION REGISTER (DRIVE REG. 16)
4469            000046                  RHEC2=46                ;ECC PATTERN REGISTER (DRIVE REG.  17)
4470
4471
 (1)                                    .SBTTL  TRAP CATCHER
 (1)
 (1)            000000                          .=0
 (1)                                    ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
 (1)                                    ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
 (1)                                    ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
 (1)
 (1)                                    .SBTTL  STARTING ADDRESS(ES)
 (1)            000200                          .=200
 (1)
 (1)  000200    000137  001676                  JMP     @#START         ;;JUMP TO STARTING ADDRESS OF PROGRAM
4472                                    ;*STARTING ADDRESS IS LOCATION 200
4473
4474  000204    000137  002262                  JMP     EXEC            ;RESTART
4475                                    ;*RESTART ADDRESS IS LOCATION 204
4476
4477  000210    000137  002430                  JMP     CHANGE          ;CHANGE RH11 ADDRESS AND START
4478                                    ;*START AT LOCATION 210 TO CHANGE THE RH11 ADDRESS FROM 176700
4479
4480
4516
```

```
4517                                ;*****************************************************************
 (1)
 (1)                                .SBTTL  COMMON TAGS
 (1)
 (1)                                ;*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
 (1)                                ;*USED IN THE PROGRAM.
 (1)
 (1)      000046                          .=46
 (1)  000046  017726                      $ENDAD                  ;;LOGICAL END OF PROGRAM
 (1)
 (1)      001100                          .=1100
 (1)
 (1)  001100                        $CMTAG:                       ;;START OF COMMON TAGS
 (1)  001100  000000                $PASS:  .WORD   0             ;;CONTAINS PASS COUNT
 (1)  001102     000                $TSTNM: .BYTE   0             ;;CONTAINS THE TEST NUMBER
 (1)  001103     000                $ERFLG: .BYTE   0             ;;CONTAINS ERROR FLAG
 (1)  001104  000000                $ICNT:  .WORD   0             ;;CONTAINS SUBTEST ITERATION COUNT
 (1)  001106  000000                $LPADR: .WORD   0             ;;CONTAINS SCOPE LOOP
 (1)  001110  000000                $LPERR: .WORD   0             ;;CONTAINS SCOPE RETURN FOR ERRORS
 (1)  001112  000000                $ERTTL: .WORD   0             ;;CONTAINS TOTAL ERRORS DETECTED
 (1)  001114     000                $ITEMB: .BYTE   0             ;;CONTAINS ITEM CONTROL BYTE
 (1)  001115     001                $ERMAX: .BYTE   1             ;;CONTAINS MAX. ERRORS PER TEST
 (1)  001116  000000                $ERRPC: .WORD   0             ;;CONTAINS PC OF LAST ERROR INSTRUCTION
 (1)  001120  000000                $GDADR: .WORD   0             ;;CONTAINS  OF 'GOOD' DATA
 (1)  001122  000000                $BDADR: .WORD   0             ;;CONTAINS  OF 'BAD' DATA
 (1)  001124  000000                $GDDAT: .WORD   0             ;;CONTAINS 'GOOD' DATA
 (1)  001126  000000                $BDDAT: .WORD   0             ;;CONTAINS 'BAD' DATA
 (1)  001130  000000  000000  000000        .WORD   0,0,0         ;;RESERVED--NOT TO BE USED
 (1)  001136  177560                $TKS:   177560                ;;TTY KBD STATUS
 (1)  001140  177562                $TKB:   177562                ;;TTY KBD BUFFER
 (1)  001142  177564                $TPS:   177564                ;;TTY PRINTER STATUS REG.
 (1)  001144  177566                $TPB:   177566                ;;TTY PRINTER BUFFER REG.
 (1)  001146     000                $NULL:  .BYTE   0             ;;CONTAINS NULL CHARACTER FOR FILLS
 (1)  001147     002                $FILLS: .BYTE   2             ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
 (1)  001150     012                $FILLC: .BYTE   12            ;;INSERT FILL CHARS. AFTER A "LINE FEED"
 (1)  001151     000                $TPFLG: .BYTE   0             ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
 (1)  001152  000000                $REGAD: .WORD   0             ;;CONTAINS THE  FROM
 (1)                                                              ;;WHICH  ($REGO) WAS OBTAINED
 (3)  001154  000000                $REGO:  .WORD   0             ;;CONTAINS (($REGAD)+0)
 (3)  001156  000000                $TMP0:  .WORD   0             ;;USER DEFINED
 (3)  001160  000000                $TMP1:  .WORD   0             ;;USER DEFINED
 (3)  001162  000000                $TMP2:  .WORD   0             ;;USER DEFINED
 (3)  001164  000000                $TMP3:  .WORD   0             ;;USER DEFINED
 (3)  001166  000000                $TMP4:  .WORD   0             ;;USER DEFINED
 (1)  001170  000000                $TIMES: 0                     ;;MAX. NUMBER OF ITERATIONS
 (1)  001172  000000                $ESCAPE:0                     ;;ESCAPE ON ERROR
 (1)  001174  177607  000377        $BELL:  .ASCIZ  <207><377><377>;;CODE FOR BELL
 (1)  001200     077                $QUES:  .ASCII  /?/           ;;QUESTION MARK
 (1)  001201     015                $CRLF:  .ASCII  <15>          ;;CARRIAGE RETURN
 (1)  001202  000012                $LF:    .ASCIZ  <12>          ;;LINE FEED
 (3)  001204  172540                $LKCSR: .WORD   172540        ;ADDR OF KW11-P STATUS REGISTER
 (3)  001206  172542                $LKCSB: .WORD   172542        ;ADDR OF KW11-P COUNTER BUFFER
 (3)  001210  000104                $LPVEC: .WORD   104           ;ADDR OF KW11-P VECTOR
 (3)  001212  177546                $LKS:   .WORD   177546        ;ADDR OF KW11-L STATUS REGISTER
```

```
(3)  001214  000100                $LLVEC: .WORD   100              ;ADDR OF KW11-L VECTOR
(3)  001216  000000                PORTA:  .WORD   0                ;ADDRESS OF PORT A
(3)  001220  000000                PORTB:  .WORD   0                ;ADDRESS OF PORT B
(3)  001222  000000                PORTC:  .WORD   0                ;ADDRESS OF DIFFERENT DRIVE
(3)  001224  000000                ASR1:   .WORD   0                ;ATA-A OR ATA-B = 1
(3)  001226  000000                PTNBR:  .WORD   0                ;CONTAINS THE PORT ADDRESS FOR ERROR TYPEOUTS
(3)  001230  000000                SEIZPT: .WORD   0                ;CONTAINS THE ADDRESS OF THE SEIZING PORT
(3)  001232  000000                OPPRT:  .WORD   0                ;CONTAINS THE ADDRESS OF THE 'OPPOSITE' PORT
(3)  001234  000000                TSTNUM: .WORD   0                ;NUMBER OF THE CURRENT TEST
(3)  001236  000000                CKERR:  .WORD   0                ;IF -1, A REGISTER MISCOMPARISON OCCURED
(3)  001240  000000                NOSEIZ: .WORD   0                ;IF -1, THE PORT IN 'SEIZPT' DID NOT SEIZE THE DRIVE
(3)  001242  000000                RELERR: .WORD   0                ;IF -1, THE PORT IN 'SEIZPT' DID NOT RELEASE THE DRIVE
(3)  001244  000000                TIME:   .WORD   0                ;ELAPSED TIME COUNTER
(3)  001246  000000                WATCH:  .WORD   0                ;WATCH DOG TIMER LOCATION
(3)  001250  000000                TIMEA:  .WORD   0                ;THE TIMEOUT ONE-SHOT VALUE MEASURED THROUGH PORT A
(3)  001252  000000                TIMEAP: .WORD   0                ;PORT A TIMEOUT VALUE + 25%
(3)  001254  000000                TIMEB:  .WORD   0                ;THE TIMEOUT ONE-SHOT VALUE MEASURED THROUGH PORT B
(3)  001256  000000                TIMEBP: .WORD   0                ;PORT B TIMEOUT VALUE + 25%
(3)  001260  000000                KYBCTL: .WORD   0                ;SINGLE TEST INDICATOR
(3)
(4)                                ;;*******************************************************************************
(3)
(3)                                .SBTTL  RH11/RP04 UNIBUS AND VECTOR ADDRESSES
(3)
(4)                                ;;*******************************************************************************
(3)
(3)  001262  176700                $RPADR: .WORD   176700           ;RH11/RP04 UNIBUS ADDRESS
(3)  001264  000254                $RPVEC: .WORD   254              ;RH11 INTERRUPT VECTOR ADDRESS
(3)
```

```
(2)                                     ;*****************************************************************
(1)
(1)                                     .SBTTL  ERROR POINTER TABLE
(1)
(1)                                     ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
(1)                                     ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
(1)                                     ;*LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
(1)                                     ;*NOTE1:        IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
(1)                                     ;*NOTE2:        EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:
(1)
(1)                                     ;*      EM              ;;POINTS TO THE ERROR MESSAGE
(1)                                     ;*      DH              ;;POINTS TO THE DATA HEADER
(1)                                     ;*      DT              ;;POINTS TO THE DATA
(1)                                     ;*      DF              ;;POINTS TO THE DATA FORMAT
(1)
(1)
(1)  001266                             $ERRTB:
 4518
 4519
 4520                                   ;ERROR 1
 4521
 4522  001266  023331                           EM1     ;DRIVE IS NON-EXISTENT ('NED' BIT SET)
 4523  001270  026036                           DH1
 4524  001272  027410                           DT1
 4525  001274  027624                           DF1
 4526
 4527                                   ;ERROR 2
 4528
 4529  001276  023377                           EM2     ;WRONG DRIVE TYPE
 4530  001300  026107                           DH2
 4531  001302  027424                           DT2
 4532  001304  027631                           DF2
 4533
 4534                                   ;ERROR 3
 4535
 4536  001306  023420                           EM3     ;CONTROLLER SELECT SWITCH ON DRIVE NOT IN 'A/B'
 4537  001310  026036                           DH1
 4538  001312  027410                           DT1
 4539  001314  027624                           DF1
 4540
 4541                                   ;ERROR 4
 4542
 4543  001316  023477                           EM4     ;DRIVE NOT ON LINE
 4544  001320  026107                           DH2
 4545  001322  027424                           DT2
 4546  001324  027631                           DF2
 4547
 4548                                   ;ERROR 5
 4549
 4550  001326  023521                           EM5     ;SERIAL NUMBER READ THROUGH EACH PORT NOT THE SAME
 4551  001330  026163                           DH5
 4552  001332  027442                           DT5
 4553  001334  027637                           DF5
 4554
```

```
                                        ;ERROR 6
                                                EM6     ;TIMEOUT HAS NOT OCCURED WITHIN 2 SECONDS
001336  023603                                  DH6
001340  026232                                  DT6
001342  027456                                  DF6
001344  027644

                                        ;ERROR 7

001346  023654                                  EM7     ;TIMEOUT ONE-SHOT IS LESS THAN 500 MS
001350  026261                                  DH7
001352  027466                                  DT7
001354  027647                                  DF7

                                        ;ERROR 10

001356  023721                                  EM10    ;READIN PRESET DOES NOT SET VOLUME VALID FOR THE PORT
001360  026036                                  DH1
001362  027410                                  DT1
001364  027624                                  DF1

                                        ;ERROR 11

001366  024006                                  EM11    ;'GO' BIT RESET DURING UNLOAD COMMAND
001370  026036                                  DH1
001372  027410                                  DT1
001374  027624                                  DF1

                                        ;ERROR 12

001376  024053                                  EM12    ;INCORRECT STATUS DURING UNLOAD COMMAND
001400  026036                                  DH1
001402  027410                                  DT1
001404  027624                                  DF1

                                        ;ERROR 13

001406  024122                                  EM13    ;DRIVE DID NOT RETURN TO NEUTRAL AFTER UNLOAD COMMAND
001410  026326                                  DH13
001412  027500                                  DT13
001414  027644                                  DF6

                                        ;ERROR 14

001416  024207                                  EM14    ;ATTENTION BIT SET ON 'OPPOSITE PORT' AFTER UNLOAD
001420  026404                                  DH14
001422  027510                                  DT14
001424  027653                                  DF14

                                        ;ERROR 15

001426  024271                                  EM15    ;ATTENTION BIT NOT SET ON PORT WHICH ISSUED 'UNLOAD'
001430  026036                                  DH1
001432  027410                                  DT1
```

```
4609  001434  027624                          DF1
4610
4611                                ;ERROR 16
4612
4613  001436  024355                          EM16    ;DRIVE NOT IN NEUTRAL AFTER UNLOAD WITH 'CONTROLLER
4614                                                  ;SELECT' SWITCH MOVED FROM 'A/B'
4615  001440  026326                          DH13
4616  001442  027500                          DT13
4617  001444  027644                          DF6
4618                                ;ERROR 17
4619
4620  001446  024501                          EM17    ;DRIVE LOCKED ON PORT 'A' BY SWITCH WHILE CYCLED UP
4621  001450  026524                          DH17
4622  001452  027526                          DT17
4623  001454  027661                          DF17
4624
4625                                ;ERROR 20
4626
4627  001456  024564                          EM20    ;DRIVE LOCKED ON PORT 'B' BY SWITCH WHILE CYCLED UP
4628  001460  026524                          DH17
4629  001462  027526                          DT17
4630  001464  027661                          DF17
4631
4632                                ;ERROR 21
4633
4634  001466  024647                          EM21    ;STATUS INCORRECT FOR PORT AFTER CYCLE UP
4635  001470  026036                          DH1
4636  001472  027410                          DT1
4637  001474  027624                          DF1
4638
4639                                ;ERROR 22
4640
4641  001476  024720                          EM22    ;REGISTER CONTENTS SEEN WHEN DRIVE SWITCHED ON 'OPPOSITE' PORT
4642  001500  026036                          DH1
4643  001502  027410                          DT1
4644  001504  027624                          DF1
4645
4646                                ;ERROR 23
4647
4648  001506  025016                          EM23    ;'NED' SET WHEN RHDS1 ACCESSED THROUGH PORT NOT SWITCHED
4649  001510  026036                          DH1
4650  001512  027410                          DT1
4651  001514  027624                          DF1
4652
4653                                ;ERROR 24
4654
4655  001516  025106                          EM24    ;DRIVE SWITCHED TO LOCKED OUT PORT WHEN RELEASED/
4656  001520  026543                          DH24
4657  001522  027534                          DT24
4658  001524  027647                          DF7
4659
4668                                ;ERROR 25
 (1)
 (1)  001526  000000                          0       ;UNUSED ERROR MESSAGES
```

```
    (1)  001530  000000                    0
    (1)  001532  000000                    0
    (1)  001534  000000                    0
    (1)
    (1)                              ;ERROR 26
    (1)
    (1)  001536  000000                    0       ;UNUSED ERROR MESSAGES
    (1)  001540  000000                    0
    (1)  001542  000000                    0
    (1)  001544  000000                    0
    (1)
    (1)                              ;ERROR 27
    (1)
    (1)  001546  000000                    0       ;UNUSED ERROR MESSAGES
    (1)  001550  000000                    0
    (1)  001552  000000                    0
    (1)  001554  000000                    0
    (1)
  4669                               ;ERROR 30
  4670
  4671  001556  025166                    EM30    ;DRIVE NOT SEIZED BY PORT 'N'
  4672  001560  026647                    DH30
  4673  001562  027546                    DT30
  4674  001564  027663                    DF30
  4675
  4676                               ;ERROR 31
  4677
  4678  001566  025217                    EM31    ;WRONG STATUS SEEN BY THE SEIZING PORT
  4679  001570  026107                    DH2
  4680  001572  027424                    DT2
  4681  001574  027631                    DF2
  4682
  4683                               ;ERROR 32
  4684
  4685  001576  025265                    EM32    ;REGISTER CONTENTS INCORRECT
  4686  001600  026107                    DH2
  4687  001602  027424                    DT2
  4688  001604  027631                    DF2
  4689
  4690                               ;ERROR 33
  4691
  4692  001606  025315                    EM33    ;CONTROL BUS PARITY ERROR WHILE READING REGISTER
  4693  001610  026036                    DH1
  4694  001612  027410                    DT1
  4695  001614  027624                    DF1
  4696
  4697                               ;ERROR 34
  4698
  4699  001616  025401                    EM34    ;CAN'T ACCESS DRIVE THROUGH EITHER PORT
  4700  001620  026772                    DH34
  4701  001622  027566                    DT34
  4702  001624  027672                    DF34
  4703
  4704                               ;ERROR 35
```

```
4705
4706  001626  025450                          EM35    ;DRIVE NOT IN NEUTRAL AFTER RELEASE, REQUEST NOT SET
4707  001630  027070                          DH35
4708  001632  027600                          DT35
4709  001634  027647                          DF7
4710
4711                                  ;ERROR 36
4712
4713  001636  025535                          EM36    ;DRIVE NOT IN NEUTRAL AFTER TIMEOUT, REQUEST NOT SET
4714  001640  027070                          DH35
4715  001642  027600                          DT35
4716  001644  027647                          DF7
4717
4718                                  ;ERROR 37
4719
4720  001646  025622                          EM37    ;REGISTER CONTENTS INCORRECT AFTER RELEASE/TIMEOUT
4721  001650  027166                          DH37
4722  001652  027546                          DT30
4723  001654  027663                          DF30
4724
4725                                  ;ERROR 40
4726
4727  001656  025703                          EM40    ;DRIVE NOT SEIZED BY PORT AFTER RELEASE WIITH REQUEST SET
4728  001660  027311                          DH40
4729  001662  027612                          DT40
4730  001664  027647                          DF7
4731
4732                                  ;ERROR 41
4733
4734  001666  025760                          EM41    ;REGISTER WRONG AFTER RELEASE WITH REQUEST SET
4735  001670  026647                          DH30
4736  001672  027546                          DT30
4737  001674  027663                          DF30
4738
4739
4740
4754  001676                          START:
 (1)  001676  012737  000340  177776          MOV     #340,@#PS       ;;LOCK OUT ALL INTERRUPTS
 (1)  001704  012706  001100                  MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
 (1)  001710  005026                          CLR     (R6)+           ;;CLEAR MEMORY LOCATION
 (1)  001712  022706  001136                  CMP     #$TKS,R6        ;;DONE?
 (1)  001716  001374                          BNE     .-6             ;;LOOP BACK IF NO
 (1)  001720  012706  001100                  MOV     #STACK,SP       ;;SETUP THE STACK POINTER
 (1)  001724  012737  017762  000020          MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
 (1)  001732  012737  000340  000022          MOV     #340,@#IOTVEC+2 ;;LEVEL 7
 (1)  001740  012737  020140  000030          MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
 (1)  001746  012737  000340  000032          MOV     #340,@#EMTVEC+2 ;;LEVEL 7
 (1)  001754  012737  021750  000034          MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
 (1)  001762  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
 (1)  001770  013737  017656  017650          MOV     $ENDCT,$EOPCT   ;;SETUP END-OF-PROGRAM COUNTER
 (1)  001776  005037  001170                  CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
 (1)  002002  005037  001172                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
 (1)  002006  112737  000001  001115          MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
 (1)  002014  012737  002014  001106          MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
```

```
 (1)  002022  012737  002022  001110          MOV     #.,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
4755  002030  000005                          RESET                   ;CLEAR THE SYSTEM
4756  002032  104400  022014          START1: TYPE    ,TITLE          ;TYPE PROGRAM NAME
4757  002036  012737  000240  002032          MOV     #NOP,START1     ;DISABLE TITLE TYPEOUT AFTER INITIAL START
4758  002044  012737  000240  002034          MOV     #NOP,START1+2   ;FROM LOCATION 200 OR 210
4759  002052  104400  022117          1$:     TYPE    ,ENTERA         ;ENTER DRIVE ADDRESS
4760  002056  104416                          RDOCT                   ;GET THE ADDRESS
4761  002060  012637  001216                  MOV     (SP)+,PORTA     ;STORE THE ADDRESS
4762  002064  023727  001216  000007          CMP     PORTA,#7        ;SEE IF ADDRESS TOO LARGE
4763  002072  101403                          BLOS    2$              ;BR IF NOT
4764  002074  104400  022147                  TYPE    ,ADRERR         ;TYPE ADDRESS ERROR MESSAGE
4765  002100  000764                          BR      1$              ;TRY AGAIN
4766  002102  013737  001216  001220  2$:     MOV     PORTA,PORTB     ;GENERATE THE PORT B ADDRESS
4767  002110  005237  001220                  INC     PORTB           ;INCREMENT THE ADDRESS
4768  002114  042737  000006  001220          BIC     #6,PORTB        ;LEAVE BIT 0
4769  002122  013746  001216                  MOV     PORTA,-(SP)     ;PUT PORT A ADDRESS ON THE STACK
4770  002126  042716  177771                  BIC     #^C6,(SP)       ;SAVE BITS 1 & 2
4771  002132  052637  001220                  BIS     (SP)+,PORTB     ;SET BITS 1 & 2 IN PORT B ADDRESS
4772  002136  104400  022171                  TYPE    ,PORTAIS        ;'PORT A ADDRESS IS '
4773  002142  013746  001216                  MOV     PORTA,-(SP)     ;PUT THE ADDRESS ON THE STACK
4774  002146  104410                          TYPOS                   ;TYPE PORT A ADDRESS
4775  002150  104400  022217                  TYPE    ,PORTBIS        ;'PORT B ADDRESS IS '
4776  002154  013746  001220                  MOV     PORTB,-(SP)     ;PUT ADDRESS ON THE STACK
4777  002160  104410                          TYPOS                   ;TYPE PORT B ADDRESS
4778  002162  104400  001201                  TYPE    ,$CRLF          ;ANOTHER CR-LF
4779  002166  013737  001216  001222          MOV     PORTA,PORTC     ;GENERATE ADDRESS OF DRIVE NOT TESTED
4780  002174  062737  000006  001222          ADD     #6,PORTC        ;COMPLEMENT SOME BITS
4781  002202  042737  177770  001222          BIC     #^C7,PORTC      ;SAVE ONLY LOWER BITS
4782  002210  013701  001216                  MOV     PORTA,R1        ;USE PORT A ADDRESS AS INDEX
4783  002214  116137  027726  001224          MOVB    ATABIT(R1),ASR1 ;GET ATTENTION BIT FOR DRIVE
4786  002222  005037  001250                  CLR     TIMEA           ;CLEAR TIMEOUT ONE-SHOT VALUE LOCATION
 (1)  002226  005037  001252                  CLR     TIMEAP          ;CLEAR TIMEOUT ONE-SHOT VALUE LOCATION
 (1)  002232  005037  001254                  CLR     TIMEB           ;CLEAR TIMEOUT ONE-SHOT VALUE LOCATION
 (1)  002236  005037  001256                  CLR     TIMEBP          ;CLEAR TIMEOUT ONE-SHOT VALUE LOCATION
4787  002242  004737  017376                  JSR     PC,CKCLK        ;SETUP CLOCK
4788  002246  000137  002262                  JMP     EXEC            ;CLOCK HAS BEEN STARTED
4789  002252  104400  022245                  TYPE    ,NOCLOCK        ;NO CLOCK ON SYSTEM
4790  002256  000000                          HALT                    ;FATAL ERROR
4791  002260  000776                          BR      .-2             ;INTERLOCK HALT
4792
4793                                  ;ROUTINE TO GET THE TEST NUMBER FROM THE OPERATOR
4794
4795  002262  000005                  EXEC:   RESET                   ;CLEAR EVERYTHING
4796  002264  005037  177776                  CLR     PS              ;CLEAR THE PROCESSOR STATUS WORD
4797  002270  104400  001201                  TYPE    ,$CRLF          ;CR-LF
4798  002274  013700  001262                  MOV     $RPADR,R0       ;RH11 ADDRESS FOR INDEXING
4799  002300  012706  001100                  MOV     #STACK,R6       ;LOAD STACK POINTER
4800  002304  004737  017376                  JSR     PC,CKCLK        ;START THE CLOCK
4801  002310  000240                          NOP                     ;RETURN IF NO CLOCK
4802  002312  005037  001260                  CLR     KYBCTL          ;CLEAR SINGLE TEST INDICATOR
4803  002316  005037  001100                  CLR     $PASS           ;CLEAR THE PASS COUNT
4804  002322  112737  000001  001115          MOVB    #1,$ERMAX       ;SET ERROR MAX TO 1
4805  002330  012737  002330  001106          MOV     #.,$LPADR       ;INITIAL SETTING FOR LOOP ADDRESS
4806  002336  012737  002336  001110          MOV     #.,$LPERR       ;INITIAL SETTING FOR LOOP ON ERROR ADDRESS
```

```
4807  002344  104400  022314                TYPE    ,TESTNO         ;ASK FOR TEST NUMBER
4808  002350  104416                       RDOCT                   ;GET THE NUMBER
4809  002352  012601                       MOV     (SP)+,R1        ;PUT ENTRY INTO R1
4810  002354  001002                       BNE     .+6             ;BR IF NOT ZERO
4811  002356  000137  002560               JMP     TST1            ;ENTER ZERO - PERFORM ALL TESTS
4812  002362  020137  027736               CMP     R1,MAXTN        ;SEE IF NUMBER GREATER THAN MAXIMUM
4813  002366  003403                       BLE     1$              ;BR IF LESS OR EQUAL
4814  002370  104400  022334               TYPE    ,BADNO          ;BAD ENTRY
4815  002374  000732                       BR      EXEC            ;TRY AGAIN
4816  002376  005301              1$:      DEC     R1              ;DECREMENT ENTRY
4817  002400  006301                       ASL     R1              ;SHIFT IT LEFT
4818  002402  016137  027676  002426       MOV     TSTADR(R1),2$   ;GET THE TEST ADDRESS
4819  002410  005237  001260               INC     KYBCTL          ;SET SINGLE TEST INDICATOR
4820  002414  012737  000001  001104       MOV     #1,$ICNT        ;PRESET ITERATION COUNT
4821  002422  000177  000000               JMP     @2$             ;GO TO THE SELECTED TEST
4822  002426  000000              2$:      .WORD   0               ;TEST ADDRESS GOES HERE
4823
4824                                       ;CHANGE THE RH11 UNIBUS ADDRESS USED BY THE PROGRAM
4825
4826  002430  000005              CHANGE:  RESET                   ;CLEAR THE SYSTEM
4827  002432  012737  000340  177776       MOV     #340,@#PS       ;LOCK OUT ALL INTERRUPTS
4828  002440  012706  001100               MOV     #STACK,SP       ;LOAD THE STACK POINTER
4829  002444  012737  021750  000034       MOV     #$TRAP,@#TRAPVEC  ;LOAD TRAP VECTOR
4830  002452  012737  000340  000036       MOV     #340,@#TRAPVEC+2  ;LEVEL 7
4831  002460  104400  022362               TYPE    ,ADDRIS         ;TYPE OUT WHAT THE PRESENT ADDRESS IS
4832  002464  013746  001262               MOV     $RPADR,-(SP)    ;PUT THE ADDRESS ON THE STACK
4833  002470  104402                       TYPOC                   ;TYPE THE ACTUAL ADDRESS
4834  002472  104400  001201               TYPE    ,$CRLF          ;CR-LF
4835  002476  104400  022442               TYPE    ,NTRH11         ;ASK FOR NEW ADDRESS
4836  002502  104416                       RDOCT
4837  002504  005716                       TST     (SP)            ;0 OR 'CR' ENTERED ?
4838  002506  001402                       BEQ     1$              ;BR IF EITHER ENTERED (NO ADDRESS CHANGE)
4839  002510  011637  001262               MOV     (SP),$RPADR     ;NEW RH11 ADDRESS
4840  002514  012737  002536  000004  1$:  MOV     #2$,4           ;LOAD TRAP ADDRESS
4841  002522  013700  001262               MOV     $RPADR,R0       ;RH11 ADDRESS
4842  002526  062700  000002               ADD     #2,R0           ;FORM ADDRESS OF RHWC
4843  002532  005710                       TST     (R0)            ;SEE IF RH11 RESPONDS AT  THAT ADDRESS
4844  002534  000405                       BR      3$              ;BR, RH11 ALIVE AT PRESENT ADDRESS
4845  002536  104400  022474       2$:     TYPE    ,NORESP         ;REPORT NO RESPONSE
4846  002542  010046                       MOV     R0,-(SP)        ;SETUP TO CONVERT THE ADDRESS
4847  002544  104402                       TYPOC                   ;TYPE THE ADDRESS
4848  002546  000730                       BR      CHANGE          ;GET ADDRESS AGAIN
4849  002550  000137  001676       3$:     JMP     START           ;GO TO THE STARTING ADDRESS
4850
4851                                       ;;**************************************************************************
4852
4853                                       .SBTTL  *** TESTS ***
4854
4855                                       ;;**************************************************************************
4856
4869
4870  002554  013700  001262       TST1AA: MOV     $RPADR,R0       ;;RESTORE R0 AFTER END OF PASS
4871
4872                                       ;**********************************************************************
```

```
                                        ;*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 RPQ4, THAT THE DRIVE IS ONLINE (RHDS1 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.
                                        ;*
                                        ;*****************************************************************
002560                                  TST1:
002560  000004                                  SCOPE                   ;INITIALIZE THE SCOPE HANDLER
002562  005737  001260                          TST     KYBCTL          ;PERFORMING ONLY SINGLE TESTS ?
002566  001406                                  BEQ     2$              ;BR IF NOT
002570  100002                                  BPL     1$              ;BR IF JUST ENTERED TEST
002572  000137  002262                          JMP     EXEC            ;RETURN & GET NEXT TEST NUMBER
002576  012737  177777  001260      1$:         MOV     #-1,KYBCTL      ;SET SINGLE TEST INDICATOR
002604  112737  000001  001102      2$:         MOVB    #1,$TSTNM       ;TEST NUMBER
002612  012737  002634  001106                  MOV     #TEST1,$LPADR   ;LOAD LOOP ON TEST ADDRESS
002620  012737  002634  001110                  MOV     #TEST1,$LPERR   ;LOAD LOOP ON ERROR ADDRESS
002626  012737  000001  001170                  MOV     #1,$TIMES       ;;DO 1 ITERATION
                                        ;;*************************************************************************
                                        ;END OF 'SCOPE' SETUP - START OF MAIN TEST

002634                                  TEST1:

                                        ;;*************************************************************************
                                        ;VERIFY THAT DRIVE IS PRESENT THROUGH PORTS A & B

002634  113760  001216  000010                  MOVB    PORTA,RHCS2(R0)  ;SELECT PORT A
002642  013737  001216  001226                  MOV     PORTA,PTNBR  ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
002650  005760  000012                          TST     RHDS1(R0)       ;SEE IF DRIVE (PORT A) PRESENT
002654  005037  001236                          CLR     CKERR           ;CLEAR THE 'CHECK ERROR' INDICATOR
002660  016037  000010  001126                  MOV     RHCS2(R0),$BDDAT  ;GET CONTENTS OF RHCS2
002666  012737  000010  001122                  MOV     #RHCS2,$BDADR  ;FORM REGISTER ADDRESS OF ERROR MESSAGE
002674  060037  001122                          ADD     R0,$BDADR       ;ADD RH11 BASE ADDRESS
002700  005037  001124                          CLR     $GDDAT          ;WHAT REGISTER SHOULD BE
002704  013737  001126  001156                  MOV     $BDDAT,$TMP0    ;MOVE REGISTER CONTENTS TO '$TMP0'
002712  042737  167777  001156                  BIC     #^CNED,$TMP0  ;SAVE SPECIFIED BITS
002720  023737  001124  001156                  CMP     $GDDAT,$TMP0    ;COMPARE THE BITS
002726  001414                                  BEQ     64$             ;BR IF OK
002730  013737  001126  001166                  MOV     $BDDAT,$TMP4    ;COPY 'BAD DATA'
002736  042737  010000  001166                  BIC     #NED,$TMP4      ;CLEAR THE MASKED BITS
002744  053737  001166  001124                  BIS     $TMP4,$GDDAT    ;'OR' WITH GOOD DATA FOR TYPEOUT
002752  104001                                  ERROR   1               ;TYPE MESSAGE 1
002754  005137  001236                          COM     CKERR           ;SET THE REGISTER COMPARE ERROR INDICATOR
002760  000240                          64$:    NOP
002762  005737  001236                          TST     CKERR           ;WAS 'NED' SET ?
002766  001403                                  BEQ     .+10            ;BR IF NOT
002770  012760  000040  000010                  MOV     #CLR,RHCS2(R0)  ;ISSUE MASSBUS INIT TO CLEAR 'NED'
002776  113760  001220  000010                  MOVB    PORTB,RHCS2(R0)  ;SELECT PORT B
```

```
  (2)  003004  013737  001220  001226          MOV     PORTB,PTNBR  ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
  (1)  003012  005760  000012                   TST     RHDS1(R0)          ;SEE IF DRIVE (PORT B) PRESENT
  (2)  003016  005037  001236                   CLR     CKERR              ;CLEAR THE 'CHECK ERROR' INDICATOR
  (2)  003022  016037  000010  001126          MOV     RHCS2(R0),$BDDAT   ;GET CONTENTS OF RHCS2
  (2)  003030  012737  000010  001122          MOV     #RHCS2,$BDADR  ;FORM REGISTER ADDRESS OF ERROR MESSAGE
  (2)  003036  060037  001122                   ADD     R0,$BDADR          ;ADD RH11 BASE ADDRESS
  (2)  003042  005037  001124                   CLR     $GDDAT             ;WHAT REGISTER SHOULD BE
  (2)  003046  013737  001126  001156          MOV     $BDDAT,$TMP0       ;MOVE REGISTER CONTENTS TO '$TMP0'
  (2)  003054  042737  167777  001156          BIC     #^CNED,$TMP0   ;SAVE SPECIFIED BITS
  (2)  003062  023737  001124  001156          CMP     $GDDAT,$TMP0       ;COMPARE THE BITS
  (2)  003070  001414                           BEQ     65$                ;BR IF OK
  (2)  003072  013737  001126  001166          MOV     $BDDAT,$TMP4       ;COPY 'BAD DATA'
  (2)  003100  042737  010000  001166          BIC     #NED,$TMP4         ;CLEAR THE MASKED BITS
  (2)  003106  053737  001166  001124          BIS     $TMP4,$GDDAT       ;'OR' WITH GOOD DATA FOR TYPEOUT
  (2)  003114  104001                           ERROR   1                  ;TYPE MESSAGE 1
  (2)  003116  005137  001236                   COM     CKERR              ;SET THE REGISTER COMPARE ERROR INDICATOR
  (2)  003122  000240                   65$:    NOP
  (1)  003124  005737  001236                   TST     CKERR              ;WAS 'NED' SET ?
  (1)  003130  001403                           BEQ     .+10               ;BR IF NOT
  (1)  003132  012760  000040  000010          MOV     #CLR,RHCS2(R0)     ;ISSUE MASSBUS INIT TO CLEAR 'NED'
4886
4887                                    ;;*************************************************************************
4888                                    ;CONFIRM THAT DRIVE IS AN RP04 AND IS DUAL PORT
4889
4893  003140  113760  001216  000010          MOVB    PORTA,RHCS2(R0)  ;SELECT PORT A
  (2)  003146  013737  001216  001226          MOV     PORTA,PTNBR  ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
  (2)  003154  005037  001236                   CLR     CKERR              ;CLEAR THE 'CHECK ERROR' INDICATOR
  (2)  003160  016037  000026  001126          MOV     RHDT(R0),$BDDAT    ;GET CONTENTS OF RHDT
  (2)  003166  012737  000026  001122          MOV     #RHDT,$BDADR   ;FORM REGISTER ADDRESS OF ERROR MESSAGE
  (2)  003174  060037  001122                   ADD     R0,$BDADR          ;ADD RH11 BASE ADDRESS
  (2)  003200  012737  024020  001124          MOV     #24020,$GDDAT  ;WHAT REGISTER SHOULD BE
  (2)  003206  023737  001124  001126          CMP     $GDDAT,$BDDAT      ;IS THE REGISTER OK ?
  (2)  003214  001403                           BEQ     66$                ;BR IF OK
  (2)  003216  104002                           ERROR   2                  ;TYPE MESSAGE 2
  (2)  003220  005137  001236                   COM     CKERR              ;SET THE REGISTER COMPARE ERROR INDICATOR
  (2)  003224  000240                   66$:    NOP
  (2)  003226  113760  001220  000010          MOVB    PORTB,RHCS2(R0)  ;SELECT PORT B
  (2)  003234  013737  001220  001226          MOV     PORTB,PTNBR  ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
  (2)  003242  005037  001236                   CLR     CKERR              ;CLEAR THE 'CHECK ERROR' INDICATOR
  (2)  003246  016037  000026  001126          MOV     RHDT(R0),$BDDAT    ;GET CONTENTS OF RHDT
  (2)  003254  012737  000026  001122          MOV     #RHDT,$BDADR   ;FORM REGISTER ADDRESS OF ERROR MESSAGE
  (2)  003262  060037  001122                   ADD     R0,$BDADR          ;ADD RH11 BASE ADDRESS
  (2)  003266  012737  024020  001124          MOV     #24020,$GDDAT  ;WHAT REGISTER SHOULD BE
  (2)  003274  023737  001124  001126          CMP     $GDDAT,$BDDAT      ;IS THE REGISTER OK ?
  (2)  003302  001403                           BEQ     67$                ;BR IF OK
  (2)  003304  104002                           ERROR   2                  ;TYPE MESSAGE 2
  (2)  003306  005137  001236                   COM     CKERR              ;SET THE REGISTER COMPARE ERROR INDICATOR
  (2)  003312  000240                   67$:    NOP
4894
4895                                    ;;*************************************************************************
4896                                    ;VERIFY THROUGH BOTH PORTS THAT THE DRIVE IS ON LINE AND IN NEUTRAL
4897
4902  003314  113760  001216  000010          MOVB    PORTA,RHCS2(R0)  ;SELECT PORT A
  (2)  003322  013737  001216  001226          MOV     PORTA,PTNBR  ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
```

```
(2) 003330 005037 001236                    CLR     CKERR               ;CLEAR THE 'CHECK ERROR' INDICATOR
(2) 003334 016037 000012 001126             MOV     RHDS1(R0),$BDDAT    ;GET CONTENTS OF RHDS1
(2) 003342 012737 000012 001122             MOV     #RHDS1,$BDADR       ;FORM REGISTER ADDRESS OF ERROR MESSAGE
(2) 003350 060037 001122                    ADD     R0,$BDADR           ;ADD RH11 BASE ADDRESS
(2) 003354 012737 001000 001124             MOV     #PGM,$GDDAT         ;WHAT REGISTER SHOULD BE
(2) 003362 013737 001126 001156             MOV     $BDDAT,$TMP0        ;MOVE REGISTER CONTENTS TO '$TMP0'
(2) 003370 042737 176777 001156             BIC     #^CPGM,$TMP0        ;SAVE SPECIFIED BITS
(2) 003376 023737 001124 001156             CMP     $GDDAT,$TMP0        ;COMPARE THE BITS
(2) 003404 001414                           BEQ     68$                 ;BR IF OK
(2) 003406 013737 001126 001166             MOV     $BDDAT,$TMP4        ;COPY 'BAD DATA'
(2) 003414 042737 001000 001166             BIC     #PGM,$TMP4          ;CLEAR THE MASKED BITS
(2) 003422 053737 001166 001124             BIS     $TMP4,$GDDAT        ;'OR' WITH GOOD DATA FOR TYPEOUT
(2) 003430 104003                           ERROR   3                   ;TYPE MESSAGE 3
(2) 003432 005137 001236                    COM     CKERR               ;SET THE REGISTER COMPARE ERROR INDICATOR
(2) 003436 000240                    68$:   NOP
(2) 003440 005037 001236                    CLR     CKERR               ;CLEAR THE 'CHECK ERROR' INDICATOR
(2) 003444 016037 000012 001126             MOV     RHDS1(R0),$BDDAT    ;GET CONTENTS OF RHDS1
(2) 003452 012737 000012 001122             MOV     #RHDS1,$BDADR       ;FORM REGISTER ADDRESS OF ERROR MESSAGE
(2) 003460 060037 001122                    ADD     R0,$BDADR           ;ADD RH11 BASE ADDRESS
(2) 003464 012737 010600 001124             MOV     #MOL!DPR!DRY,$GDDAT ;WHAT REGISTER SHOULD BE
(2) 003472 013737 001126 001156             MOV     $BDDAT,$TMP0        ;MOVE REGISTER CONTENTS TO '$TMP0'
(2) 003500 042737 167177 001156             BIC     #^C10600,$TMP0      ;SAVE SPECIFIED BITS
(2) 003506 023737 001124 001156             CMP     $GDDAT,$TMP0        ;COMPARE THE BITS
(2) 003514 001414                           BEQ     69$                 ;BR IF OK
(2) 003516 013737 001126 001166             MOV     $BDDAT,$TMP4        ;COPY 'BAD DATA'
(2) 003524 042737 010600 001166             BIC     #10600,$TMP4        ;CLEAR THE MASKED BITS
(2) 003532 053737 001166 001124             BIS     $TMP4,$GDDAT        ;'OR' WITH GOOD DATA FOR TYPEOUT
(2) 003540 104004                           ERROR   4                   ;TYPE MESSAGE 4
(2) 003542 005137 001236                    COM     CKERR               ;SET THE REGISTER COMPARE ERROR INDICATOR
(2) 003546 000240                    69$:   NOP
(2) 003550 113760 001220 000010             MOVB    PORTB,RHCS2(R0)     ;SELECT PORT B
(2) 003556 013737 001220 001226             MOV     PORTB,PTNBR         ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
(2) 003564 005037 001236                    CLR     CKERR               ;CLEAR THE 'CHECK ERROR' INDICATOR
(2) 003570 016037 000012 001126             MOV     RHDS1(R0),$BDDAT    ;GET CONTENTS OF RHDS1
(2) 003576 012737 000012 001122             MOV     #RHDS1,$BDADR       ;FORM REGISTER ADDRESS OF ERROR MESSAGE
(2) 003604 060037 001122                    ADD     R0,$BDADR           ;ADD RH11 BASE ADDRESS
(2) 003610 012737 001000 001124             MOV     #PGM,$GDDAT         ;WHAT REGISTER SHOULD BE
(2) 003616 013737 001126 001156             MOV     $BDDAT,$TMP0        ;MOVE REGISTER CONTENTS TO '$TMP0'
(2) 003624 042737 176777 001156             BIC     #^CPGM,$TMP0        ;SAVE SPECIFIED BITS
(2) 003632 023737 001124 001156             CMP     $GDDAT,$TMP0        ;COMPARE THE BITS
(2) 003640 001414                           BEQ     70$                 ;BR IF OK
(2) 003642 013737 001126 001166             MOV     $BDDAT,$TMP4        ;COPY 'BAD DATA'
(2) 003650 042737 001000 001166             BIC     #PGM,$TMP4          ;CLEAR THE MASKED BITS
(2) 003656 053737 001166 001124             BIS     $TMP4,$GDDAT        ;'OR' WITH GOOD DATA FOR TYPEOUT
(2) 003664 104003                           ERROR   3                   ;TYPE MESSAGE 3
(2) 003666 005137 001236                    COM     CKERR               ;SET THE REGISTER COMPARE ERROR INDICATOR
(2) 003672 000240                    70$:   NOP
(2) 003674 005037 001236                    CLR     CKERR               ;CLEAR THE 'CHECK ERROR' INDICATOR
(2) 003700 016037 000012 001126             MOV     RHDS1(R0),$BDDAT    ;GET CONTENTS OF RHDS1
(2) 003706 012737 000012 001122             MOV     #RHDS1,$BDADR       ;FORM REGISTER ADDRESS OF ERROR MESSAGE
(2) 003714 060037 001122                    ADD     R0,$BDADR           ;ADD RH11 BASE ADDRESS
(2) 003720 012737 010600 001124             MOV     #MOL!DPR!DRY,$GDDAT ;WHAT REGISTER SHOULD BE
(2) 003726 013737 001126 001156             MOV     $BDDAT,$TMP0        ;MOVE REGISTER CONTENTS TO '$TMP0'
(2) 003734 042737 167177 001156             BIC     #^C10600,$TMP0      ;SAVE SPECIFIED BITS
```

```
(2)  003742  023737  001124  001156          CMP     $GDDAT,$TMP0    ;COMPARE THE BITS
(2)  003750  001414                          BEQ     71$             ;BR IF OK
(2)  003752  013737  001126  001166          MOV     $BDDAT,$TMP4    ;COPY 'BAD DATA'
(2)  003760  042737  010600  001166          BIC     #10600,$TMP4    ;CLEAR THE MASKED BITS
(2)  003766  053737  001166  001124          BIS     $TMP4,$GDDAT    ;'OR' WITH GOOD DATA FOR TYPEOUT
(2)  003774  104004                          ERROR   4               ;TYPE MESSAGE 4
(2)  003776  005137  001236                  COM     CKERR           ;SET THE REGISTER COMPARE ERROR INDICATOR
(2)  004002  000240                  71$:    NOP
4903
4904                                  ;;*************************************************************************
4905                                  ;VERIFY THAT DRIVE SERIAL NUMBER SEEN THROUGH BOTH PORTS IS THE SAME
4906
4907  004004  113760  001216  000010          MOVB    PORTA,RHCS2(R0) ;SELECT PORT A
4908  004012  016037  000030  001124          MOV     RHSN(R0),$GDDAT ;STORE THE PORT A SERIAL NUMBER
4909  004020  113760  001220  000010          MOVB    PORTB,RHCS2(R0) ;SELECT PORT B
4910  004026  016037  000030  001126          MOV     RHSN(R0),$BDDAT ;STORE THE PORT B SERIAL NUMBER
4911  004034  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;ARE THEY THE SAME ?
4912  004042  001406                          BEQ     1$              ;BR IF THEY ARE
4913  004044  104005                          ERROR   5               ;REPORT THE ERROR
4914  004046  032737  100000  177570          BIT     #SW15,SWR       ;HALT ON ERROR ?
4915  004054  001001                          BNE     1$              ;BR IF SET - PROGRAM HAS ALREADY HALTED
4916  004056  000000                          HALT                    ;HALT, POSSIBLE CABLE CONNECTION PROBLEM
4917  004060                       1$:
 (1)
 (1)                                          ;IF ERROR OCCURED, CHECK FOR LOOP ON TEST
 (1)
 (1)  004060  105737  001103                  TSTB    $ERFLG          ;DID AN ERROR OCCUR ?
 (3)  004064  001412                          BEQ     TST2            ;;BR IF NOT
 (1)  004066  032737  001000  177570          BIT     #SW09,SWR       ;SEE IF LOOP ON ERROR SET (SWR9=1)
 (3)  004074  001406                          BEQ     TST2            ;;BR IF NOT
 (1)  004076  105037  001103                  CLRB    $ERFLG          ;CLEAR THE ERROR FLAG
 (1)  004102  005037  001170                  CLR     $TIMES          ;CLEAR THE MAX ITERATION COUNT
 (1)  004106  000177  174776                  JMP     @$LPERR         ;GO TO THE LOOP ADDRESS
4918
4956
4957
 (5)                                  ;*************************************************************************
 (4)                                  ;*TEST 2         SET 'VV' FOR PORT A
 (5)                                  ;*
 (5)                                  ;*SET VOLUME VALID
 (5)                                  ;*
 (5)                                  ;*  A.  ISSUE A DRIVE CLEAR COMMAND THROUGH PORT A.
 (5)                                  ;*
 (5)                                  ;*  B.  ISSUE A READIN PRESET COMMAND THROUGH PORT A.  VERIFY
 (5)                                  ;*      THAT THE 'VV' BIT IS SET FOR PORT A.
 (5)                                  ;*
 (5)                                  ;*  C.  ISSUE A RELEASE COMMAND THROUGH PORT A.  VERIFY THAT
 (5)                                  ;*      THE DRIVE RETURNED TO NEUTRAL AND THAT NEITHER ATTENTION
 (5)                                  ;*      BIT IS SET.
 (5)                                  ;*
 (4)                                  ;*************************************************************************
 (4)  004112                          TST2:
 (4)  004112  000004                          SCOPE                   ;INITIALIZE THE SCOPE HANDLER
 (4)  004114  005737  001260                  TST     KYBCTL          ;PERFORMING ONLY SINGLE TESTS ?
```

```
(4)  004120  001406                          BEQ     2$              ;BR IF NOT
(4)  004122  100002                          BPL     1$              ;BR IF JUST ENTERED TEST
(4)  004124  000137  002262                  JMP     EXEC            ;RETURN & GET NEXT TEST NUMBER
(4)  004130  012737  177777  001260  1$:     MOV     #-1,KYBCTL      ;SET SINGLE TEST INDICATOR
(4)  004136  112737  000002  001102  2$:     MOVB    #2,$TSTNM       ;TEST NUMBER
(4)  004144  012737  004166  001106          MOV     #TEST2,$LPADR   ;LOAD LOOP ON TEST ADDRESS
(4)  004152  012737  004166  001110          MOV     #TEST2,$LPERR   ;LOAD LOOP ON ERROR ADDRESS
(2)  004160  012737  000001  001170          MOV     #1,$TIMES       ;;DO 1 ITERATION
(1)
(2)
(3)                                  ;;**************************************************************************
(2)                                  ;END OF 'SCOPE' SETUP - START OF MAIN TEST
(2)
(2)  004166                          TEST2:
(2)  004166  113760  001216  000010          MOVB    PORTA,RHCS2(R0)  ;SELECT PORT A
(2)  004174  013737  001216  001226          MOV     PORTA,PTNBR  ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
(1)
(2)                                  ;;**************************************************************************
(1)                                  ;SET VOLUME VALUE FOR PORT
(1)
(1)  004202  012760  000011  000000          MOV     #11,RHCS1(R0)   ;ISSUE A DRIVE CLEAR
(1)  004210  012760  000021  000000          MOV     #21,RHCS1(R0)   ;ISSUE A READIN PRESET
(1)  004216  012760  010000  000032          MOV     #FMT22,RHOF(R0)  ;SET FMT22
(1)
(2)                                  ;;**************************************************************************
(1)                                  ;VERIFY THAT THE DRIVE STATUS IS CORRECT
(1)
(2)  004224  005037  001236                  CLR     CKERR           ;CLEAR THE 'CHECK ERROR' INDICATOR
(2)  004230  016037  000012  001126          MOV     RHDS1(R0),$BDDAT  ;GET CONTENTS OF RHDS1
(2)  004236  012737  000012  001122          MOV     #RHDS1,$BDADR   ;FORM REGISTER ADDRESS OF ERROR MESSAGE
(2)  004244  060037  001122                  ADD     R0,$BDADR       ;ADD RH11 BASE ADDRESS
(2)  004250  012737  011700  001124          MOV     #MOL!PGM!DPR!DRY!VV,$GDDAT  ;WHAT REGISTER SHOULD BE
(2)  004256  013737  001126  001156          MOV     $BDDAT,$TMP0    ;MOVE REGISTER CONTENTS TO '$TMP0'
(2)  004264  042737  106077  001156          BIC     #^C71700,$TMP0  ;SAVE SPECIFIED BITS
(2)  004272  023737  001124  001156          CMP     $GDDAT,$TMP0    ;COMPARE THE BITS
(2)  004300  001414                          BEQ     64$             ;BR IF OK
(2)  004302  013737  001126  001166          MOV     $BDDAT,$TMP4    ;COPY 'BAD DATA'
(2)  004310  042737  071700  001166          BIC     #71700,$TMP4    ;CLEAR THE MASKED BITS
(2)  004316  053737  001166  001124          BIS     $TMP4,$GDDAT    ;'OR' WITH GOOD DATA FOR TYPEOUT
(2)  004324  104010                          ERROR   10              ;TYPE MESSAGE 10
(2)  004326  005137  001236                  COM     CKERR           ;SET THE REGISTER COMPARE ERROR INDICATOR
(2)  004332  000240                  64$:    NOP
(1)
(2)                                  ;;**************************************************************************
(2)
(2)                                  ;RELEASE THE DRIVE FROM PORT A
(2)
(3)  004334  113760  001216  000010          MOVB    PORTA,RHCS2(R0)  ;SELECT PORT A
(3)  004342  013737  001216  001226          MOV     PORTA,PTNBR  ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
(2)  004350  012760  000013  000000          MOV     #13,RHCS1(R0)   ;ISSUE RELEASE THROUGH PORT A
(3)
(3)                                  ;VERIFY THAT THE DRIVE IS IN NEUTRAL
(3)
(3)  004356  005037  001242                  CLR     RELERR          ;CLEAR THE 'RELEASE ERROR ' INDICATOR
```

```
(3)  004362  012737  000012  001122          MOV     #RHDS1,$BDADR   ;FORM THE ADDRESS OF RHDS1 FOR TYPEOUT
(3)  004370  060037  001122                  ADD     R0,$BDADR       ;ADD THE I/O BASE ADDRESS
(3)  004374  012737  011600  001124          MOV     #MOL!PGM!DPR!DRY,$GDDAT  ;COMPARISON CONSTANT
(3)  004402  113760  001216  000010          MOVB    PORTA,RHCS2(R0) ;SELECT PORT A.
(3)  004410  016037  000012  001162          MOV     RHDS1(R0),$TMP2 ;GET THE DRIVE STATUS REGISTER FROM PORT A.
(3)  004416  013737  001162  001156          MOV     $TMP2,$TMP0     ;COPY IT INTO '$TMP0'
(3)  004424  042737  100100  001156          BIC     #ATA!VV,$TMP0   ;CLEAR PORT DEPENDENT BITS FROM THE COPY
(3)  004432  113760  001220  000010          MOVB    PORTB,RHCS2(R0) ;SELECT PORT B.
(3)  004440  016037  000012  001164          MOV     RHDS1(R0),$TMP3 ;GET THE DRIVE STATUS REGISTER FROM PORT B.
(3)  004446  013737  001164  001160          MOV     $TMP3,$TMP1     ;COPY IT INTO '$TMP1'
(3)  004454  042737  100100  001160          BIC     #ATA!VV,$TMP1   ;CLEAR PORT DEPENDENT BITS FROM THE COPY
(3)  004462  023737  001156  001160          CMP     $TMP0,$TMP1     ;IS THE STATUS REGISTER THE SAME FROM BOTH PORTS ?
(3)  004470  001006                          BNE     65$             ;BR IF NOT
(3)  004472  005737  001156                  TST     $TMP0           ;REGISTERS ARE THE SAME: ARE THEY ZERO ?
(3)  004476  001037                          BNE     67$             ;BR IF NOT
(3)  004500  104034                          ERROR   34              ;REPORT DRIVE NOT IN NEUTRAL OR NOT SEIZED
(3)  004502  000137  004666                  JMP     69$             ;BYPASS THE REST OF THE CHECKS
(3)  004506  013737  001162  001126  65$:    MOV     $TMP2,$BDDAT    ;SET UP POSSIBLE BAD DATA FOR ERROR MESSAGE
(3)  004514  013737  001220  001226          MOV     PORTB,PTNBR     ;SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
(3)  004522  113760  001220  000010          MOVB    PORTB,RHCS2(R0) ;SELECT PORT B.
(3)  004530  005737  001156                  TST     $TMP0           ;SEE IF STATUS EQ 0 FROM PORT A.
(3)  004534  001414                          BEQ     66$             ;BR IF ZERO
(3)  004536  013737  001216  001226          MOV     PORTA,PTNBR     ;SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
(3)  004544  013737  001164  001126          MOV     $TMP3,$BDDAT    ;'BAD DATA' FOR ERROR TYPE OUT
(3)  004552  113760  001216  000010          MOVB    PORTA,RHCS2(R0) ;SELECT PORT A.
(3)  004560  005737  001160                  TST     $TMP1           ;SEE IF STATUS EQ ZERO FROM PORT B.
(3)  004564  001004                          BNE     67$             ;BR IF NOT
(3)  004566  012737  177777  001242  66$:    MOV     #-1,RELERR      ;SET 'RELEASE ERROR' INDICATOR
(3)  004574  104036                          ERROR   36              ;TYPE ERROR MESSAGE 36
(3)  004576  013737  001162  001126  67$:    MOV     $TMP2,$BDDAT    ;LOOK FOR BIT FAILURES WHEN RHDS1 READ
(3)  004604  013737  001216  001226          MOV     PORTA,PTNBR     ;CHANGE PORT NUMBER
(3)  004612  042737  100100  001162          BIC     #ATA!VV,$TMP2   ;DON'T CHECK ATTN BIT OR VV BIT
(3)  004620  023737  001124  001162          CMP     $GDDAT,$TMP2    ;ALL BITS OK ?
(3)  004626  001401                          BEQ     68$             ;BR IF OK FROM PORT A.
(3)  004630  104037                          ERROR   37              ;REPORT ERROR
(3)  004632  013737  001164  001126  68$:    MOV     $TMP3,$BDDAT    ;CHECK RHDS1 FOR BIT FAILURES - FROM PORT B.
(3)  004640  013737  001220  001226          MOV     PORTB,PTNBR     ;CHANGE PORT NUMBER
(3)  004646  042737  100100  001164          BIC     #ATA!VV,$TMP3   ;DON'T CHECK ATTN BIT OR VV BIT
(3)  004654  023737  001124  001164          CMP     $GDDAT,$TMP3    ;SEE IF READ OK FROM PORT B.
(3)  004662  001401                          BEQ     69$             ;BR IF OK
(3)  004664  104037                          ERROR   37              ;REPORT THE ERROR
(3)  004666  000240                  69$:    NOP
(2)
(2)                                          ;IF ERROR OCCURED, CHECK FOR LOOP ON TEST
(2)
(2)  004670  105737  001103                  TSTB    $ERFLG          ;DID AN ERROR OCCUR ?
(4)  004674  001412                          BEQ     TST3            ;;BR IF NOT
(2)  004676  032737  001000  177570          BIT     #SWD9,SWR       ;SEE IF LOOP ON ERROR SET (SWR9=1)
(4)  004704  001406                          BEQ     TST3            ;;BR IF NOT
(2)  004706  105037  001103                  CLRB    $ERFLG          ;CLEAR THE ERROR FLAG
(2)  004712  005037  001170                  CLR     $TIMES          ;CLEAR THE MAX ITERATION COUNT
(2)  004716  000177  174166                  JMP     @$LPERR         ;GO TO THE LOOP ADDRESS
4958
(5)                                          ;******************************************************************
```

```
(4)                                     ;*TEST 3        SET 'VV' FOR PORT B
(5)                                     ;*
(5)                                     ;*SET VOLUME VALID
(5)                                     ;*
(5)                                     ;*  A.  ISSUE A DRIVE CLEAR COMMAND THROUGH PORT B.
(5)                                     ;*
(5)                                     ;*  B.  ISSUE A READIN PRESET COMMAND THROUGH PORT B.  VERIFY
(5)                                     ;*      THAT THE 'VV' BIT IS SET FOR PORT B.
(5)                                     ;*
(5)                                     ;*  C.  ISSUE A RELEASE COMMAND THROUGH PORT B.  VERIFY THAT
(5)                                     ;*      THE DRIVE RETURNED TO NEUTRAL AND THAT NEITHER ATTENTION
(5)                                     ;*      BIT IS SET.
(4)                                     ;*
(4)                                     ;*****************************************************************
(4)  004722                             TST3:
(4)  004722  000004                             SCOPE                   ;INITIALIZE THE SCOPE HANDLER
(4)  004724  005737  001260                     TST     KYBCTL          ;PERFORMING ONLY SINGLE TESTS ?
(4)  004730  001406                             BEQ     2$              ;BR IF NOT
(4)  004732  100002                             BPL     1$              ;BR IF JUST ENTERED TEST
(4)  004734  000137  002262                     JMP     EXEC            ;RETURN & GET NEXT TEST NUMBER
(4)  004740  012737  177777  001260  1$:       MOV     #-1,KYBCTL      ;SET SINGLE TEST INDICATOR
(4)  004746  112737  000003  001102  2$:       MOVB    #3,$TSTNM       ;TEST NUMBER
(4)  004754  012737  004776  001106            MOV     #TEST3,$LPADR   ;LOAD LOOP ON TEST ADDRESS
(4)  004762  012737  004776  001110            MOV     #TEST3,$LPERR   ;LOAD LOOP ON ERROR ADDRESS
(2)  004770  012737  000001  001170            MOV     #1,$TIMES       ;;DO 1 ITERATION
(1)
(2)
(3)                                     ;;*****************************************************************
(2)                                     ;END OF 'SCOPE' SETUP - START OF MAIN TEST
(2)
(2)  004776                             TEST3:
(2)  004776  113760  001220  000010            MOVB    PORTB,RHCS2(R0)  ;SELECT PORT B
(2)  005004  013737  001220  001226            MOV     PORTB,PTNBR  ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
(1)
(2)                                     ;;*****************************************************************
(1)                                     ;SET VOLUME VALUE FOR PORT
(1)
(1)  005012  012760  000011  000000            MOV     #11,RHCS1(R0)   ;ISSUE A DRIVE CLEAR
(1)  005020  012760  000021  000000            MOV     #21,RHCS1(R0)   ;ISSUE A READIN PRESET
(1)  005026  012760  010000  000032            MOV     #FMT22,RHOF(R0)  ;SET FMT22
(1)
(2)                                     ;;*****************************************************************
(1)                                     ;VERIFY THAT THE DRIVE STATUS IS CORRECT
(1)
(2)  005034  005037  001236                     CLR     CKERR           ;CLEAR THE 'CHECK ERROR' INDICATOR
(2)  005040  016037  000012  001126            MOV     RHDS1(R0),$BDDAT  ;GET CONTENTS OF RHDS1
(2)  005046  012737  000012  001122            MOV     #RHDS1,$BDADR  ;FORM REGISTER ADDRESS OF ERROR MESSAGE
(2)  005054  060037  001122                     ADD     R0,$BDADR       ;ADD RH11 BASE ADDRESS
(2)  005060  012737  011700  001124            MOV     #MOL!PGM!DPR!DRY!VV,$GDDAT  ;WHAT REGISTER SHOULD BE
(2)  005066  013737  001126  001156            MOV     $BDDAT,$TMP0    ;MOVE REGISTER CONTENTS TO '$TMP0'
(2)  005074  042737  106077  001156            BIC     #^C71700,$TMP0  ;SAVE SPECIFIED BITS
(2)  005102  023737  001124  001156            CMP     $GDDAT,$TMP0    ;COMPARE THE BITS
(2)  005110  001414                             BEQ     64$             ;BR IF OK
(2)  005112  013737  001126  001166            MOV     $BDDAT,$TMP4    ;COPY 'BAD DATA'
```

```
(2)  005120  042737  071700  001166          BIC     #71700,$TMP4    ;CLEAR THE MASKED BITS
(2)  005126  053737  001166  001124          BIS     $TMP4,$GDDAT    ;'OR' WITH GOOD DATA FOR TYPEOUT
(2)  005134  104010                          ERROR   10              ;TYPE MESSAGE 10
(2)  005136  005137  001236                  COM     CKERR           ;SET THE REGISTER COMPARE ERROR INDICATOR
(2)  005142  000240                  64$:    NOP
(1)
(2)                                         ;;**************************************************************************
(2)
(2)                                         ;RELEASE THE DRIVE FROM PORT B
(2)
(3)  005144  113760  001220  000010          MOVB    PORTB,RHCS2(R0)  ;SELECT PORT B
(3)  005152  013737  001220  001226          MOV     PORTB,PTNBR  ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
(2)  005160  012760  000013  000000          MOV     #13,RHCS1(R0)   ;ISSUE RELEASE THROUGH PORT B
(3)
(3)                                         ;VERIFY THAT THE DRIVE IS IN NEUTRAL
(3)
(3)  005166  005037  001242                  CLR     RELERR          ;CLEAR THE 'RELEASE ERROR ' INDICATOR
(3)  005172  012737  000012  001122          MOV     #RHDS1,$BDADR   ;FORM THE ADDRESS OF RHDS1 FOR TYPEOUT
(3)  005200  060037  001122                  ADD     R0,$BDADR       ;ADD THE I/O BASE ADDRESS
(3)  005204  012737  011600  001124          MOV     #MOL!PGM!DPR!DRY,$GDDAT  ;COMPARISON CONSTANT
(3)  005212  113760  001216  000010          MOVB    PORTA,RHCS2(R0) ;SELECT PORT A.
(3)  005220  016037  000012  001162          MOV     RHDS1(R0),$TMP2 ;GET THE DRIVE STATUS REGISTER FROM PORT A.
(3)  005226  013737  001162  001156          MOV     $TMP2,$TMP0     ;COPY IT INTO '$TMP0'
(3)  005234  042737  100100  001156          BIC     #ATA!VV,$TMP0   ;CLEAR PORT DEPENDENT BITS FROM THE COPY
(3)  005242  113760  001220  000010          MOVB    PORTB,RHCS2(R0) ;SELECT PORT B.
(3)  005250  016037  000012  001164          MOV     RHDS1(R0),$TMP3 ;GET THE DRIVE STATUS REGISTER FROM PORT B.
(3)  005256  013737  001164  001160          MOV     $TMP3,$TMP1     ;COPY IT INTO '$TMP1'
(3)  005264  042737  100100  001160          BIC     #ATA!VV,$TMP1   ;CLEAR PORT DEPENDENT BITS FROM THE COPY
(3)  005272  023737  001156  001160          CMP     $TMP0,$TMP1     ;IS THE STATUS REGISTER THE SAME FROM BOTH PORTS ?
(3)  005300  001006                          BNE     65$             ;BR IF NOT
(3)  005302  005737  001156                  TST     $TMP0           ;REGISTERS ARE THE SAME: ARE THEY ZERO ?
(3)  005306  001037                          BNE     67$             ;BR IF NOT
(3)  005310  104034                          ERROR   34              ;REPORT DRIVE NOT IN NEUTRAL OR NOT SEIZED
(3)  005312  000137  005476                  JMP     69$             ;BYPASS THE REST OF THE CHECKS
(3)  005316  013737  001162  001126  65$:    MOV     $TMP2,$BDDAT    ;SET UP POSSIBLE BAD DATA FOR ERROR MESSAGE
(3)  005324  013737  001220  001226          MOV     PORTB,PTNBR     ;SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
(3)  005332  113760  001220  000010          MOVB    PORTB,RHCS2(R0) ;SELECT PORT B.
(3)  005340  005737  001156                  TST     $TMP0           ;SEE IF STATUS EQ 0 FROM PORT A.
(3)  005344  001414                          BEQ     66$             ;BR IF ZERO
(3)  005346  013737  001216  001226          MOV     PORTA,PTNBR     ;SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
(3)  005354  013737  001164  001126          MOV     $TMP3,$BDDAT    ;'BAD DATA' FOR ERROR TYPE OUT
(3)  005362  113760  001216  000010          MOVB    PORTA,RHCS2(R0) ;SELECT PORT A.
(3)  005370  005737  001160                  TST     $TMP1           ;SEE IF STATUS EQ ZERO FROM PORT B.
(3)  005374  001004                          BNE     67$             ;BR IF NOT
(3)  005376  012737  177777  001242  66$:    MOV     #-1,RELERR      ;SET 'RELEASE ERROR' INDICATOR
(3)  005404  104036                          ERROR   36              ;TYPE ERROR MESSAGE 36
(3)  005406  013737  001162  001126  67$:    MOV     $TMP2,$BDDAT    ;LOOK FOR BIT FAILURES WHEN RHDS1 READ
(3)  005414  013737  001216  001226          MOV     PORTA,PTNBR     ;CHANGE PORT NUMBER
(3)  005422  042737  100100  001162          BIC     #ATA!VV,$TMP2   ;DON'T CHECK ATTN BIT OR VV BIT
(3)  005430  023737  001124  001162          CMP     $GDDAT,$TMP2    ;ALL BITS OK ?
(3)  005436  001401                          BEQ     68$             ;BR IF OK FROM PORT A.
(3)  005440  104037                          ERROR   37              ;REPORT ERROR
(3)  005442  013737  001164  001126  68$:    MOV     $TMP3,$BDDAT    ;CHECK RHDS1 FOR BIT FAILURES - FROM PORT B.
(3)  005450  013737  001220  001226          MOV     PORTB,PTNBR     ;CHANGE PORT NUMBER
```

```
(3)  005456  042737  100100  001164          BIC     #ATA!VV,$TMP3   ;DON'T CHECK ATTN BIT OR VV BIT
(3)  005464  023737  001124  001164          CMP     $GDDAT,$TMP3    ;SEE IF READ OK FROM PORT B.
(3)  005472  001401                          BEQ     69$             ;BR IF OK
(3)  005474  104037                          ERROR   37              ;REPORT THE ERROR
(3)  005476  000240                  69$:    NOP
(2)
(2)                                          ;IF ERROR OCCURED, CHECK FOR LOOP ON TEST
(2)
(2)  005500  105737  001103                  TSTB    $ERFLG          ;DID AN ERROR OCCUR ?
(4)  005504  001412                          BEQ     TST4            ;;BR IF NOT
(2)  005506  032737  001000  177570          BIT     #SW09,SWR       ;SEE IF LOOP ON ERROR SET (SWR9=1)
(4)  005514  001406                          BEQ     TST4            ;;BR IF NOT
(2)  005516  105037  001103                  CLRB    $ERFLG          ;CLEAR THE ERROR FLAG
(2)  005522  005037  001170                  CLR     $TIMES          ;CLEAR THE MAX ITERATION COUNT
(2)  005526  000177  173356                  JMP     @$LPERR         ;GO TO THE LOOP ADDRESS
4959
4964
5020
(1)
(5)                                  ;*****************************************************************
(4)                                  ;*TEST 4        MEASURE THE TIMEOUT ONE-SHOT THROUGH PORT A
(5)                                  ;*
(5)                                  ;*MEASURE THE TIMEOUT ONE-SHOT VALUE THROUGH PORT A
(5)                                  ;*
(5)                                  ;*  A.  WRITE 0'S INTO RHDS1 THROUGH PORT A AND VERIFY THAT THE
(5)                                  ;*      DRIVE HAS BEEN SEIZED.
(5)                                  ;*
(5)                                  ;*  B.  WAIT FOR TIMEOUT TO OCCUR.  MEASURE THE DURATION OF THE TIMEOUT
(5)                                  ;*      ONE-SHOT AND SAVE THE VALUE FOR LATER USE.
(5)                                  ;*
(5)                                  ;*  C.  VERIFY THAT THE TIMEOUT OCCURED AND THAT THE DRIVE RETURNS
(5)                                  ;*      TO NEUTRAL
(5)                                  ;*
(4)                                  ;*****************************************************************
(4)  005532                          TST4:
(4)  005532  000004                          SCOPE                   ;INITIALIZE THE SCOPE HANDLER
(4)  005534  005737  001260                  TST     KYBCTL          ;PERFORMING ONLY SINGLE TESTS ?
(4)  005540  001406                          BEQ     2$              ;BR IF NOT
(4)  005542  100002                          BPL     1$              ;BR IF JUST ENTERED TEST
(4)  005544  000137  002262                  JMP     EXEC            ;RETURN & GET NEXT TEST NUMBER
(4)  005550  012737  177777  001260  1$:     MOV     #-1,KYBCTL      ;SET SINGLE TEST INDICATOR
(4)  005556  112737  000004  001102  2$:     MOVB    #4,$TSTNM       ;TEST NUMBER
(4)  005564  012737  005606  001106          MOV     #TEST4,$LPADR   ;LOAD LOOP ON TEST ADDRESS
(4)  005572  012737  005606  001110          MOV     #TEST4,$LPERR   ;LOAD LOOP ON ERROR ADDRESS
(2)  005600  012737  000001  001170          MOV     #1,$TIMES       ;;DO 1 ITERATION
(1)
(2)
(3)                                  ;;*************************************************************************
(2)                                  ;END OF 'SCOPE' SETUP - START OF MAIN TEST
(2)
(2)  005606                          TEST4:
(1)  005606  005037  001250                  CLR     TIMEA           ;CLEAR THE TIMEOUT VALUE STORAGE LOCATION
(1)  005612  005037  001252                  CLR     TIMEAP  ;CLEAR THE + 25% TOLERANCE LOCATION
(2)
```

```
(3)                                     ;;*************************************************************************
(2)                                     ;START THE TIMER
(2)
(2)  005616  005037  001244                     CLR     TIME            ;CLEAR THE ELAPSED TIME COUNTER
(2)  005622  012737  003720  001246             MOV     #2000.,WATCH    ;SET WATCH TO 2000 MS
(1)
(2)                                     ;;*************************************************************************
(2)
(2)                                             ;SEIZE THE DRIVE THROUGH PORT A
(2)
(2)  005630  113760  001216  000010             MOVB    PORTA,RHCS2(R0) ;SELECT PORT A
(2)  005636  013737  001216  001230             MOV     PORTA,SEIZPT  ;STORE SEIZING PORT'S ADDRESS
(2)  005644  005060  000012                     CLR     RHDS1(R0)       ;WRITE RHDS1
(3)  005650  113760  001220  000010             MOVB    PORTB,RHCS2(R0)  ;SELECT PORT B
(2)  005656  013737  001220  001226             MOV     PORTB,PTNBR  ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
(2)  005664  013737  001220  001232             MOV     PORTB,OPPRT     ;'OPPOSITE' PORT ADDRESS
(2)  005672  016037  000012  001126             MOV     RHDS1(R0),$BDDAT  ;SEE IF DRIVE SEIZED BY PORT A
(2)  005700  010037  001122                     MOV     R0,$BDADR       ;RH11 BASE ADDRESS
(2)  005704  062737  000012  001122             ADD     #RHDS1,$BDADR   ;GENERATE BAD REGISTER ADDRESS
(2)  005712  005037  001124                     CLR     $GDDAT          ;REGISTER SHOULD BE ZERO
(2)  005716  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;IS THE REGISTER ZERO
(2)  005724  001403                             BEQ     .+10            ;BR IF IT IS
(2)  005726  104030                             ERROR   30              ;REPORT THE ERROR
(2)  005730  000137  006442                     JMP     4$              ;BYPASS REST OF THE SUBTEST
(3)  005734  113760  001216  000010             MOVB    PORTA,RHCS2(R0)  ;SELECT PORT A
(2)  005742  013737  001216  001226             MOV     PORTA,PTNBR  ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
(2)  005750  016037  000012  001126             MOV     RHDS1(R0),$BDDAT  ;SEE IF SEIZING PORT SEES CORRECT STATUS
(2)  005756  012737  011700  001124             MOV     #MOL!PGM!DPR!DRY!VV,$GDDAT  ;EXPECTED STATUS
(2)  005764  013737  001124  001160             MOV     $GDDAT,$TMP1  ;USE GOOD DATA AS A MASK
(2)  005772  005137  001160                     COM     $TMP1           ;COMPLEMENT THE EXPECTED STATUS
(2)  005776  013737  001126  001156             MOV     $BDDAT,$TMP0    ;SAVE THE ACTUAL STATUS
(2)  006004  043737  001160  001156             BIC     $TMP1,$TMP0     ;CLEAR UNWANTED BITS
(2)  006012  023737  001124  001156             CMP     $GDDAT,$TMP0    ;ARE THE EXPECTED STATUS BITS SET ?
(2)  006020  001401                             BEQ     .+4             ;BR IF THEY ARE
(2)  006022  104031                             ERROR   31                      ;REPORT THE ERROR
(1)
(2)                                     ;;*************************************************************************
(1)                                     ;WAIT FOR PORT A TO TIMEOUT
(1)
(2)  006024  113760  001220  000010             MOVB    PORTB,RHCS2(R0)  ;SELECT PORT B
(2)  006032  013737  001220  001226             MOV     PORTB,PTNBR  ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
(1)  006040  005760  000012          1$:        TST     RHDS1(R0)       ;WAIT FOR THE DRIVE TO TIMEOUT
(1)  006044  001006                             BNE     2$              ;BR WHEN TIMEOUT OCCURS
(1)  006046  005737  001246                     TST     WATCH           ;CHECK WATCH
(1)  006052  001372                             BNE     1$              ;BR IF NOT ZERO
(1)  006054  104006                             ERROR   6               ;NO TIMEOUT WITHIN 2 SECONDS
(1)  006056  000137  006114                     JMP     3$              ;BYPASS THE REST OF THE TEST
(1)  006062  013737  001244  001250  2$:        MOV     TIME,TIMEA      ;SAVE THE ELAPSED TIME FOR PORT A
(1)  006070  004537  017562                     JSR     R5,TOLER        ;CALCULATE THE TOLERANCE
(1)  006074  001250                             .WORD   TIMEA           ;TIMEOUT VALUE FOR PORT A
(1)  006076  012637  001252                     MOV     (SP)+,TIMEAP    ;+25% TOLERANCE
(1)
(2)                                     ;;*************************************************************************
(1)                                     ;VERIFY THAT THE TIMEOUT ONE-SHOT VALUE IS AT LEAST 500 MS
```

```
(1)
(1)  006102  023727  001250  000764          CMP     TIMEA,#500.     ;IS TIMEOUT VALUE AT LEAST 500 MS ?
(1)  006110  103001                           BHIS    3$              ;BR IF IT IS
(1)  006112  104007                           ERROR   7               ;TIMEOUT LESS THAN 500 MS
(1)
(2)                                  ;;*************************************************************************
(1)                                  ;VERIFY THAT THE DRIVE RETURNED TO NEUTRAL AFTER PORT A TIMED OUT
(1)
(2)  006114                          3$:
(2)
(2)                                          ;VERIFY THAT THE DRIVE IS IN NEUTRAL
(2)
(2)  006114  005037  001242                  CLR     RELERR          ;CLEAR THE 'RELEASE ERROR ' INDICATOR
(2)  006120  012737  000012  001122          MOV     #RHDS1,$BDADR   ;FORM THE ADDRESS OF RHDS1 FOR TYPEOUT
(2)  006126  060037  001122                  ADD     R0,$BDADR       ;ADD THE I/O BASE ADDRESS
(2)  006132  012737  011700  001124          MOV     #MOL!PGM!DPR!DRY!VV,$GDDAT  ;COMPARISON CONSTANT
(2)  006140  113760  001216  000010          MOVB    PORTA,RHCS2(R0) ;SELECT PORT A.
(2)  006146  016037  000012  001162          MOV     RHDS1(R0),$TMP2 ;GET THE DRIVE STATUS REGISTER FROM PORT A.
(2)  006154  013737  001162  001156          MOV     $TMP2,$TMP0     ;COPY IT INTO '$TMP0'
(2)  006162  042737  100100  001156          BIC     #ATA!VV,$TMP0   ;CLEAR PORT DEPENDENT BITS FROM THE COPY
(2)  006170  113760  001220  000010          MOVB    PORTB,RHCS2(R0) ;SELECT PORT B.
(2)  006176  016037  000012  001164          MOV     RHDS1(R0),$TMP3 ;GET THE DRIVE STATUS REGISTER FROM PORT B.
(2)  006204  013737  001164  001160          MOV     $TMP3,$TMP1     ;COPY IT INTO '$TMP1'
(2)  006212  042737  100100  001160          BIC     #ATA!VV,$TMP1   ;CLEAR PORT DEPENDENT BITS FROM THE COPY
(2)  006220  023737  001156  001160          CMP     $TMP0,$TMP1     ;IS THE STATUS REGISTER THE SAME FROM BOTH PORTS ?
(2)  006226  001006                          BNE     64$             ;BR IF NOT
(2)  006230  005737  001156                  TST     $TMP0           ;REGISTERS ARE THE SAME: ARE THEY ZERO ?
(2)  006234  001045                          BNE     66$             ;BR IF NOT
(2)  006236  104034                          ERROR   34              ;REPORT DRIVE NOT IN NEUTRAL OR NOT SEIZED
(2)  006240  000137  006440                  JMP     68$             ;BYPASS THE REST OF THE CHECKS
(2)  006244  013737  001162  001126  64$:    MOV     $TMP2,$BDDAT    ;SET UP POSSIBLE BAD DATA FOR ERROR MESSAGE
(2)  006252  013737  001220  001226          MOV     PORTB,PTNBR     ;SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
(2)  006260  113760  001220  000010          MOVB    PORTB,RHCS2(R0) ;SELECT PORT B.
(2)  006266  005737  001156                  TST     $TMP0           ;SEE IF STATUS EQ 0 FROM PORT A.
(2)  006272  001414                          BEQ     65$             ;BR IF ZERO
(2)  006274  013737  001216  001226          MOV     PORTA,PTNBR     ;SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
(2)  006302  013737  001164  001126          MOV     $TMP3,$BDDAT    ;'BAD DATA' FOR ERROR TYPE OUT
(2)  006310  113760  001216  000010          MOVB    PORTA,RHCS2(R0) ;SELECT PORT A.
(2)  006316  005737  001160                  TST     $TMP1           ;SEE IF STATUS EQ ZERO FROM PORT B.
(2)  006322  001012                          BNE     66$             ;BR IF NOT
(2)  006324  012737  177777  001242  65$:    MOV     #-1,RELERR      ;SET 'RELEASE ERROR' INDICATOR
(2)  006332  012760  000011  000000          MOV     #11,RHCS1(R0)   ;CLEAR THE DRIVE
(2)  006340  012760  000013  000000          MOV     #13,RHCS1(R0)   ;RELEASE THE DRIVE
(2)  006346  104035                          ERROR   35              ;TYPE ERROR MESSAGE 35
(2)  006350  013737  001162  001126  66$:    MOV     $TMP2,$BDDAT    ;LOOK FOR BIT FAILURES WHEN RHDS1 READ
(2)  006356  013737  001216  001226          MOV     PORTA,PTNBR     ;CHANGE PORT NUMBER
(2)  006364  042737  100000  001162          BIC     #ATA,$TMP2      ;DON'T CHECK THE ATTN BIT
(2)  006372  023737  001124  001162          CMP     $GDDAT,$TMP2    ;ALL BITS OK ?
(2)  006400  001401                          BEQ     67$             ;BR IF OK FROM PORT A.
(2)  006402  104037                          ERROR   37              ;REPORT ERROR
(2)  006404  013737  001164  001126  67$:    MOV     $TMP3,$BDDAT    ;CHECK RHDS1 FOR BIT FAILURES - FROM PORT B.
(2)  006412  013737  001220  001226          MOV     PORTB,PTNBR     ;CHANGE PORT NUMBER
(2)  006420  042737  100000  001164          BIC     #ATA,$TMP3      ;DON'T CHECK THE ATTN BIT
(2)  006426  023737  001124  001164          CMP     $GDDAT,$TMP3    ;SEE IF READ OK FROM PORT B.
```

```
(2)  006434  001401                          BEQ     68$             ;BR IF OK
(2)  006436  104037                          ERROR   37              ;REPORT THE ERROR
(2)  006440  000240                  68$:    NOP
(2)  006442                          4$:
(2)
(2)                                          ;IF ERROR OCCURED, CHECK FOR LOOP ON TEST
(2)
(2)  006442  105737  001103                  TSTB    $ERFLG          ;DID AN ERROR OCCUR ?
(4)  006446  001412                          BEQ     TST5            ;;BR IF NOT
(2)  006450  032737  001000  177570          BIT     #SWD9,SWR       ;SEE IF LOOP ON ERROR SET (SWR9=1)
(4)  006456  001406                          BEQ     TST5            ;;BR IF NOT
(2)  006460  105037  001103                  CLRB    $ERFLG          ;CLEAR THE ERROR FLAG
(2)  006464  005037  001170                  CLR     $TIMES          ;CLEAR THE MAX ITERATION COUNT
(2)  006470  000177  172414                  JMP     @$LPERR         ;GO TO THE LOOP ADDRESS
5021
(1)
(5)                                  ;*****************************************************************
(4)                                  ;*TEST 5        MEASURE THE TIMEOUT ONE-SHOT THROUGH PORT B
(5)                                  ;*
(5)                                  ;*MEASURE THE TIMEOUT ONE-SHOT VALUE THROUGH PORT B
(5)                                  ;*
(5)                                  ;*  A.  WRITE 0'S INTO RHDS1 THROUGH PORT B AND VERIFY THAT THE
(5)                                  ;*      DRIVE HAS BEEN SEIZED.
(5)                                  ;*
(5)                                  ;*  B.  WAIT FOR TIMEOUT TO OCCUR.  MEASURE THE DURATION OF THE TIMEOUT
(5)                                  ;*      ONE-SHOT AND SAVE THE VALUE FOR LATER USE.
(5)                                  ;*
(5)                                  ;*  C.  VERIFY THAT THE TIMEOUT OCCURED AND THAT THE DRIVE RETURNS
(5)                                  ;*      TO NEUTRAL
(5)                                  ;*
(4)                                  ;*****************************************************************
(4)  006474                          TST5:
(4)  006474  000004                          SCOPE                   ;INITIALIZE THE SCOPE HANDLER
(4)  006476  005737  001260                  TST     KYBCTL          ;PERFORMING ONLY SINGLE TESTS ?
(4)  006502  001406                          BEQ     2$              ;BR IF NOT
(4)  006504  100002                          BPL     1$              ;BR IF JUST ENTERED TEST
(4)  006506  000137  002262                  JMP     EXEC            ;RETURN & GET NEXT TEST NUMBER
(4)  006512  012737  177777  001260  1$:     MOV     #-1,KYBCTL      ;SET SINGLE TEST INDICATOR
(4)  006520  112737  000005  001102  2$:     MOVB    #5,$TSTNM       ;TEST NUMBER
(4)  006526  012737  006550  001106          MOV     #TEST5,$LPADR   ;LOAD LOOP ON TEST ADDRESS
(4)  006534  012737  006550  001110          MOV     #TEST5,$LPERR   ;LOAD LOOP ON ERROR ADDRESS
(2)  006542  012737  000001  001170          MOV     #1,$TIMES       ;;DO 1 ITERATION
(1)
(2)
(3)                                  ;;*************************************************************************
(2)                                  ;END OF 'SCOPE' SETUP - START OF MAIN TEST
(2)
(2)  006550                          TEST5:
(1)  006550  005037  001254                  CLR     TIMEB                   ;CLEAR THE TIMEOUT VALUE STORAGE LOCATION
(1)  006554  005037  001256                  CLR     TIMEBP  ;CLEAR THE + 25% TOLERANCE LOCATION
(2)
(3)                                  ;;*************************************************************************
(2)                                  ;START THE TIMER
(2)
```

```
(2)  006560  005037  001244                    CLR     TIME              ;CLEAR THE ELAPSED TIME COUNTER
(2)  006564  012737  003720  001246            MOV     #2000.,WATCH      ;SET WATCH TO 2000 MS
(1)
(2)                                        ;;*************************************************************************
(2)
(2)                                        ;SEIZE THE DRIVE THROUGH PORT B
(2)
(2)  006572  113760  001220  000010            MOVB    PORTB,RHCS2(R0) ;SELECT PORT B
(2)  006600  013737  001220  001230            MOV     PORTB,SEIZPT  ;STORE SEIZING PORT'S ADDRESS
(2)  006606  005060  000012                    CLR     RHDS1(R0)       ;WRITE RHDS1
(3)  006612  113760  001216  000010            MOVB    PORTA,RHCS2(R0)  ;SELECT PORT A
(3)  006620  013737  001216  001226            MOV     PORTA,PTNBR  ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
(2)  006626  013737  001216  001232            MOV     PORTA,OPPRT     ;'OPPOSITE' PORT ADDRESS
(2)  006634  016037  000012  001126            MOV     RHDS1(R0),$BDDAT  ;SEE IF DRIVE SEIZED BY PORT B
(2)  006642  010037  001122                    MOV     R0,$BDADR       ;RH11 BASE ADDRESS
(2)  006646  062737  000012  001122            ADD     #RHDS1,$BDADR   ;GENERATE BAD REGISTER ADDRESS
(2)  006654  005037  001124                    CLR     $GDDAT          ;REGISTER SHOULD BE ZERO
(2)  006660  023737  001124  001126            CMP     $GDDAT,$BDDAT   ;IS THE REGISTER ZERO
(2)  006666  001403                            BEQ     .+10            ;BR IF IT IS
(2)  006670  104030                            ERROR   30              ;REPORT THE ERROR
(2)  006672  000137  007404                    JMP     4$              ;BYPASS REST OF THE SUBTEST
(3)  006676  113760  001220  000010            MOVB    PORTB,RHCS2(R0)  ;SELECT PORT B
(3)  006704  013737  001220  001226            MOV     PORTB,PTNBR  ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
(2)  006712  016037  000012  001126            MOV     RHDS1(R0),$BDDAT  ;SEE IF SEIZING PORT SEES CORRECT STATUS
(2)  006720  012737  01170_  001124            MOV     #MOL!PGM!DPR!DRY!VV,$GDDAT  ;EXPECTED STATUS
(2)  006726  013737  001124  001160            MOV     $GDDAT,$TMP1  ;USE GOOD DATA AS A MASK
(2)  006734  005137  001160                    COM     $TMP1           ;COMPLEMENT THE EXPECTED STATUS
(2)  006740  013737  001126  001156            MOV     $BDDAT,$TMP0    ;SAVE THE ACTUAL STATUS
(2)  006746  043737  001160  001156            BIC     $TMP1,$TMP0     ;CLEAR UNWANTED BITS
(2)  006754  023737  001124  001156            CMP     $GDDAT,$TMP0    ;ARE THE EXPECTED STATUS BITS SET ?
(2)  006762  001401                            BEQ     .+4             ;BR IF THEY ARE
(2)  006764  104031                            ERROR   31                      ;REPORT THE ERROR
(1)
(2)                                        ;;*************************************************************************
(1)                                        ;WAIT FOR PORT B TO TIMEOUT
(1)
(2)  006766  113760  001216  000010                MOVB    PORTA,RHCS2(R0)  ;SELECT PORT A
(2)  006774  013737  001216  001226                MOV     PORTA,PTNBR  ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
(1)  007002  005760  000012            1$:     TST     RHDS1(R0)       ;WAIT FOR THE DRIVE TO TIMEOUT
(1)  007006  001006                            BNE     2$              ;BR WHEN TIMEOUT OCCURS
(1)  007010  005737  001246                    TST     WATCH           ;CHECK WATCH
(1)  007014  001372                            BNE     1$              ;BR IF NOT ZERO
(1)  007016  104006                            ERROR   6               ;NO TIMEOUT WITHIN 2 SECONDS
(1)  007020  000137  007056                    JMP     3$              ;BYPASS THE REST OF THE TEST
(1)  007024  013737  001244  001254    2$:     MOV     TIME,TIMEB      ;SAVE THE ELAPSED TIME FOR PORT B
(1)  007032  004537  017562                    JSR     R5,TOLER        ;CALCULATE THE TOLERANCE
(1)  007036  001254                            .WORD   TIMEB           ;TIMEOUT VALUE FOR PORT B
(1)  007040  012637  001256                    MOV     (SP)+,TIMEBP    ;+25% TOLERANCE
(1)
(2)                                        ;;*************************************************************************
(1)                                        ;VERIFY THAT THE TIMEOUT ONE-SHOT VALUE IS AT LEAST 500 MS
(1)
(1)  007044  023727  001254  000764            CMP     TIMEB,#500.     ;IS TIMEOUT VALUE AT LEAST 500 MS ?
(1)  007052  103001                            BHIS    3$              ;BR IF IT IS
```

```
(1)  007054  104007                          ERROR   7               ;TIMEOUT LESS THAN 500 MS
(1)
(2)                                ;;*************************************************************************
(1)                                ;VERIFY THAT THE DRIVE RETURNED TO NEUTRAL AFTER PORT B TIMED OUT
(1)
(2)  007056                        3$:
(2)
(2)                                ;VERIFY THAT THE DRIVE IS IN NEUTRAL
(2)
(2)  007056  005037  001242                   CLR     RELERR          ;CLEAR THE 'RELEASE ERROR ' INDICATOR
(2)  007062  012737  000012  001122           MOV     #RHDS1,$BDADR   ;FORM THE ADDRESS OF RHDS1 FOR TYPEOUT
(2)  007070  060037  001122                   ADD     R0,$BDADR       ;ADD THE I/O BASE ADDRESS
(2)  007074  012737  011700  001124           MOV     #MOL!PGM!DPR!DRY!VV,$GDDAT  ;COMPARISON CONSTANT
(2)  007102  113760  001216  000010           MOVB    PORTA,RHCS2(R0) ;SELECT PORT A.
(2)  007110  016037  000012  001162           MOV     RHDS1(R0),$TMP2 ;GET THE DRIVE STATUS REGISTER FROM PORT A.
(2)  007116  013737  001162  001156           MOV     $TMP2,$TMP0     ;COPY IT INTO '$TMP0'
(2)  007124  042737  100100  001156           BIC     #ATA!VV,$TMP0   ;CLEAR PORT DEPENDENT BITS FROM THE COPY
(2)  007132  113760  001220  000010           MOVB    PORTB,RHCS2(R0) ;SELECT PORT B.
(2)  007140  016037  000012  001164           MOV     RHDS1(R0),$TMP3 ;GET THE DRIVE STATUS REGISTER FROM PORT B.
(2)  007146  013737  001164  001160           MOV     $TMP3,$TMP1     ;COPY IT INTO '$TMP1'
(2)  007154  042737  100100  001160           BIC     #ATA!VV,$TMP1   ;CLEAR PORT DEPENDENT BITS FROM THE COPY
(2)  007162  023737  001156  001160           CMP     $TMP0,$TMP1     ;IS THE STATUS REGISTER THE SAME FROM BOTH PORTS ?
(2)  007170  001006                           BNE     64$             ;BR IF NOT
(2)  007172  005737  001156                   TST     $TMP0           ;REGISTERS ARE THE SAME: ARE THEY ZERO ?
(2)  007176  001045                           BNE     66$             ;BR IF NOT
(2)  007200  104034                           ERROR   34              ;REPORT DRIVE NOT IN NEUTRAL OR NOT SEIZED
(2)  007202  000137  007402                   JMP     68$             ;BYPASS THE REST OF THE CHECKS
(2)  007206  013737  001162  001126  64$:     MOV     $TMP2,$BDDAT    ;SET UP POSSIBLE BAD DATA FOR ERROR MESSAGE
(2)  007214  013737  001220  001226           MOV     PORTB,PTNBR     ;SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
(2)  007222  113760  001220  000010           MOVB    PORTB,RHCS2(R0) ;SELECT PORT B.
(2)  007230  005737  001156                   TST     $TMP0           ;SEE IF STATUS EQ 0 FROM PORT A.
(2)  007234  001414                           BEQ     65$             ;BR IF ZERO
(2)  007236  013737  001216  001226           MOV     PORTA,PTNBR     ;SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
(2)  007244  013737  001164  001126           MOV     $TMP3,$BDDAT    ;'BAD DATA' FOR ERROR TYPE OUT
(2)  007252  113760  001216  000010           MOVB    PORTA,RHCS2(R0) ;SELECT PORT A.
(2)  007260  005737  001160                   TST     $TMP1           ;SEE IF STATUS EQ ZERO FROM PORT B.
(2)  007264  001012                           BNE     66$             ;BR IF NOT
(2)  007266  012737  177777  001242  65$:     MOV     #-1,RELERR      ;SET 'RELEASE ERROR' INDICATOR
(2)  007274  012760  000011  000000           MOV     #11,RHCS1(R0)   ;CLEAR THE DRIVE
(2)  007302  012760  000013  000000           MOV     #13,RHCS1(R0)   ;RELEASE THE DRIVE
(2)  007310  104035                           ERROR   35              ;TYPE ERROR MESSAGE 35
(2)  007312  013737  001162  001126  66$:     MOV     $TMP2,$BDDAT    ;LOOK FOR BIT FAILURES WHEN RHDS1 READ
(2)  007320  013737  001216  001226           MOV     PORTA,PTNBR     ;CHANGE PORT NUMBER
(2)  007326  042737  100000  001162           BIC     #ATA,$TMP2      ;DON'T CHECK THE ATTN BIT
(2)  007334  023737  001124  001162           CMP     $GDDAT,$TMP2    ;ALL BITS OK ?
(2)  007342  001401                           BEQ     67$             ;BR IF OK FROM PORT A.
(2)  007344  104037                           ERROR   37              ;REPORT ERROR
(2)  007346  013737  001164  001126  67$:     MOV     $TMP3,$BDDAT    ;CHECK RHDS1 FOR BIT FAILURES - FROM PORT B.
(2)  007354  013737  001220  001226           MOV     PORTB,PTNBR     ;CHANGE PORT NUMBER
(2)  007362  042737  100000  001164           BIC     #ATA,$TMP3      ;DON'T CHECK THE ATTN BIT
(2)  007370  023737  001124  001164           CMP     $GDDAT,$TMP3    ;SEE IF READ OK FROM PORT B.
(2)  007376  001401                           BEQ     68$             ;BR IF OK
(2)  007400  104037                           ERROR   37              ;REPORT THE ERROR
(2)  007402  000240                68$:       NOP
```

```
(2)  007404                          4$:
(2)
(2)                                         ;IF ERROR OCCURED, CHECK FOR LOOP ON TEST
(2)
(2)  007404  105737  001103                 TSTB    $ERFLG          ;DID AN ERROR OCCUR ?
(4)  007410  001412                         BEQ     TST6            ;;BR IF NOT
(2)  007412  032737  001000  177570         BIT     #SW09,SWR       ;SEE IF LOOP ON ERROR SET (SWR9=1)
(4)  007420  001406                         BEQ     TST6            ;;BR IF NOT
(2)  007422  105037  001103                 CLRB    $ERFLG          ;CLEAR THE ERROR FLAG
(2)  007426  005037  001170                 CLR     $TIMES          ;CLEAR THE MAX ITERATION COUNT
(2)  007432  000177  171452                 JMP     @$LPERR         ;GO TO THE LOOP ADDRESS
5022
5123
(5)                                 ;*****************************************************************
(4)                                 ;*TEST 6        TEST UNLOAD COMMAND THROUGH PORT A
(5)                                 ;*
(5)                                 ;*VERIFY THAT THE UNLOAD COMMAND FUNCTIONS PROPERLY AND THAT A PORT
(5)                                 ;*      TIMEOUT WILL NOT OCCUR WHILE THE 'GO' BIT IS SET.
(5)                                 ;*
(5)                                 ;*  A.  ISSUE AN UNLOAD COMMAND THROUGH PORT A; VERIFY THAT THE
(5)                                 ;*      DRIVE IS SEIZED.
(5)                                 ;*
(5)                                 ;*  B.  WAIT THE MEASURED TIMEOUT INTERVAL + 25%; VERIFY THAT THE DRIVE
(5)                                 ;*      DOES NOT TIME OUT.  VERIFY THAT THE 'GO' BIT IS STILL SET AND
(5)                                 ;*      THAT 'DRY' AND 'PIP' ARE NOT SET.
(5)                                 ;*
(5)                                 ;*  C.  REQUEST THAT THE OPERATOR PRESS THE 'STANDBY' BUTTON ON THE DRIVE.
(5)                                 ;*
(5)                                 ;*  D.  WHEN THE DRIVE CYCLES UP, VERIFY THAT THE DRIVE IS STILL SEIZED
(5)                                 ;*      BY PORT A, THAT THE 'VV' BIT FOR PORT A IS RESET, THAT THE
(5)                                 ;*      ATTENTION BIT FOR PORT A IS SET, AND THAT THE ATTENTION BIT
(5)                                 ;*      FOR PORT B IS NOT SET.
(5)                                 ;*
(5)                                 ;*  E.  WAIT FOR THE PORT TIMEOUT TO RELEASE THE DRIVE.  WHEN THE TIMEOUT
(5)                                 ;*      OCCURS, VERIFY THAT THE DRIVE RETURNED TO NEUTRAL, THAT THE
(5)                                 ;*      ATTENTION BIT FOR PORT A IS STILL SET, AND THAT THE ATTENTION
(5)                                 ;*      BIT FOR PORT B IS NOT SET.
(5)                                 ;*
(5)                                 ;*  F.  VERIFY THAT THE 'VV' BIT FOR PORT B IS NOT SET.
(5)                                 ;*
(5)                                 ;*  G.  ISSUE A PACK ACKNOWLEDGE INSTRUCTION THROUGH BOTH PORTS.
(5)                                 ;*
(4)                                 ;*****************************************************************
(4)  007436                         TST6:
(4)  007436  000004                         SCOPE                   ;INITIALIZE THE SCOPE HANDLER
(4)  007440  005737  001260                 TST     KYBCTL          ;PERFORMING ONLY SINGLE TESTS ?
(4)  007444  001406                         BEQ     2$              ;BR IF NOT
(4)  007446  100002                         BPL     1$              ;BR IF JUST ENTERED TEST
(4)  007450  000137  002262                 JMP     EXEC            ;RETURN & GET NEXT TEST NUMBER
(4)  007454  012737  177777  001260  1$:    MOV     #-1,KYBCTL      ;SET SINGLE TEST INDICATOR
(4)  007462  112737  000006  001102  2$:    MOVB    #6,$TSTNM       ;TEST NUMBER
(4)  007470  012737  007512  001106         MOV     #TEST6,$LPADR   ;LOAD LOOP ON TEST ADDRESS
(4)  007476  012737  007512  001110         MOV     #TEST6,$LPERR   ;LOAD LOOP ON ERROR ADDRESS
(2)  007504  012737  000001  001170         MOV     #1,$TIMES       ;;DO 1 ITERATION
```

```
(1)
(2)
(3)                                     ;;*****************************************************************************
(2)                                     ;END OF 'SCOPE' SETUP - START OF MAIN TEST
(2)
(2)  007512                             TEST6:
(2)
(2)                                             ;CLEAR ATTENTION BITS FOR BOTH PORTS
(2)
(2)  007512  113760  001216  000010             MOVB    PORTA,RHCS2(R0) ;SELECT PORT #A
(2)  007520  005060  000012                     CLR     RHDS1(R0)       ;SEIZE THE DRIVE
(2)  007524  012760  000011  000000             MOV     #11,RHCS1(R0)   ;ISSUE DRIVE CLEAR
(2)  007532  012760  000013  000000             MOV     #13,RHCS1(R0)   ;RELEASE THE DRIVE
(2)  007540  113760  001220  000010             MOVB    PORTB,RHCS2(R0) ;SELECT PORT #B
(2)  007546  005060  000012                     CLR     RHDS1(R0)       ;SEIZE THE DRIVE THROUGH PORT 'B'
(2)  007552  012760  000011  000000             MOV     #11,RHCS1(R0)   ;ISSUE DRIVE CLEAR
(2)  007560  012760  000013  000000             MOV     #13,RHCS1(R0)   ;RELEASE THE DRIVE
(2)  007566  113760  001216  000010             MOVB    PORTA,RHCS2(R0)  ;SELECT PORT A
(2)  007574  013737  001216  001226             MOV     PORTA,PTNBR  ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
(1)  007602  013737  001216  001230             MOV     PORTA,SEIZPT    ;SEIZING PORT'S ADDRESS
(1)
(2)                                     ;;*****************************************************************************
(1)                                     ;DO AN UNLOAD COMMAND THROUGH PORT A
(1)
(1)  007610  012760  000003  000000             MOV     #3,RHCS1(R0)    ;ISSUE AN UNLOAD COMMAND THROUGH PORT A
(1)  007616  013737  001252  001246             MOV     TIMEAP,WATCH    ;TIMEOUT ONESHOT VALUE + 25%
(1)  007624  005737  001246             1$:     TST     WATCH           ;FINISHED TIMEOUT ?
(1)  007630  001375                             BNE     1$              ;BR IF NOT
(1)
(2)                                     ;;*****************************************************************************
(1)                                     ;IS THE STATUS OK ?
(1)
(2)  007632  005037  001236                     CLR     CKERR           ;CLEAR THE 'CHECK ERROR' INDICATOR
(2)  007636  016037  000000  001126             MOV     RHCS1(R0),$BDDAT  ;GET CONTENTS OF RHCS1
(2)  007644  012737  000000  001122             MOV     #RHCS1,$BDADR   ;FORM REGISTER ADDRESS OF ERROR MESSAGE
(2)  007652  060037  001122                     ADD     R0,$BDADR       ;ADD RH11 BASE ADDRESS
(2)  007656  012737  004001  001124             MOV     #DVA!GO,$GDDAT  ;WHAT REGISTER SHOULD BE
(2)  007664  013737  001126  001156             MOV     $BDDAT,$TMP0    ;MOVE REGISTER CONTENTS TO '$TMP0'
(2)  007672  042737  173776  001156             BIC     #^C4001,$TMP0   ;SAVE SPECIFIED BITS
(2)  007700  023737  001124  001156             CMP     $GDDAT,$TMP0    ;COMPARE THE BITS
(2)  007706  001414                             BEQ     64$             ;BR IF OK
(2)  007710  013737  001126  001166             MOV     $BDDAT,$TMP4    ;COPY 'BAD DATA'
(2)  007716  042737  004001  001166             BIC     #4001,$TMP4     ;CLEAR THE MASKED BITS
(2)  007724  053737  001166  001124             BIS     $TMP4,$GDDAT    ;'OR' WITH GOOD DATA FOR TYPEOUT
(2)  007732  104011                             ERROR   11              ;TYPE MESSAGE 11
(2)  007734  005137  001236                     COM     CKERR           ;SET THE REGISTER COMPARE ERROR INDICATOR
(2)  007740  000240                     64$:    NOP
(2)  007742  005037  001236                     CLR     CKERR           ;CLEAR THE 'CHECK ERROR' INDICATOR
(2)  007746  016037  000012  001126             MOV     RHDS1(R0),$BDDAT  ;GET CONTENTS OF RHDS1
(2)  007754  012737  000012  001122             MOV     #RHDS1,$BDADR   ;FORM REGISTER ADDRESS OF ERROR MESSAGE
(2)  007762  060037  001122                     ADD     R0,$BDADR       ;ADD RH11 BASE ADDRESS
(2)  007766  012737  021500  001124             MOV     #PIP!PGM!DPR!VV,$GDDAT  ;WHAT REGISTER SHOULD BE
(2)  007774  013737  001126  001156             MOV     $BDDAT,$TMP0    ;MOVE REGISTER CONTENTS TO '$TMP0'
(2)  010002  042737  000077  001156             BIC     #^C177700,$TMP0  ;SAVE SPECIFIED BITS
```

```
(2)  010010  023737  001124  001156          CMP     $GDDAT,$TMP0    ;COMPARE THE BITS
(2)  010016  001414                           BEQ     65$             ;BR IF OK
(2)  010020  013737  001126  001166          MOV     $BDDAT,$TMP4    ;COPY 'BAD DATA'
(2)  010026  042737  177700  001166          BIC     #177700,$TMP4   ;CLEAR THE MASKED BITS
(2)  010034  053737  001166  001124          BIS     $TMP4,$GDDAT    ;'OR' WITH GOOD DATA FOR TYPEOUT
(2)  010042  104012                           ERROR   12              ;TYPE MESSAGE 12
(2)  010044  005137  001236                   COM     CKERR           ;SET THE REGISTER COMPARE ERROR INDICATOR
(2)  010050  000240                  65$:     NOP
(1)  010052  104400  022564                   TYPE    ,STANDBY        ;TYPE THE STANDBY MESSAGE
(2)  010056  013746  001216                   MOV     PORTA,-(SP)     ;;SAVE PORTA FOR TYPEOUT
(2)                                                                   ;;TYPE THE PORT NUMBER
(2)  010062  104410                           TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
(1)  010064  104400  001201                   TYPE    ,$CRLF          ;CR-LF
(1)
(2)                                 ;;*****************************************************************************
(1)                                 ;WAIT FOR 'MOL' TO SET
(1)
(1)  010070  032760  010000  000012  2$:      BIT     #MOL,RHDS1(R0)  ;WAIT FOR MOL TO SET
(1)  010076  001774                           BEQ     2$              ;LOOP UNTIL 'MOL' SETS
(1)  010100  012737  003720  001246           MOV     #2000.,WATCH    ;SETUP A 2 SECOND STALL
(2)  010106  113760  001220  000010           MOVB    PORTB,RHCS2(R0)  ;SELECT PORT B
(2)  010114  013737  001220  001226           MOV     PORTB,PTNBR  ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
(1)  010122  005760  000012          3$:      TST     RHDS1(R0)       ;DRIVE TIMEDOUT ?
(1)  010126  001004                           BNE     4$              ;BR IF IT HAS
(1)  010130  005737  001246                   TST     WATCH           ;2 SECONDS ELAPSED ?
(1)  010134  001372                           BNE     3$              ;BR IF NOT
(1)  010136  104006                           ERROR   6               ;NO TIMEOUT AFTER 2 SECONDS
(1)
(2)                                 ;;*****************************************************************************
(2)  010140                         4$:
(2)
(2)                                 ;VERIFY THAT THE DRIVE IS IN NEUTRAL
(2)
(2)  010140  005037  001242                   CLR     RELERR          ;CLEAR THE 'RELEASE ERROR ' INDICATOR
(2)  010144  012737  000012  001122           MOV     #RHDS1,$BDADR   ;FORM THE ADDRESS OF RHDS1 FOR TYPEOUT
(2)  010152  060037  001122                   ADD     R0,$BDADR       ;ADD THE I/O BASE ADDRESS
(2)  010156  012737  011600  001124           MOV     #MOL!PGM!DPR!DRY,$GDDAT  ;COMPARISON CONSTANT
(2)  010164  113760  001216  000010           MOVB    PORTA,RHCS2(R0) ;SELECT PORT A.
(2)  010172  016037  000012  001162           MOV     RHDS1(R0),$TMP2 ;GET THE DRIVE STATUS REGISTER FROM PORT A.
(2)  010200  013737  001162  001156           MOV     $TMP2,$TMP0     ;COPY IT INTO '$TMP0'
(2)  010206  042737  100100  001156           BIC     #ATA!VV,$TMP0   ;CLEAR PORT DEPENDENT BITS FROM THE COPY
(2)  010214  113760  001220  000010           MOVB    PORTB,RHCS2(R0) ;SELECT PORT B.
(2)  010222  016037  000012  001164           MOV     RHDS1(R0),$TMP3 ;GET THE DRIVE STATUS REGISTER FROM PORT B.
(2)  010230  013737  001164  001160           MOV     $TMP3,$TMP1     ;COPY IT INTO '$TMP1'
(2)  010236  042737  100100  001160           BIC     #ATA!VV,$TMP1   ;CLEAR PORT DEPENDENT BITS FROM THE COPY
(2)  010244  023737  001156  001160           CMP     $TMP0,$TMP1     ;IS THE STATUS REGISTER THE SAME FROM BOTH PORTS ?
(2)  010252  001006                           BNE     66$             ;BR IF NOT
(2)  010254  005737  001156                   TST     $TMP0           ;REGISTERS ARE THE SAME: ARE THEY ZERO ?
(2)  010260  001037                           BNE     68$             ;BR IF NOT
(2)  010262  104034                           ERROR   34              ;REPORT DRIVE NOT IN NEUTRAL OR NOT SEIZED
(2)  010264  000137  010450                   JMP     70$             ;BYPASS THE REST OF THE CHECKS
(2)  010270  013737  001162  001126  66$:     MOV     $TMP2,$BDDAT    ;SET UP POSSIBLE BAD DATA FOR ERROR MESSAGE
(2)  010276  013737  001220  001226           MOV     PORTB,PTNBR     ;SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
(2)  010304  113760  001220  000010           MOVB    PORTB,RHCS2(R0) ;SELECT PORT B.
```

```
(2)  010312  005737  001156                    TST     $TMP0           ;SEE IF STATUS EQ 0 FROM PORT A.
(2)  010316  001414                            BEQ     67$             ;BR IF ZERO
(2)  010320  013737  001216  001226            MOV     PORTA,PTNBR     ;SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
(2)  010326  013737  001164  001126            MOV     $TMP3,$BDDAT    ;'BAD DATA' FOR ERROR TYPE OUT
(2)  010334  113760  001216  000010            MOVB    PORTA,RHCS2(R0) ;SELECT PORT A.
(2)  010342  005737  001160                    TST     $TMP1           ;SEE IF STATUS EQ ZERO FROM PORT B.
(2)  010346  001004                            BNE     68$             ;BR IF NOT
(2)  010350  012737  177777  001242  67$:      MOV     #-1,RELERR      ;SET 'RELEASE ERROR' INDICATOR
(2)  010356  104013                            ERROR   13              ;TYPE ERROR MESSAGE 13
(2)  010360  013737  001162  001126  68$:      MOV     $TMP2,$BDDAT    ;LOOK FOR BIT FAILURES WHEN RHDS1 READ
(2)  010366  013737  001216  001226            MOV     PORTA,PTNBR     ;CHANGE PORT NUMBER
(2)  010374  042737  100100  001162            BIC     #ATA!VV,$TMP2   ;DON'T CHECK ATTN BIT OR VV BIT
(2)  010402  023737  001124  001162            CMP     $GDDAT,$TMP2    ;ALL BITS OK ?
(2)  010410  001401                            BEQ     69$             ;BR IF OK FROM PORT A.
(2)  010412  104037                            ERROR   37              ;REPORT ERROR
(2)  010414  013737  001164  001126  69$:      MOV     $TMP3,$BDDAT    ;CHECK RHDS1 FOR BIT FAILURES - FROM PORT B.
(2)  010422  013737  001220  001226            MOV     PORTB,PTNBR     ;CHANGE PORT NUMBER
(2)  010430  042737  100100  001164            BIC     #ATA!VV,$TMP3   ;DON'T CHECK ATTN BIT OR VV BIT
(2)  010436  023737  001124  001164            CMP     $GDDAT,$TMP3    ;SEE IF READ OK FROM PORT B.
(2)  010444  001401                            BEQ     70$             ;BR IF OK
(2)  010446  104037                            ERROR   37              ;REPORT THE ERROR
(2)  010450  000240                    70$:    NOP
(1)
(2)                                            ;;*****************************************************************************
(1)                                            ;CHECK THE ATTENTION BITS
(2)  010452  113760  001220  000010                    MOVB    PORTB,RHCS2(R0)  ;SELECT PORT B
(2)  010460  013737  001220  001226                    MOV     PORTB,PTNBR  ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
(2)  010466  005037  001236                            CLR     CKERR        ;CLEAR THE 'CHECK ERROR' INDICATOR
(2)  010472  016037  000012  001126                    MOV     RHDS1(R0),$BDDAT  ;GET CONTENTS OF RHDS1
(2)  010500  012737  000012  001122                    MOV     #RHDS1,$BDADR  ;FORM REGISTER ADDRESS OF ERROR MESSAGE
(2)  010506  060037  001122                            ADD     R0,$BDADR      ;ADD RH11 BASE ADDRESS
(2)  010512  005037  001124                            CLR     $GDDAT         ;WHAT REGISTER SHOULD BE
(2)  010516  013737  001126  001156                    MOV     $BDDAT,$TMP0   ;MOVE REGISTER CONTENTS TO '$TMP0'
(2)  010524  042737  077777  001156                    BIC     #^CATA,$TMP0  ;SAVE SPECIFIED BITS
(2)  010532  023737  001124  001156                    CMP     $GDDAT,$TMP0   ;COMPARE THE BITS
(2)  010540  001414                                    BEQ     71$            ;BR IF OK
(2)  010542  013737  001126  001166                    MOV     $BDDAT,$TMP4   ;COPY 'BAD DATA'
(2)  010550  042737  100000  001166                    BIC     #ATA,$TMP4     ;CLEAR THE MASKED BITS
(2)  010556  053737  001166  001124                    BIS     $TMP4,$GDDAT   ;'OR' WITH GOOD DATA FOR TYPEOUT
(2)  010564  104014                                    ERROR   14             ;TYPE MESSAGE 14
(2)  010566  005137  001236                            COM     CKERR          ;SET THE REGISTER COMPARE ERROR INDICATOR
(2)  010572  000240                            71$:    NOP
(2)  010574  113760  001216  000010                    MOVB    PORTA,RHCS2(R0)  ;SELECT PORT A
(2)  010602  013737  001216  001226                    MOV     PORTA,PTNBR  ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
(2)  010610  005037  001236                            CLR     CKERR        ;CLEAR THE 'CHECK ERROR' INDICATOR
(2)  010614  016037  000012  001126                    MOV     RHDS1(R0),$BDDAT  ;GET CONTENTS OF RHDS1
(2)  010622  012737  000012  001122                    MOV     #RHDS1,$BDADR  ;FORM REGISTER ADDRESS OF ERROR MESSAGE
(2)  010630  060037  001122                            ADD     R0,$BDADR      ;ADD RH11 BASE ADDRESS
(2)  010634  012737  100000  001124                    MOV     #ATA,$GDDAT  ;WHAT REGISTER SHOULD BE
(2)  010642  013737  001126  001156                    MOV     $BDDAT,$TMP0   ;MOVE REGISTER CONTENTS TO '$TMP0'
(2)  010650  042737  077777  001156                    BIC     #^CATA,$TMP0  ;SAVE SPECIFIED BITS
(2)  010656  023737  001124  001156                    CMP     $GDDAT,$TMP0   ;COMPARE THE BITS
(2)  010664  001414                                    BEQ     72$            ;BR IF OK
(2)  010666  013737  001126  001166                    MOV     $BDDAT,$TMP4   ;COPY 'BAD DATA'
```

```
(2)  010674  042737  100000  001166          BIC     #ATA,$TMP4      ;CLEAR THE MASKED BITS
(2)  010702  053737  001166  001124          BIS     $TMP4,$GDDAT    ;'OR' WITH GOOD DATA FOR TYPEOUT
(2)  010710  104015                          ERROR   15              ;TYPE MESSAGE 15
(2)  010712  005137  001236                  COM     CKERR           ;SET THE REGISTER COMPARE ERROR INDICATOR
(2)  010716  000240                  72$:    NOP
(1)
(2)                                         ;;*********************************************************************
(1)                                         ;SET 'VV' FOR EACH PORT
(1)
(1)  010720  012760  000011  000000          MOV     #11,RHCS1(R0)   ;CLEAR THE DRIVE
(1)  010726  012760  000021  000000          MOV     #21,RHCS1(R0)   ;DO A READIN PRESET THROUGH PORT A
(1)  010734  012760  000013  000000          MOV     #13,RHCS1(R0)   ;RELEASE THE DRIVE
(1)  010742  113760  001220  000010          MOVB    PORTB,RHCS2(R0) ;SELECT PORT B
(1)  010750  012760  000021  000000          MOV     #21,RHCS1(R0)   ;DO A READIN PRESET THROUGH PORT B
(1)  010756  012760  010000  000032          MOV     #FMT22,RHOF(R0) ;SET 'FMT22'
(1)  010764  012760  000013  000000          MOV     #13,RHCS1(R0)   ;RELEASE THE DRIVE
(2)
(2)                                         ;IF ERROR OCCURED, CHECK FOR LOOP ON TEST
(2)
(2)  010772  105737  001103                  TSTB    $ERFLG          ;DID AN ERROR OCCUR ?
(4)  010776  001412                          BEQ     TST7            ;;BR IF NOT
(2)  011000  032737  001000  177570          BIT     #SWD9,SWR       ;SEE IF LOOP ON ERROR SET (SWR9=1)
(4)  011006  001406                          BEQ     TST7            ;;BR IF NOT
(2)  011010  105037  001103                  CLRB    $ERFLG          ;CLEAR THE ERROR FLAG
(2)  011014  005037  001170                  CLR     $TIMES          ;CLEAR THE MAX ITERATION COUNT
(2)  011020  000177  170064                  JMP     @$LPERR         ;GO TO THE LOOP ADDRESS
5124
(5)                                         ;*********************************************************************
(4)                                         ;*TEST 7        TEST UNLOAD COMMAND THROUGH PORT B
(5)                                         ;*
(5)                                         ;*VERIFY THAT THE UNLOAD COMMAND FUNCTIONS PROPERLY AND THAT A PORT
(5)                                         ;*      TIMEOUT WILL NOT OCCUR WHILE THE 'GO' BIT IS SET.
(5)                                         ;*
(5)                                         ;*  A.  ISSUE AN UNLOAD COMMAND THROUGH PORT B; VERIFY THAT THE
(5)                                         ;*      DRIVE IS SEIZED.
(5)                                         ;*
(5)                                         ;*  B.  WAIT THE MEASURED TIMEOUT INTERVAL + 25%; VERIFY THAT THE DRIVE
(5)                                         ;*      DOES NOT TIME OUT.  VERIFY THAT THE 'GO' BIT IS STILL SET AND
(5)                                         ;*      THAT 'DRY' AND 'PIP' ARE NOT SET.
(5)                                         ;*
(5)                                         ;*  C.  REQUEST THAT THE OPERATOR PRESS THE 'STANDBY' BUTTON ON THE DRIVE.
(5)                                         ;*
(5)                                         ;*  D.  WHEN THE DRIVE CYCLES UP, VERIFY THAT THE DRIVE IS STILL SEIZED
(5)                                         ;*      BY PORT B, THAT THE 'VV' BIT FOR PORT B IS RESET, THAT THE
(5)                                         ;*      ATTENTION BIT FOR PORT B IS SET, AND THAT THE ATTENTION BIT
(5)                                         ;*      FOR PORT A IS NOT SET.
(5)                                         ;*
(5)                                         ;*  E.  WAIT FOR THE PORT TIMEOUT TO RELEASE THE DRIVE.  WHEN THE TIMEOUT
(5)                                         ;*      OCCURS, VERIFY THAT THE DRIVE RETURNED TO NEUTRAL, THAT THE
(5)                                         ;*      ATTENTION BIT FOR PORT B IS STILL SET, AND THAT THE ATTENTION
(5)                                         ;*      BIT FOR PORT A IS NOT SET.
(5)                                         ;*
(5)                                         ;*  F.  VERIFY THAT THE 'VV' BIT FOR PORT A IS NOT SET.
(5)                                         ;*
```

```
(5)                                     ;*  G.  ISSUE A PACK ACKNOWLEDGE INSTRUCTION THROUGH BOTH PORTS.
(5)                                     ;*
(4)                                     ;*****************************************************************
(4)  011024                             TST7:
(4)  011024  000004                             SCOPE                           ;INITIALIZE THE SCOPE HANDLER
(4)  011026  005737  001260                      TST     KYBCTL                  ;PERFORMING ONLY SINGLE TESTS ?
(4)  011032  001406                              BEQ     2$                      ;BR IF NOT
(4)  011034  100002                              BPL     1$                      ;BR IF JUST ENTERED TEST
(4)  011036  000137  002262                      JMP     EXEC                    ;RETURN & GET NEXT TEST NUMBER
(4)  011042  012737  177777  001260  1$:        MOV     #-1,KYBCTL              ;SET SINGLE TEST INDICATOR
(4)  011050  112737  000007  001102  2$:        MOVB    #7,$TSTNM               ;TEST NUMBER
(4)  011056  012737  011100  001106             MOV     #TEST7,$LPADR           ;LOAD LOOP ON TEST ADDRESS
(4)  011064  012737  011100  001110             MOV     #TEST7,$LPERR           ;LOAD LOOP ON ERROR ADDRESS
(2)  011072  012737  000001  001170             MOV     #1,$TIMES               ;;DO 1 ITERATION
(1)
(2)
(3)                                     ;;*************************************************************************
(2)                                     ;END OF 'SCOPE' SETUP - START OF MAIN TEST
(2)
(2)  011100                             TEST7:
(2)
(2)                                             ;CLEAR ATTENTION BITS FOR BOTH PORTS
(2)
(2)  011100  113760  001216  000010             MOVB    PORTA,RHCS2(R0) ;SELECT PORT #A
(2)  011106  005060  000012                     CLR     RHDS1(R0)       ;SEIZE THE DRIVE
(2)  011112  012760  000011  000000             MOV     #11,RHCS1(R0)   ;ISSUE DRIVE CLEAR
(2)  011120  012760  000013  000000             MOV     #13,RHCS1(R0)   ;RELEASE THE DRIVE
(2)  011126  113760  001220  000010             MOVB    PORTB,RHCS2(R0) ;SELECT PORT #B
(2)  011134  005060  000012                     CLR     RHDS1(R0)       ;SEIZE THE DRIVE THROUGH PORT 'B'
(2)  011140  012760  000011  000000             MOV     #11,RHCS1(R0)   ;ISSUE DRIVE CLEAR
(2)  011146  012760  000013  000000             MOV     #13,RHCS1(R0)   ;RELEASE THE DRIVE
(2)  011154  113760  001220  000010             MOVB    PORTB,RHCS2(R0)  ;SELECT PORT B
(2)  011162  013737  001220  001226             MOV     PORTB,PTNBR  ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
(1)  011170  013737  001220  001230             MOV     PORTB,SEIZPT    ;SEIZING PORT'S ADDRESS
(1)
(2)                                     ;;*************************************************************************
(1)                                     ;DO AN UNLOAD COMMAND THROUGH PORT B
(1)
(1)  011176  012760  000003  000000             MOV     #3,RHCS1(R0)    ;ISSUE AN UNLOAD COMMAND THROUGH PORT B
(1)  011204  013737  001256  001246             MOV     TIMEBP,WATCH    ;TIMEOUT ONESHOT VALUE + 25%
(1)  011212  005737  001246          1$:        TST     WATCH           ;FINISHED TIMEOUT ?
(1)  011216  001375                             BNE     1$              ;BR IF NOT
(1)
(2)                                     ;;*************************************************************************
(1)                                     ;IS THE STATUS OK ?
(1)
(2)  011220  005037  001236                     CLR     CKERR           ;CLEAR THE 'CHECK ERROR' INDICATOR
(2)  011224  016037  000000  001126             MOV     RHCS1(R0),$BDDAT  ;GET CONTENTS OF RHCS1
(2)  011232  012737  000000  001122             MOV     #RHCS1,$BDADR   ;FORM REGISTER ADDRESS OF ERROR MESSAGE
(2)  011240  060037  001122                     ADD     R0,$BDADR       ;ADD RH11 BASE ADDRESS
(2)  011244  012737  004001  001124             MOV     #DVA!GO,$GDDAT  ;WHAT REGISTER SHOULD BE
(2)  011252  013737  001126  001156             MOV     $BDDAT,$TMP0    ;MOVE REGISTER CONTENTS TO '$TMP0'
(2)  011260  042737  173776  001156             BIC     #^C400!,$TMP0   ;SAVE SPECIFIED BITS
(2)  011266  023737  001124  001156             CMP     $GDDAT,$TMP0    ;COMPARE THE BITS
```

```
(2)  011274  001414                        BEQ     64$             ;BR IF OK
(2)  011276  013737  001126  001166        MOV     $BDDAT,$TMP4    ;COPY 'BAD DATA'
(2)  011304  042737  004001  001166        BIC     #4001,$TMP4     ;CLEAR THE MASKED BITS
(2)  011312  053737  001166  001124        BIS     $TMP4,$GDDAT    ;'OR' WITH GOOD DATA FOR TYPEOUT
(2)  011320  104011                        ERROR   11              ;TYPE MESSAGE 11
(2)  011322  005137  001236                COM     CKERR           ;SET THE REGISTER COMPARE ERROR INDICATOR
(2)  011326  000240                 64$:    NOP
(2)  011330  005037  001236                CLR     CKERR           ;CLEAR THE 'CHECK ERROR' INDICATOR
(2)  011334  016037  000012  001126        MOV     RHDS1(R0),$BDDAT  ;GET CONTENTS OF RHDS1
(2)  011342  012737  000012  001122        MOV     #RHDS1,$BDADR   ;FORM REGISTER ADDRESS OF ERROR MESSAGE
(2)  011350  060037  001122                ADD     R0,$BDADR       ;ADD RH11 BASE ADDRESS
(2)  011354  012737  021500  001124        MOV     #PIP!PGM!DPR!VV,$GDDAT  ;WHAT REGISTER SHOULD BE
(2)  011362  013737  001126  001156        MOV     $BDDAT,$TMP0    ;MOVE REGISTER CONTENTS TO '$TMP0'
(2)  011370  042737  000077  001156        BIC     #^C177700,$TMP0  ;SAVE SPECIFIED BITS
(2)  011376  023737  001124  001156        CMP     $GDDAT,$TMP0    ;COMPARE THE BITS
(2)  011404  001414                        BEQ     65$             ;BR IF OK
(2)  011406  013737  001126  001166        MOV     $BDDAT,$TMP4    ;COPY 'BAD DATA'
(2)  011414  042737  177700  001166        BIC     #177700,$TMP4   ;CLEAR THE MASKED BITS
(2)  011422  053737  001166  001124        BIS     $TMP4,$GDDAT    ;'OR' WITH GOOD DATA FOR TYPEOUT
(2)  011430  104012                        ERROR   12              ;TYPE MESSAGE 12
(2)  011432  005137  001236                COM     CKERR           ;SET THE REGISTER COMPARE ERROR INDICATOR
(2)  011436  000240                 65$:    NOP
(1)  011440  104400  022564                TYPE    ,STANDBY        ;TYPE THE STANDBY MESSAGE
(2)  011444  013746  001220                MOV     PORTB,-(SP)     ;;SAVE PORTB FOR TYPEOUT
(2)                                        ;;TYPE THE PORT NUMBER
(2)  011450  104410                        TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
(1)  011452  104400  001201                TYPE    ,$CRLF          ;CR-LF
(1)
(2)                                 ;;*****************************************************************************
(1)                                 ;WAIT FOR 'MOL' TO SET
(1)
(1)  011456  032760  010000  000012  2$:     BIT     #MOL,RHDS1(R0)  ;WAIT FOR MOL TO SET
(1)  011464  001774                        BEQ     2$              ;LOOP UNTIL 'MOL' SETS
(1)  011466  012737  003720  001246        MOV     #2000.,WATCH    ;SETUP A 2 SECOND STALL
(2)  011474  113760  001216  000010        MOVB    PORTA,RHCS2(R0)  ;SELECT PORT A
(2)  011502  013737  001216  001226        MOV     PORTA,PTNBR  ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
(1)  011510  005760  000012         3$:    TST     RHDS1(R0)       ;DRIVE TIMEDOUT ?
(1)  011514  001004                        BNE     4$              ;BR IF IT HAS
(1)  011516  005737  001246                TST     WATCH           ;2 SECONDS ELAPSED ?
(1)  011522  001372                        BNE     3$              ;BR IF NOT
(1)  011524  104006                        ERROR   6               ;NO TIMEOUT AFTER 2 SECONDS
(1)
(2)                                 ;;*****************************************************************************
(2)  011526                         4$:
(2)
(2)                                         ;VERIFY THAT THE DRIVE IS IN NEUTRAL
(2)
(2)  011526  005037  001242                CLR     RELERR          ;CLEAR THE 'RELEASE ERROR ' INDICATOR
(2)  011532  012737  000012  001122        MOV     #RHDS1,$BDADR   ;FORM THE ADDRESS OF RHDS1 FOR TYPEOUT
(2)  011540  060037  001122                ADD     R0,$BDADR       ;ADD THE I/O BASE ADDRESS
(2)  011544  012737  011600  001124        MOV     #MOL!PGM!DPR!DRY,$GDDAT  ;COMPARISON CONSTANT
(2)  011552  113760  001216  000010        MOVB    PORTA,RHCS2(R0) ;SELECT PORT A.
(2)  011560  016037  000012  001162        MOV     RHDS1(R0),$TMP2 ;GET THE DRIVE STATUS REGISTER FROM PORT A.
(2)  011566  013737  001162  001156        MOV     $TMP2,$TMP0     ;COPY IT INTO '$TMP0'
```

```
(2)  011574  042737  100100  001156          BIC     #ATA!VV,$TMP0   ;CLEAR PORT DEPENDENT BITS FROM THE COPY
(2)  011602  113760  001220  000010          MOVB    PORTB,RHCS2(R0) ;SELECT PORT B.
(2)  011610  016037  000012  001164          MOV     RHDS1(R0),$TMP3 ;GET THE DRIVE STATUS REGISTER FROM PORT B.
(2)  011616  013737  001164  001160          MOV     $TMP3,$TMP1     ;COPY IT INTO '$TMP1'
(2)  011624  042737  100100  001160          BIC     #ATA!VV,$TMP1   ;CLEAR PORT DEPENDENT BITS FROM THE COPY
(2)  011632  023737  001156  001160          CMP     $TMP0,$TMP1     ;IS THE STATUS REGISTER THE SAME FROM BOTH PORTS ?
(2)  011640  001006                          BNE     66$             ;BR IF NOT
(2)  011642  005737  001156                  TST     $TMP0           ;REGISTERS ARE THE SAME: ARE THEY ZERO ?
(2)  011646  001037                          BNE     68$             ;BR IF NOT
(2)  011650  104034                          ERROR   34              ;REPORT DRIVE NOT IN NEUTRAL OR NOT SEIZED
(2)  011652  000137  012036                  JMP     70$             ;BYPASS THE REST OF THE CHECKS
(2)  011656  013737  001162  001126  66$:    MOV     $TMP2,$BDDAT    ;SET UP POSSIBLE BAD DATA FOR ERROR MESSAGE
(2)  011664  013737  001220  001226          MOV     PORTB,PTNBR     ;SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
(2)  011672  113760  001220  000010          MOVB    PORTB,RHCS2(R0) ;SELECT PORT B.
(2)  011700  005737  001156                  TST     $TMP0           ;SEE IF STATUS EQ 0 FROM PORT A.
(2)  011704  001414                          BEQ     67$             ;BR IF ZERO
(2)  011706  013737  001216  001226          MOV     PORTA,PTNBR     ;SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
(2)  011714  013737  001164  001126          MOV     $TMP3,$BDDAT    ;'BAD DATA' FOR ERROR TYPE OUT
(2)  011722  113760  001216  000010          MOVB    PORTA,RHCS2(R0) ;SELECT PORT A.
(2)  011730  005737  001160                  TST     $TMP1           ;SEE IF STATUS EQ ZERO FROM PORT B.
(2)  011734  001004                          BNE     68$             ;BR IF NOT
(2)  011736  012737  177777  001242  67$:    MOV     #-1,RELERR      ;SET 'RELEASE ERROR' INDICATOR
(2)  011744  104013                          ERROR   13              ;TYPE ERROR MESSAGE 13
(2)  011746  013737  001162  001126  68$:    MOV     $TMP2,$BDDAT    ;LOOK FOR BIT FAILURES WHEN RHDS1 READ
(2)  011754  013737  001216  001226          MOV     PORTA,PTNBR     ;CHANGE PORT NUMBER
(2)  011762  042737  100100  001162          BIC     #ATA!VV,$TMP2   ;DON'T CHECK ATTN BIT OR VV BIT
(2)  011770  023737  001124  001162          CMP     $GDDAT,$TMP2    ;ALL BITS OK ?
(2)  011776  001401                          BEQ     69$             ;BR IF OK FROM PORT A.
(2)  012000  104037                          ERROR   37              ;REPORT ERROR
(2)  012002  013737  001164  001126  69$:    MOV     $TMP3,$BDDAT    ;CHECK RHDS1 FOR BIT FAILURES - FROM PORT B.
(2)  012010  013737  001220  001226          MOV     PORTB,PTNBR     ;CHANGE PORT NUMBER
(2)  012016  042737  100100  001164          BIC     #ATA!VV,$TMP3   ;DON'T CHECK ATTN BIT OR VV BIT
(2)  012024  023737  001124  001164          CMP     $GDDAT,$TMP3    ;SEE IF READ OK FROM PORT B.
(2)  012032  001401                          BEQ     70$             ;BR IF OK
(2)  012034  104037                          ERROR   37              ;REPORT THE ERROR
(2)  012036  000240                  70$:    NOP
(1)
(2)                                         ;;*************************************************************************
(1)                                         ;CHECK THE ATTENTION BITS
(2)  012040  113760  001216  000010                 MOVB    PORTA,RHCS2(R0)  ;SELECT PORT A
(2)  012046  013737  001216  001226                 MOV     PORTA,PTNBR  ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
(2)  012054  005037  001236                         CLR     CKERR        ;CLEAR THE 'CHECK ERROR' INDICATOR
(2)  012060  016037  000012  001126                 MOV     RHDS1(R0),$BDDAT  ;GET CONTENTS OF RHDS1
(2)  012066  012737  000012  001122                 MOV     #RHDS1,$BDADR  ;FORM REGISTER ADDRESS OF ERROR MESSAGE
(2)  012074  060037  001122                         ADD     R0,$BDADR    ;ADD RH11 BASE ADDRESS
(2)  012100  005037  001124                         CLR     $GDDAT       ;WHAT REGISTER SHOULD BE
(2)  012104  013737  001126  001156                 MOV     $BDDAT,$TMP0 ;MOVE REGISTER CONTENTS TO '$TMP0'
(2)  012112  042737  077777  001156                 BIC     #^CATA,$TMP0  ;SAVE SPECIFIED BITS
(2)  012120  023737  001124  001156                 CMP     $GDDAT,$TMP0 ;COMPARE THE BITS
(2)  012126  001414                                 BEQ     71$          ;BR IF OK
(2)  012130  013737  001126  001166                 MOV     $BDDAT,$TMP4 ;COPY 'BAD DATA'
(2)  012136  042737  100000  001166                 BIC     #ATA,$TMP4   ;CLEAR THE MASKED BITS
(2)  012144  053737  001166  001124                 BIS     $TMP4,$GDDAT ;'OR' WITH GOOD DATA FOR TYPEOUT
(2)  012152  104014                                 ERROR   14           ;TYPE MESSAGE 14
```

```
(2)  012154  005137  001236          COM     CKERR           ;SET THE REGISTER COMPARE ERROR INDICATOR
(2)  012160  000240            71$: NOP
(2)  012162  113760  001220  000010 MOVB    PORTB,RHCS2(R0)  ;SELECT PORT B
(2)  012170  013737  001220  001226 MOV     PORTB,PTNBR  ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
(2)  012176  005037  001236          CLR     CKERR           ;CLEAR THE 'CHECK ERROR' INDICATOR
(2)  012202  016037  000012  001126 MOV     RHDS1(R0),$BDDAT  ;GET CONTENTS OF RHDS1
(2)  012210  012737  000012  001122 MOV     #RHDS1,$BDADR  ;FORM REGISTER ADDRESS OF ERROR MESSAGE
(2)  012216  060037  001122          ADD     R0,$BDADR       ;ADD RH11 BASE ADDRESS
(2)  012222  012737  100000  001124 MOV     #ATA,$GDDAT  ;WHAT REGISTER SHOULD BE
(2)  012230  013737  001126  001156 MOV     $BDDAT,$TMP0    ;MOVE REGISTER CONTENTS TO '$TMP0'
(2)  012236  042737  077777  001156 BIC     #^CATA,$TMP0  ;SAVE SPECIFIED BITS
(2)  012244  023737  001124  001156 CMP     $GDDAT,$TMP0    ;COMPARE THE BITS
(2)  012252  001414                BEQ     72$             ;BR IF OK
(2)  012254  013737  001126  001166 MOV     $BDDAT,$TMP4    ;COPY 'BAD DATA'
(2)  012262  042737  100000  001166 BIC     #ATA,$TMP4      ;CLEAR THE MASKED BITS
(2)  012270  053737  001166  001124 BIS     $TMP4,$GDDAT    ;'OR' WITH GOOD DATA FOR TYPEOUT
(2)  012276  104015                ERROR   15              ;TYPE MESSAGE 15
(2)  012300  005137  001236          COM     CKERR           ;SET THE REGISTER COMPARE ERROR INDICATOR
(2)  012304  000240            72$: NOP
(1)
(2)                                ;;*********************************************************************
(1)                                ;SET 'VV' FOR EACH PORT
(1)
(1)  012306  012760  000011  000000 MOV     #11,RHCS1(R0)   ;CLEAR THE DRIVE
(1)  012314  012760  000021  000000 MOV     #21,RHCS1(R0)   ;DO A READIN PRESET THROUGH PORT B
(1)  012322  012760  000013  000000 MOV     #13,RHCS1(R0)   ;RELEASE THE DRIVE
(1)  012330  113760  001216  000010 MOVB    PORTA,RHCS2(R0)  ;SELECT PORT A
(1)  012336  012760  000021  000000 MOV     #21,RHCS1(R0)   ;DO A READIN PRESET THROUGH PORT A
(1)  012344  012760  010000  000032 MOV     #FMT22,RHOF(R0) ;SET 'FMT22'
(1)  012352  012760  000013  000000 MOV     #13,RHCS1(R0)   ;RELEASE THE DRIVE
(2)
(2)                                ;IF ERROR OCCURED, CHECK FOR LOOP ON TEST
(2)
(2)  012360  105737  001103          TSTB    $ERFLG          ;DID AN ERROR OCCUR ?
(4)  012364  001412                BEQ     TST10           ;;BR IF NOT
(2)  012366  032737  001000  177570 BIT     #SW09,SWR       ;SEE IF LOOP ON ERROR SET (SWR9=1)
(4)  012374  001406                BEQ     TST10           ;;BR IF NOT
(2)  012376  105037  001103          CLRB    $ERFLG          ;CLEAR THE ERROR FLAG
(2)  012402  005037  001170          CLR     $TIMES          ;CLEAR THE MAX ITERATION COUNT
(2)  012406  000177  166476          JMP     @$LPERR         ;GO TO THE LOOP ADDRESS
5125
5198
5199
(5)                                ;*********************************************************************
(4)                                ;*TEST 10       TEST 'CONTROLLER SELECT' SWITCH THROUGH PORT A DURING UNLOAD
(5)                                ;*
(5)                                ;*VERIFY THAT THE 'CONTROLLER SELECT' SWITCH IS INHIBITED WHEN THE
(5)                                ;*      RPQ4 IS CYCLED DOWN BY AN UNLOAD COMMAND.
(5)                                ;*
(5)                                ;*  A.  ISSUE AN UNLOAD COMMAND THROUGH PORT A.
(5)                                ;*
(5)                                ;*  B.  REQUEST THAT THE OPERATOR SWITCH THE 'CONTROLLER SELECT' SWITCH
(5)                                ;*      TO A AND THEN PRESS THE 'STANDBY' BUTTON.
(5)                                ;*
```

```
(5)                                     ;*  C.  WAIT FOR 'MOL' TO SET BY MONITORING RHDS1 THROUGH PORT
(5)                                     ;*      A.  WHEN THE DRIVE HAS CYCLED UP ('MOL' HAS SET), ISSUE
(5)                                     ;*      RELEASE COMMAND THROUGH PORT A.
(5)                                     ;*
(5)                                     ;*  D.  VERIFY THAT THE DRIVE CAN BE ACCESSED BY BOTH PORTS AND THAT
(5)                                     ;*      THE DRIVE IS STILL IN NEUTRAL.
(5)                                     ;*
(5)                                     ;*  E.  REQUEST THAT THE OPERATOR RETURN THE 'CONTROLLER SELECT' SWITCH
(5)                                     ;*      TO A/B'.
(5)                                     ;*
(4)                                     ;*****************************************************************
(4)  012412                             TST10:
(4)  012412  000004                             SCOPE                   ;INITIALIZE THE SCOPE HANDLER
(4)  012414  005737  001260                     TST     KYBCTL          ;PERFORMING ONLY SINGLE TESTS ?
(4)  012420  001406                             BEQ     2$              ;BR IF NOT
(4)  012422  100002                             BPL     1$              ;BR IF JUST ENTERED TEST
(4)  012424  000137  002262                     JMP     EXEC            ;RETURN & GET NEXT TEST NUMBER
(4)  012430  012737  177777  001260  1$:       MOV     #-1,KYBCTL      ;SET SINGLE TEST INDICATOR
(4)  012436  112737  000010  001102  2$:       MOVB    #10,$TSTNM      ;TEST NUMBER
(4)  012444  012737  012466  001106            MOV     #TEST10,$LPADR  ;LOAD LOOP ON TEST ADDRESS
(4)  012452  012737  012466  001110            MOV     #TEST10,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
(2)  012460  012737  000001  001170            MOV     #1,$TIMES       ;;DO 1 ITERATION
(1)
(2)
(3)                                     ;;*****************************************************************************
(2)                                     ;END OF 'SCOPE' SETUP - START OF MAIN TEST
(2)
(2)  012466                             TEST10:
(2)
(2)                                             ;CLEAR ATTENTION BITS FOR BOTH PORTS
(2)
(2)  012466  113760  001216  000010            MOVB    PORTA,RHCS2(R0) ;SELECT PORT #A
(2)  012474  005060  000012                     CLR     RHDS1(R0)       ;SEIZE THE DRIVE
(2)  012500  012760  000011  000000            MOV     #11,RHCS1(R0)   ;ISSUE DRIVE CLEAR
(2)  012506  012760  000013  000000            MOV     #13,RHCS1(R0)   ;RELEASE THE DRIVE
(2)  012514  113760  001220  000010            MOVB    PORTB,RHCS2(R0) ;SELECT PORT #B
(2)  012522  005060  000012                     CLR     RHDS1(R0)       ;SEIZE THE DRIVE THROUGH PORT 'B'
(2)  012526  012760  000011  000000            MOV     #11,RHCS1(R0)   ;ISSUE DRIVE CLEAR
(2)  012534  012760  000013  000000            MOV     #13,RHCS1(R0)   ;RELEASE THE DRIVE
(2)  012542  113760  001216  000010            MOVB    PORTA,RHCS2(R0)  ;SELECT PORT A
(2)  012550  013737  001216  001226            MOV     PORTA,PTNBR  ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
(1)  012556  013737  001216  001230            MOV     PORTA,SEIZPT    ;SEIZING PORT'S ADDRESS
(1)  012564  012760  000003  000000            MOV     #3,RHCS1(R0)    ;ISSUE AN UNLOAD COMMAND THROUGH PORT A
(1)  012572  104400  022776                     TYPE    ,SWTCHA ;SWITCH TO PORT A MESSAGE
(1)  012576  104400  022564                     TYPE    ,STANDBY        ;TYPE PRESS STANDBY
(2)  012602  013746  001216                     MOV     PORTA,-(SP)     ;;SAVE PORTA FOR TYPEOUT
(2)                                             ;;TYPE THE PORT NUMBER
(2)  012606  104410                             TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
(1)  012610  104400  001201                     TYPE    ,$CRLF          ;CR-LF
(1)  012614  032760  010000  000012  1$:       BIT     #MOL,RHDS1(R0)  ;TEST 'MOL'
(1)  012622  001774                             BEQ     1$              ;BR IF NOT SET
(1)  012624  012760  000011  000000            MOV     #11,RHCS1(R0)   ;ISSUE A DRIVE CLEAR
(1)  012632  012760  000021  000000            MOV     #21,RHCS1(R0)   ;ISSUE A READIN PRESET
(1)  012640  012760  000013  000000            MOV     #13,RHCS1(R0)   ;RELEASE THE DRIVE
```

```
(2)
(2)                                                 ;VERIFY THAT THE DRIVE IS IN NEUTRAL
(2)
(2)  012646  005037  001242                         CLR     RELERR          ;CLEAR THE 'RELEASE ERROR ' INDICATOR
(2)  012652  012737  000012  001122                 MOV     #RHDS1,$BDADR   ;FORM THE ADDRESS OF RHDS1 FOR TYPEOUT
(2)  012660  060037  001122                         ADD     R0,$BDADR       ;ADD THE I/O BASE ADDRESS
(2)  012664  012737  011600  001124                 MOV     #MOL!PGM!DPR!DRY,$GDDAT  ;COMPARISON CONSTANT
(2)  012672  113760  001216  000010                 MOVB    PORTA,RHCS2(R0) ;SELECT PORT A.
(2)  012700  016037  000012  001162                 MOV     RHDS1(R0),$TMP2 ;GET THE DRIVE STATUS REGISTER FROM PORT A.
(2)  012706  013737  001162  001156                 MOV     $TMP2,$TMP0     ;COPY IT INTO '$TMP0'
(2)  012714  042737  100100  001156                 BIC     #ATA!VV,$TMP0   ;CLEAR PORT DEPENDENT BITS FROM THE COPY
(2)  012722  113760  001220  000010                 MOVB    PORTB,RHCS2(R0) ;SELECT PORT B.
(2)  012730  016037  000012  001164                 MOV     RHDS1(R0),$TMP3 ;GET THE DRIVE STATUS REGISTER FROM PORT B.
(2)  012736  013737  001164  001160                 MOV     $TMP3,$TMP1     ;COPY IT INTO '$TMP1'
(2)  012744  042737  100100  001160                 BIC     #ATA!VV,$TMP1   ;CLEAR PORT DEPENDENT BITS FROM THE COPY
(2)  012752  023737  001156  001160                 CMP     $TMP0,$TMP1     ;IS THE STATUS REGISTER THE SAME FROM BOTH PORTS ?
(2)  012760  001006                                 BNE     64$             ;BR IF NOT
(2)  012762  005737  001156                         TST     $TMP0           ;REGISTERS ARE THE SAME: ARE THEY ZERO ?
(2)  012766  001037                                 BNE     66$             ;BR IF NOT
(2)  012770  104034                                 ERROR   34              ;REPORT DRIVE NOT IN NEUTRAL OR NOT SEIZED
(2)  012772  000137  013156                         JMP     68$             ;BYPASS THE REST OF THE CHECKS
(2)  012776  013737  001162  001126  64$:           MOV     $TMP2,$BDDAT    ;SET UP POSSIBLE BAD DATA FOR ERROR MESSAGE
(2)  013004  013737  001220  001226                 MOV     PORTB,PTNBR     ;SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
(2)  013012  113760  001220  000010                 MOVB    PORTB,RHCS2(R0) ;SELECT PORT B.
(2)  013020  005737  001156                         TST     $TMP0           ;SEE IF STATUS EQ 0 FROM PORT A.
(2)  013024  001414                                 BEQ     65$             ;BR IF ZERO
(2)  013026  013737  001216  001226                 MOV     PORTA,PTNBR     ;SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
(2)  013034  013737  001164  001126                 MOV     $TMP3,$BDDAT    ;'BAD DATA' FOR ERROR TYPE OUT
(2)  013042  113760  001216  000010                 MOVB    PORTA,RHCS2(R0) ;SELECT PORT A.
(2)  013050  005737  001160                         TST     $TMP1           ;SEE IF STATUS EQ ZERO FROM PORT B.
(2)  013054  001004                                 BNE     66$             ;BR IF NOT
(2)  013056  012737  177777  001242  65$:           MOV     #-1,RELERR      ;SET 'RELEASE ERROR' INDICATOR
(2)  013064  104016                                 ERROR   16              ;TYPE ERROR MESSAGE 16
(2)  013066  013737  001162  001126  66$:           MOV     $TMP2,$BDDAT    ;LOOK FOR BIT FAILURES WHEN RHDS1 READ
(2)  013074  013737  001216  001226                 MOV     PORTA,PTNBR     ;CHANGE PORT NUMBER
(2)  013102  042737  000100  001162                 BIC     #VV,$TMP2       ;DON'T CHECK THE VV BIT
(2)  013110  023737  001124  001162                 CMP     $GDDAT,$TMP2    ;ALL BITS OK ?
(2)  013116  001401                                 BEQ     67$             ;BR IF OK FROM PORT A.
(2)  013120  104037                                 ERROR   37              ;REPORT ERROR
(2)  013122  013737  001164  001126  67$:           MOV     $TMP3,$BDDAT    ;CHECK RHDS1 FOR BIT FAILURES - FROM PORT B.
(2)  013130  013737  001220  001226                 MOV     PORTB,PTNBR     ;CHANGE PORT NUMBER
(2)  013136  042737  000100  001164                 BIC     #VV,$TMP3       ;DON'T CHECK THE VV BIT
(2)  013144  023737  001124  001164                 CMP     $GDDAT,$TMP3    ;SEE IF READ OK FROM PORT B.
(2)  013152  001401                                 BEQ     68$             ;BR IF OK
(2)  013154  104037                                 ERROR   37              ;REPORT THE ERROR
(2)  013156  000240                         68$:    NOP
(1)
(1)                                                 ;IF ERROR OCCURED, CHECK FOR LOOP ON TEST
(1)
(1)  013160  105737  001103                         TSTB    $ERFLG          ;DID AN ERROR OCCUR
(1)  013164  001412                                 BEQ     2$              ;BR IF NOT
(1)  013166  032737  001000  177570                 BIT     #SW09,SWR       ;SEE IF LOOP ON ERROR (SWR9=1)
(1)  013174  001406                                 BEQ     2$              ;BR IF NOT
(1)  013176  105037  001103                         CLRB    $ERFLG          ;CLEAR THE ERROR FLAG
```

```
(1)  013202  005037  001170                    CLR     $TIMES          ;CLEAR THE MAX ITERATION COUNT
(1)  013206  000177  165676                    JMP     @$LPERR         ;GO TO THE LOOP ADDRESS
(1)  013212  005737  001260          2$:       TST     KYBCTL          ;SINGLE TEST MODE ?
(1)  013216  001406                            BEQ     3$              ;BR IF NOT
(1)  013220  032737  040000  177570            BIT     #SW14,SWR       ;LOOP ON TEST ?
(1)  013226  001002                            BNE     3$              ;BR IF LOOPING
(1)  013230  104400  022707                    TYPE    ,SWTCHN         ;RETURN CONTROLLER SELECT SWITCH TO 'A/B'
(2)  013234                          3$:
(2)  013234  113760  001220  000010            MOVB    PORTB,RHCS2(RO)  ;SELECT PORT B
(2)  013242  013737  001220  001226            MOV     PORTB,PTNBR  ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
(1)  013250  012760  000011  000000            MOV     #11,RHCS1(RO)   ;ISSUE A DRIVE CLEAR
(1)  013256  012760  000021  000000            MOV     #21,RHCS1(RO)   ;ISSUE A READIN PRESET
(1)  013264  012760  010000  000032            MOV     #FMT22,RHOF(RO) ;SET 'FMT22'
(1)  013272  012760  000013  000000            MOV     #13,RHCS1(RO)   ;RELEASE PORT B
(3)  013300  000400                            BR      TST11           ;;GO TO NEXT TEST
5200
 (5)                                ;*****************************************************************
 (4)                                ;*TEST 11       TEST 'CONTROLLER SELECT' SWITCH THROUGH PORT B DURING UNLOAD
 (5)                                ;*
 (5)                                ;*VERIFY THAT THE 'CONTROLLER SELECT' SWITCH IS INHIBITED WHEN THE
 (5)                                ;*      RPO4 IS CYCLED DOWN BY AN UNLOAD COMMAND.
 (5)                                ;*
 (5)                                ;*  A.  ISSUE AN UNLOAD COMMAND THROUGH PORT B.
 (5)                                ;*
 (5)                                ;*  B.  REQUEST THAT THE OPERATOR SWITCH THE 'CONTROLLER SELECT' SWITCH
 (5)                                ;*      TO B AND THEN PRESS THE 'STANDBY' BUTTON.
 (5)                                ;*
 (5)                                ;*  C.  WAIT FOR 'MOL' TO SET BY MONITORING RHDS1 THROUGH PORT
 (5)                                ;*      B.  WHEN THE DRIVE HAS CYCLED UP ('MOL' HAS SET), ISSUE
 (5)                                ;*      RELEASE COMMAND THROUGH PORT B.
 (5)                                ;*
 (5)                                ;*  D.  VERIFY THAT THE DRIVE CAN BE ACCESSED BY BOTH PORTS AND THAT
 (5)                                ;*      THE DRIVE IS STILL IN NEUTRAL.
 (5)                                ;*
 (5)                                ;*  E.  REQUEST THAT THE OPERATOR RETURN THE 'CONTROLLER SELECT' SWITCH
 (5)                                ;*      TO 'A/B.
 (5)                                ;*
 (4)                                ;*****************************************************************
 (4)  013302                        TST11:
 (4)  013302  000004                            SCOPE                   ;INITIALIZE THE SCOPE HANDLER
 (4)  013304  005737  001260                    TST     KYBCTL          ;PERFORMING ONLY SINGLE TESTS ?
 (4)  013310  001406                            BEQ     2$              ;BR IF NOT
 (4)  013312  100002                            BPL     1$              ;BR IF JUST ENTERED TEST
 (4)  013314  000137  002262                    JMP     EXEC            ;RETURN & GET NEXT TEST NUMBER
 (4)  013320  012737  177777  001260  1$:       MOV     #-1,KYBCTL      ;SET SINGLE TEST INDICATOR
 (4)  013326  112737  000011  001102  2$:       MOVB    #11,$TSTNM      ;TEST NUMBER
 (4)  013334  012737  013356  001106            MOV     #TEST11,$LPADR  ;LOAD LOOP ON TEST ADDRESS
 (4)  013342  012737  013356  001110            MOV     #TEST11,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
 (2)  013350  012737  000001  001170            MOV     #1,$TIMES       ;;DO 1 ITERATION
 (1)
 (2)
 (3)                                ;;*************************************************************************
 (2)                                ;END OF 'SCOPE' SETUP - START OF MAIN TEST
 (2)
```

```
(2)  013356                          TEST11:
(2)
(2)                                          ;CLEAR ATTENTION BITS FOR BOTH PORTS
(2)
(2)  013356  113760  001216  000010         MOVB    PORTA,RHCS2(R0) ;SELECT PORT #A
(2)  013364  005060  000012                 CLR     RHDS1(R0)       ;SEIZE THE DRIVE
(2)  013370  012760  000011  000000         MOV     #11,RHCS1(R0)   ;ISSUE DRIVE CLEAR
(2)  013376  012760  000013  000000         MOV     #13,RHCS1(R0)   ;RELEASE THE DRIVE
(2)  013404  113760  001220  000010         MOVB    PORTB,RHCS2(R0) ;SELECT PORT #B
(2)  013412  005060  000012                 CLR     RHDS1(R0)       ;SEIZE THE DRIVE THROUGH PORT 'B'
(2)  013416  012760  000011  000000         MOV     #11,RHCS1(R0)   ;ISSUE DRIVE CLEAR
(2)  013424  012760  000013  000000         MOV     #13,RHCS1(R0)   ;RELEASE THE DRIVE
(2)  013432  113760  001220  000010         MOVB    PORTB,RHCS2(R0)  ;SELECT PORT B
(2)  013440  013737  001220  001226         MOV     PORTB,PTNBR  ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
(1)  013446  013737  001220  001230         MOV     PORTB,SEIZPT    ;SEIZING PORT'S ADDRESS
(1)  013454  012760  000003  000000         MOV     #3,RHCS1(R0)    ;ISSUE AN UNLOAD COMMAND THROUGH PORT B
(1)  013462  104400  023061                 TYPE    ,SWTCHB ;SWITCH TO PORT B MESSAGE
(1)  013466  104400  022564                 TYPE    ,STANDBY        ;TYPE PRESS STANDBY
(2)  013472  013746  001220                 MOV     PORTB,-(SP)     ;;SAVE PORTB FOR TYPEOUT
(2)                                          ;;TYPE THE PORT NUMBER
(2)  013476  104410                         TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
(1)  013500  104400  001201                 TYPE    ,$CRLF          ;CR-LF
(1)  013504  032760  010000  000012  1$:    BIT     #MOL,RHDS1(R0)  ;TEST 'MOL'
(1)  013512  001774                         BEQ     1$              ;BR IF NOT SET
(1)  013514  012760  000011  000000         MOV     #11,RHCS1(R0)   ;ISSUE A DRIVE CLEAR
(1)  013522  012760  000021  000000         MOV     #21,RHCS1(R0)   ;ISSUE A READIN PRESET
(1)  013530  012760  000013  000000         MOV     #13,RHCS1(R0)   ;RELEASE THE DRIVE
(2)
(2)                                          ;VERIFY THAT THE DRIVE IS IN NEUTRAL
(2)
(2)  013536  005037  001242                 CLR     RELERR          ;CLEAR THE 'RELEASE ERROR ' INDICATOR
(2)  013542  012737  000012  001122         MOV     #RHDS1,$BDADR   ;FORM THE ADDRESS OF RHDS1 FOR TYPEOUT
(2)  013550  060037  001122                 ADD     R0,$BDADR       ;ADD THE I/O BASE ADDRESS
(2)  013554  012737  011600  001124         MOV     #MOL!PGM!DPR!DRY,$GDDAT  ;COMPARISON CONSTANT
(2)  013562  113760  001216  000010         MOVB    PORTA,RHCS2(R0) ;SELECT PORT A.
(2)  013570  016037  000012  001162         MOV     RHDS1(R0),$TMP2 ;GET THE DRIVE STATUS REGISTER FROM PORT A.
(2)  013576  013737  001162  001156         MOV     $TMP2,$TMP0     ;COPY IT INTO '$TMP0'
(2)  013604  042737  100100  001156         BIC     #ATA!VV,$TMP0   ;CLEAR PORT DEPENDENT BITS FROM THE COPY
(2)  013612  113760  001220  000010         MOVB    PORTB,RHCS2(R0) ;SELECT PORT B.
(2)  013620  016037  000012  001164         MOV     RHDS1(R0),$TMP3 ;GET THE DRIVE STATUS REGISTER FROM PORT B.
(2)  013626  013737  001164  001160         MOV     $TMP3,$TMP1     ;COPY IT INTO '$TMP1'
(2)  013634  042737  100100  001160         BIC     #ATA!VV,$TMP1   ;CLEAR PORT DEPENDENT BITS FROM THE COPY
(2)  013642  023737  001156  001160         CMP     $TMP0,$TMP1     ;IS THE STATUS REGISTER THE SAME FROM BOTH PORTS ?
(2)  013650  001006                         BNE     64$             ;BR IF NOT
(2)  013652  005737  001156                 TST     $TMP0           ;REGISTERS ARE THE SAME: ARE THEY ZERO ?
(2)  013656  001037                         BNE     66$             ;BR IF NOT
(2)  013660  104034                         ERROR   34              ;REPORT DRIVE NOT IN NEUTRAL OR NOT SEIZED
(2)  013662  000137  014046                 JMP     68$             ;BYPASS THE REST OF THE CHECKS
(2)  013666  013737  001162  001126  64$:    MOV     $TMP2,$BDDAT    ;SET UP POSSIBLE BAD DATA FOR ERROR MESSAGE
(2)  013674  013737  001220  001226         MOV     PORTB,PTNBR     ;SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
(2)  013702  113760  001220  000010         MOVB    PORTB,RHCS2(R0) ;SELECT PORT B.
(2)  013710  005737  001156                 TST     $TMP0           ;SEE IF STATUS EQ 0 FROM PORT A.
(2)  013714  001414                         BEQ     65$             ;BR IF ZERO
(2)  013716  013737  001216  001226         MOV     PORTA,PTNBR     ;SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
```

```
(2)  013724  013737  001164  001126          MOV     $TMP3,$BDDAT    ;'BAD DATA' FOR ERROR TYPE OUT
(2)  013732  113760  001216  000010          MOVB    PORTA,RHCS2(R0) ;SELECT PORT A.
(2)  013740  005737  001160                  TST     $TMP1           ;SEE IF STATUS EQ ZERO FROM PORT B.
(2)  013744  001004                          BNE     66$             ;BR IF NOT
(2)  013746  012737  177777  001242  65$:    MOV     #-1,RELERR      ;SET 'RELEASE ERROR' INDICATOR
(2)  013754  104016                          ERROR   16              ;TYPE ERROR MESSAGE 16
(2)  013756  013737  001162  001126  66$:    MOV     $TMP2,$BDDAT    ;LOOK FOR BIT FAILURES WHEN RHDS1 READ
(2)  013764  013737  001216  001226          MOV     PORTA,PTNBR     ;CHANGE PORT NUMBER
(2)  013772  042737  000100  001162          BIC     #VV,$TMP2       ;DON'T CHECK THE VV BIT
(2)  014000  023737  001124  001162          CMP     $GDDAT,$TMP2    ;ALL BITS OK ?
(2)  014006  001401                          BEQ     67$             ;BR IF OK FROM PORT A.
(2)  014010  104037                          ERROR   37              ;REPORT ERROR
(2)  014012  013737  001164  001126  67$:    MOV     $TMP3,$BDDAT    ;CHECK RHDS1 FOR BIT FAILURES - FROM PORT B.
(2)  014020  013737  001220  001226          MOV     PORTB,PTNBR     ;CHANGE PORT NUMBER
(2)  014026  042737  000100  001164          BIC     #VV,$TMP3       ;DON'T CHECK THE VV BIT
(2)  014034  023737  001124  001164          CMP     $GDDAT,$TMP3    ;SEE IF READ OK FROM PORT B.
(2)  014042  001401                          BEQ     68$             ;BR IF OK
(2)  014044  104037                          ERROR   37              ;REPORT THE ERROR
(2)  014046  000240                  68$:    NOP
(1)
(1)                                          ;IF ERROR OCCURED, CHECK FOR LOOP ON TEST
(1)
(1)  014050  105737  001103                  TSTB    $ERFLG          ;DID AN ERROR OCCUR
(1)  014054  001412                          BEQ     2$              ;BR IF NOT
(1)  014056  032737  001000  177570          BIT     #SW09,SWR       ;SEE IF LOOP ON ERROR (SWR9=1)
(1)  014064  001406                          BEQ     2$              ;BR IF NOT
(1)  014066  105037  001103                  CLRB    $ERFLG          ;CLEAR THE ERROR FLAG
(1)  014072  005037  001170                  CLR     $TIMES          ;CLEAR THE MAX ITERATION COUNT
(1)  014076  000177  165006                  JMP     @$LPERR         ;GO TO THE LOOP ADDRESS
(1)  014102  032737  040000  177570  2$:     BIT     #SW14,SWR       ;LOOP ON TEST ?
(1)  014110  001002                          BNE     3$              ;BR IF LOOPING
(1)  014112  104400  022707                  TYPE    ,SWTCHN         ;RETURN CONTROLLER SELECT SWITCH TO 'A/B'
(2)  014116                          3$:
(2)  014116  113760  001216  000010          MOVB    PORTA,RHCS2(R0)  ;SELECT PORT A
(2)  014124  013737  001216  001226          MOV     PORTA,PTNBR  ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
(1)  014132  012760  000011  000000          MOV     #11,RHCS1(R0)    ;ISSUE A DRIVE CLEAR
(1)  014140  012760  000021  000000          MOV     #21,RHCS1(R0)    ;ISSUE A READIN PRESET
(1)  014146  012760  010000  000032          MOV     #FMT22,RHOF(R0)  ;SET 'FMT22'
(1)  014154  012760  000013  000000          MOV     #13,RHCS1(R0)    ;RELEASE PORT A
(3)  014162  000400                          BR      TST12            ;;GO TO NEXT TEST
5201
5218
5219                                         ;*****************************************************************
(3)                                          ;*TEST 12       TEST 'CONTROLLER SELECT' SWITCH, DRIVE CYCLED UP
(4)                                          ;*
(4)                                          ;*TEST THE OPERATION OF THE 'CONTROLLER SELECT' SWITCH (DRIVE CYCLED UP).
(4)                                          ;*
(4)                                          ;*  A.  SWITCH TO CONTROLLER 'A' POSITION.  VERIFY THAT THE DRIVE IS IN
(4)                                          ;*      NEUTRAL AND THAT THE STATUS BITS IN RHDS1, AS READ THROUGH BOTH
(4)                                          ;*      PORTS, ARE CORRECT.
(4)                                          ;*
(4)                                          ;*  B.  SWITCH TO CONTROLLER 'B' POSITION. VERIFY THAT THE DRIVE IS IN
(4)                                          ;*      NEUTRAL AND THAT THE STATUS BITS IN RHDS1, AS READ THROUGH BOTH
(4)                                          ;*      PORTS, ARE CORRECT.
```

```
 (4)                                    ;*
 (4)                                    ;*  C.  RETURN THE 'CONTROLLER SELECT' SWITCH TO THE 'A/B' POSITION.  VERIFY
 (4)                                    ;*      THE DRIVE STATE.
 (4)                                    ;*
 (3)                                    ;*********************************************************************
 (3)  014164                            TST12:
 (3)  014164  000004                            SCOPE                           ;INITIALIZE THE SCOPE HANDLER
 (3)  014166  005737  001260                    TST     KYBCTL                  ;PERFORMING ONLY SINGLE TESTS ?
 (3)  014172  001406                            BEQ     2$                      ;BR IF NOT
 (3)  014174  100002                            BPL     1$                      ;BR IF JUST ENTERED TEST
 (3)  014176  000137  002262                    JMP     EXEC                    ;RETURN & GET NEXT TEST NUMBER
 (3)  014202  012737  177777  001260  1$:      MOV     #-1,KYBCTL              ;SET SINGLE TEST INDICATOR
 (3)  014210  112737  000012  001102  2$:      MOVB    #12,$TSTNM              ;TEST NUMBER
 (3)  014216  012737  014240  001106          MOV     #TEST12,$LPADR          ;LOAD LOOP ON TEST ADDRESS
 (3)  014224  012737  014240  001110          MOV     #TEST12,$LPERR          ;LOAD LOOP ON ERROR ADDRESS
 (1)  014232  012737  000001  001170          MOV     #1,$TIMES               ;;DO 1 ITERATION
5220
5221
 (2)                                    ;;*********************************************************************
 (1)                                    ;END OF 'SCOPE' SETUP - START OF MAIN TEST
 (1)
 (1)  014240                            TEST12:
5222
 (1)                                            ;CLEAR ATTENTION BITS FOR BOTH PORTS
 (1)
 (1)  014240  113760  001216  000010            MOVB    PORTA,RHCS2(R0) ;SELECT PORT #A
 (1)  014246  005060  000012                    CLR     RHDS1(R0)       ;SEIZE THE DRIVE
 (1)  014252  012760  000011  000000            MOV     #11,RHCS1(R0)   ;ISSUE DRIVE CLEAR
 (1)  014260  012760  000013  000000            MOV     #13,RHCS1(R0)   ;RELEASE THE DRIVE
 (1)  014266  113760  001220  000010            MOVB    PORTB,RHCS2(R0) ;SELECT PORT #B
 (1)  014274  005060  000012                    CLR     RHDS1(R0)       ;SEIZE THE DRIVE THROUGH PORT 'B'
 (1)  014300  012760  000011  000000            MOV     #11,RHCS1(R0)   ;ISSUE DRIVE CLEAR
 (1)  014306  012760  000013  000000            MOV     #13,RHCS1(R0)   ;RELEASE THE DRIVE
5223  014314  104400  022776                    TYPE    ,SWTCHA         ;SWITCH TO 'A'
5224  014320  104400  023144                    TYPE    ,CONTUE         ;PRESS 'CONTINUE'
5225  014324  000000                            HALT
5226
 (1)                                            ;VERIFY THAT THE DRIVE IS IN NEUTRAL
 (1)
 (1)  014326  005037  001242                    CLR     RELERR          ;CLEAR THE 'RELEASE ERROR ' INDICATOR
 (1)  014332  012737  000012  001122            MOV     #RHDS1,$BDADR   ;FORM THE ADDRESS OF RHDS1 FOR TYPEOUT
 (1)  014340  060037  001122                    ADD     R0,$BDADR       ;ADD THE I/O BASE ADDRESS
 (1)  014344  012737  011700  001124            MOV     #MOL!PGM!DPR!DRY!VV,$GDDAT  ;COMPARISON CONSTANT
 (1)  014352  113760  001216  000010            MOVB    PORTA,RHCS2(R0) ;SELECT PORT A.
 (1)  014360  016037  000012  001162            MOV     RHDS1(R0),$TMP2 ;GET THE DRIVE STATUS REGISTER FROM PORT A.
 (1)  014366  013737  001162  001156            MOV     $TMP2,$TMP0     ;COPY IT INTO '$TMP0'
 (1)  014374  042737  100100  001156            BIC     #ATA!VV,$TMP0   ;CLEAR PORT DEPENDENT BITS FROM THE COPY
 (1)  014402  113760  001220  000010            MOVB    PORTB,RHCS2(R0) ;SELECT PORT B.
 (1)  014410  016037  000012  001164            MOV     RHDS1(R0),$TMP3 ;GET THE DRIVE STATUS REGISTER FROM PORT B.
 (1)  014416  013737  001164  001160            MOV     $TMP3,$TMP1     ;COPY IT INTO '$TMP1'
 (1)  014424  042737  100100  001160            BIC     #ATA!VV,$TMP1   ;CLEAR PORT DEPENDENT BITS FROM THE COPY
 (1)  014432  023737  001156  001160            CMP     $TMP0,$TMP1     ;IS THE STATUS REGISTER THE SAME FROM BOTH PORTS ?
 (1)  014440  001006                            BNE     64$             ;BR IF NOT
 (1)  014442  005737  001156                    TST     $TMP0           ;REGISTERS ARE THE SAME: ARE THEY ZERO ?
```

```
(1)  014446  001045                          BNE     66$             ;BR IF NOT
(1)  014450  104034                          ERROR   34              ;REPORT DRIVE NOT IN NEUTRAL OR NOT SEIZED
(1)  014452  000137  014636                  JMP     68$             ;BYPASS THE REST OF THE CHECKS
(1)  014456  013737  001162  001126  64$:    MOV     $TMP2,$BDDAT    ;SET UP POSSIBLE BAD DATA FOR ERROR MESSAGE
(1)  014464  013737  001220  001226          MOV     PORTB,PTNBR     ;SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
(1)  014472  113760  001220  000010          MOVB    PORTB,RHCS2(R0) ;SELECT PORT B.
(1)  014500  005737  001156                  TST     $TMP0           ;SEE IF STATUS EQ 0 FROM PORT A.
(1)  014504  001414                          BEQ     65$             ;BR IF ZERO
(1)  014506  013737  001216  001226          MOV     PORTA,PTNBR     ;SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
(1)  014514  013737  001164  001126          MOV     $TMP3,$BDDAT    ;'BAD DATA' FOR ERROR TYPE OUT
(1)  014522  113760  001216  000010          MOVB    PORTA,RHCS2(R0) ;SELECT PORT A.
(1)  014530  005737  001160                  TST     $TMP1           ;SEE IF STATUS EQ ZERO FROM PORT B.
(1)  014534  001012                          BNE     66$             ;BR IF NOT
(1)  014536  012737  177777  001242  65$:    MOV     #-1,RELERR      ;SET 'RELEASE ERROR' INDICATOR
(1)  014544  012760  000011  000000          MOV     #11,RHCS1(R0)   ;CLEAR THE DRIVE
(1)  014552  012760  000013  000000          MOV     #13,RHCS1(R0)   ;RELEASE THE DRIVE
(1)  014560  104017                          ERROR   17              ;TYPE ERROR MESSAGE 17
(1)  014562  013737  001162  001126  66$:    MOV     $TMP2,$BDDAT    ;LOOK FOR BIT FAILURES WHEN RHDS1 READ
(1)  014570  013737  001216  001226          MOV     PORTA,PTNBR     ;CHANGE PORT NUMBER
(1)  014576  023737  001124  001162          CMP     $GDDAT,$TMP2    ;ALL BITS OK ?
(1)  014604  001401                          BEQ     67$             ;BR IF OK FROM PORT A.
(1)  014606  104037                          ERROR   37              ;REPORT ERROR
(1)  014610  013737  001164  001126  67$:    MOV     $TMP3,$BDDAT    ;CHECK RHDS1 FOR BIT FAILURES - FROM PORT B.
(1)  014616  013737  001220  001226          MOV     PORTB,PTNBR     ;CHANGE PORT NUMBER
(1)  014624  023737  001124  001164          CMP     $GDDAT,$TMP3    ;SEE IF READ OK FROM PORT B.
(1)  014632  001401                          BEQ     68$             ;BR IF OK
(1)  014634  104037                          ERROR   37              ;REPORT THE ERROR
(1)  014636  000240                  68$:    NOP
5227 014640  104400  023061                  TYPE    ,SWTCHB         ;SWITCH TO 'B'
5228 014644  104400  023144                  TYPE    ,CONTUE         ;PRESS 'CONTINUE'
5229 014650  000000                          HALT
5230
(1)                                          ;VERIFY THAT THE DRIVE IS IN NEUTRAL
(1)
(1)  014652  005037  001242                  CLR     RELERR          ;CLEAR THE 'RELEASE ERROR ' INDICATOR
(1)  014656  012737  000012  001122          MOV     #RHDS1,$BDADR   ;FORM THE ADDRESS OF RHDS1 FOR TYPEOUT
(1)  014664  060037  001122                  ADD     R0,$BDADR       ;ADD THE I/O BASE ADDRESS
(1)  014670  012737  011700  001124          MOV     #MOL!PGM!DPR!DRY!VV,$GDDAT  ;COMPARISON CONSTANT
(1)  014676  113760  001216  000010          MOVB    PORTA,RHCS2(R0) ;SELECT PORT A.
(1)  014704  016037  000012  001162          MOV     RHDS1(R0),$TMP2 ;GET THE DRIVE STATUS REGISTER FROM PORT A.
(1)  014712  013737  001162  001156          MOV     $TMP2,$TMP0     ;COPY IT INTO '$TMP0'
(1)  014720  042737  100100  001156          BIC     #ATA!VV,$TMP0   ;CLEAR PORT DEPENDENT BITS FROM THE COPY
(1)  014726  113760  001220  000010          MOVB    PORTB,RHCS2(R0) ;SELECT PORT B.
(1)  014734  016037  000012  001164          MOV     RHDS1(R0),$TMP3 ;GET THE DRIVE STATUS REGISTER FROM PORT B.
(1)  014742  013737  001164  001160          MOV     $TMP3,$TMP1     ;COPY IT INTO '$TMP1'
(1)  014750  042737  100100  001160          BIC     #ATA!VV,$TMP1   ;CLEAR PORT DEPENDENT BITS FROM THE COPY
(1)  014756  023737  001156  001160          CMP     $TMP0,$TMP1     ;IS THE STATUS REGISTER THE SAME FROM BOTH PORTS ?
(1)  014764  001006                          BNE     69$             ;BR IF NOT
(1)  014766  005737  001156                  TST     $TMP0           ;REGISTERS ARE THE SAME: ARE THEY ZERO ?
(1)  014772  001045                          BNE     71$             ;BR IF NOT
(1)  014774  104034                          ERROR   34              ;REPORT DRIVE NOT IN NEUTRAL OR NOT SEIZED
(1)  014776  000137  015162                  JMP     73$             ;BYPASS THE REST OF THE CHECKS
(1)  015002  013737  001162  001126  69$:    MOV     $TMP2,$BDDAT    ;SET UP POSSIBLE BAD DATA FOR ERROR MESSAGE
(1)  015010  013737  001220  001226          MOV     PORTB,PTNBR     ;SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
```

```
    (1)  015016  113760  001220  000010          MOVB    PORTB,RHCS2(R0) ;SELECT PORT B.
    (1)  015024  005737  001156                  TST     $TMP0           ;SEE IF STATUS EQ 0 FROM PORT A.
    (1)  015030  001414                          BEQ     70$             ;BR IF ZERO
    (1)  015032  013737  001216  001226          MOV     PORTA,PTNBR     ;SEIZING PORT IF TEST SHOWS DRIVE NOT IN NEUTRAL
    (1)  015040  013737  001164  001126          MOV     $TMP3,$BDDAT    ;'BAD DATA' FOR ERROR TYPE OUT
    (1)  015046  113760  001216  000010          MOVB    PORTA,RHCS2(R0) ;SELECT PORT A.
    (1)  015054  005737  001160                  TST     $TMP1           ;SEE IF STATUS EQ ZERO FROM PORT B.
    (1)  015060  001012                          BNE     71$             ;BR IF NOT
    (1)  015062  012737  177777  001242  70$:    MOV     #-1,RELERR      ;SET 'RELEASE ERROR' INDICATOR
    (1)  015070  012760  000011  000000          MOV     #11,RHCS1(R0)   ;CLEAR THE DRIVE
    (1)  015076  012760  000013  000000          MOV     #13,RHCS1(R0)   ;RELEASE THE DRIVE
    (1)  015104  104020                          ERROR   20              ;TYPE ERROR MESSAGE 20
    (1)  015106  013737  001162  001126  71$:    MOV     $TMP2,$BDDAT    ;LOOK FOR BIT FAILURES WHEN RHDS1 READ
    (1)  015114  013737  001216  001226          MOV     PORTA,PTNBR     ;CHANGE PORT NUMBER
    (1)  015122  023737  001124  001162          CMP     $GDDAT,$TMP2    ;ALL BITS OK ?
    (1)  015130  001401                          BEQ     72$             ;BR IF OK FROM PORT A.
    (1)  015132  104037                          ERROR   37              ;REPORT ERROR
    (1)  015134  013737  001164  001126  72$:    MOV     $TMP3,$BDDAT    ;CHECK RHDS1 FOR BIT FAILURES - FROM PORT B.
    (1)  015142  013737  001220  001226          MOV     PORTB,PTNBR     ;CHANGE PORT NUMBER
    (1)  015150  023737  001124  001164          CMP     $GDDAT,$TMP3    ;SEE IF READ OK FROM PORT B.
    (1)  015156  001401                          BEQ     73$             ;BR IF OK
    (1)  015160  104037                          ERROR   37              ;REPORT THE ERROR
    (1)  015162  000240                  73$:    NOP
 5231    015164  104400  022707                  TYPE    ,SWTCHN         ;RETURN SWITCH TO 'A/B'
 5232
    (1)                                          ;IF ERROR OCCURED, CHECK FOR LOOP ON TEST
    (1)
    (1)  015170  105737  001103                  TSTB    $ERFLG          ;DID AN ERROR OCCUR ?
    (3)  015174  001412                          BEQ     TST13           ;;BR IF NOT
    (1)  015176  032737  001000  177570          BIT     #SW09,SWR       ;SEE IF LOOP ON ERROR SET (SWR9=1)
    (3)  015204  001406                          BEQ     TST13           ;;BR IF NOT
    (1)  015206  105037  001103                  CLRB    $ERFLG          ;CLEAR THE ERROR FLAG
    (1)  015212  005037  001170                  CLR     $TIMES          ;CLEAR THE MAX ITERATION COUNT
    (1)  015216  000177  163666                  JMP     @$LPERR         ;GO TO THE LOOP ADDRESS
 5233
 5365
 5366
    (5)                                  ;*****************************************************************
    (4)                                  ;*TEST 13       TEST 'CONTROLLER SELECT' SWITCH LOCKED ON PORT A
    (5)                                  ;*
    (5)                                  ;*TEST THE OPERATION OF THE 'CONTROLLER SELECT' SWITCH (DRIVE CYCLED DOWN).
    (5)                                  ;*
    (5)                                  ;*  A.  CYCLE THE DRIVE DOWN.
    (5)                                  ;*
    (5)                                  ;*  B.  SWITCH TO CONTROLLER A POSITION.  VERIFY THAT THE DRIVE IS IN
    (5)                                  ;*      NEUTRAL AND THAT THE STATUS BITS IN RHDS1, AS READ THROUGH BOTH
    (5)                                  ;*      PORTS, ARE CORRECT.
    (5)                                  ;*
    (5)                                  ;*  C.  SWITCH THE 'CONTROLLER SELECT' SWITCH TO A; CYCLE THE DRIVE UP.
    (5)                                  ;*
    (5)                                  ;*  D.  WHEN THE DRIVE CYCLES UP, VERIFY THAT 'VV-A IS RESET, AND
    (5)                                  ;*      THAT 'ATA-A IS SET.
    (5)                                  ;*
    (5)                                  ;*  E.  ISSUE A DRIVE CLEAR COMMAND AND A READIN PRESET COMMAND THROUGH
```

```
(5)                                     ;*      PORT A.
(5)                                     ;*
(5)                                     ;*  F.  VERIFY THAT THE DRIVE CANNOT BE ACCESSED THROUGH PORT B AND
(5)                                     ;*      'NED' DOES NOT SET WHEN ATEMPTING TO ACCESS THE DRIVE THROUGH
(5)                                     ;*      PORT B.  ATTEMPT TO SET PORT REQUEST BY WRITING 0'S
(5)                                     ;*      INTO RHDS1 THROUGH PORT B.
(5)                                     ;*
(5)                                     ;*  G.  ISSUE A RELEASE COMMAND THROUGH PORT A.  VERIFY THAT THE
(5)                                     ;*      DRIVE REMAINS LOCKED ON PORT A.
(5)                                     ;*
(4)                                     ;*************************************************************
(4)  015222                             TST13:
(4)  015222  000004                             SCOPE                   ;INITIALIZE THE SCOPE HANDLER
(4)  015224  005737  001260                     TST     KYBCTL          ;PERFORMING ONLY SINGLE TESTS ?
(4)  015230  001406                             BEQ     2$              ;BR IF NOT
(4)  015232  100002                             BPL     1$              ;BR IF JUST ENTERED TEST
(4)  015234  000137  002262                     JMP     EXEC            ;RETURN & GET NEXT TEST NUMBER
(4)  015240  012737  177777  001260  1$:       MOV     #-1,KYBCTL      ;SET SINGLE TEST INDICATOR
(4)  015246  112737  000013  001102  2$:       MOVB    #13,$TSTNM      ;TEST NUMBER
(4)  015254  012737  015276  001106             MOV     #TEST13,$LPADR  ;LOAD LOOP ON TEST ADDRESS
(4)  015262  012737  015276  001110             MOV     #TEST13,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
(2)  015270  012737  000001  001170             MOV     #1,$TIMES       ;;DO 1 ITERATION
(1)
(2)
(3)                                     ;;*************************************************************************
(2)                                     ;END OF 'SCOPE' SETUP - START OF MAIN TEST
(2)
(2)  015276                             TEST13:
(2)  015276  113760  001216  000010             MOVB    PORTA,RHCS2(R0) ;SELECT PORT A
(2)  015304  013737  001216  001226             MOV     PORTA,PTNBR     ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
(1)  015312  104400  023213                     TYPE    ,CYCLED         ;'CYCLE DOWN THE DRIVE'
(1)
(2)                                     ;;*************************************************************************
(1)                                     ;WAIT FOR 'MOL' TO RESET
(1)
(1)  015316  032760  010000  000012  1$:       BIT     #MOL,RHDS1(R0)  ;TEST 'MOL'
(1)  015324  001374                             BNE     1$              ;BR IF IT IS STILL SET
(1)  015326  104400  022776                     TYPE    ,SWTCHA         ;SWITCH TO 'A'
(1)  015332  104400  023235                     TYPE    ,CYCLEU         ;'CYCLE UP THE DRIVE'
(1)
(2)                                     ;;*************************************************************************
(1)                                     ;WAIT FOR DRIVE TO CYCLE UP AFTER SWITCH CHANGED
(1)
(1)  015336  032760  010000  000012  2$:       BIT     #MOL,RHDS1(R0)  ;TEST 'MOL' AGAIN
(1)  015344  001774                             BEQ     2$              ;BR IF NOT SET
(1)
(2)                                     ;;*************************************************************************
(1)                                     ;DRIVE IS CYCLED UP, CHECK STATUS THROUGH PORT A
(1)
(2)  015346  005037  001236                     CLR     CKERR           ;CLEAR THE 'CHECK ERROR' INDICATOR
(2)  015352  016037  000012  001126             MOV     RHDS1(R0),$BDDAT ;GET CONTENTS OF RHDS1
(2)  015360  012737  000012  001122             MOV     #RHDS1,$BDADR   ;FORM REGISTER ADDRESS OF ERROR MESSAGE
(2)  015366  060037  001122                     ADD     R0,$BDADR       ;ADD RH11 BASE ADDRESS
(2)  015372  012737  110600  001124             MOV     #ATA!MOL!DPR!DRY,$GDDAT ;WHAT REGISTER SHOULD BE
```

```
(2) 015400 013737 001126 001156          MOV     $BDDAT,$TMP0    ;MOVE REGISTER CONTENTS TO '$TMP0'
(2) 015406 042737 066077 001156          BIC     #^C111700,$TMP0  ;SAVE SPECIFIED BITS
(2) 015414 023737 001124 001156          CMP     $GDDAT,$TMP0    ;COMPARE THE BITS
(2) 015422 001414                        BEQ     64$             ;BR IF OK
(2) 015424 013737 001126 001166          MOV     $BDDAT,$TMP4    ;COPY 'BAD DATA'
(2) 015432 042737 111700 001166          BIC     #111700,$TMP4   ;CLEAR THE MASKED BITS
(2) 015440 053737 001166 001124          BIS     $TMP4,$GDDAT    ;'OR' WITH GOOD DATA FOR TYPEOUT
(2) 015446 104021                        ERROR   21              ;TYPE MESSAGE 21
(2) 015450 005137 001236                 COM     CKERR           ;SET THE REGISTER COMPARE ERROR INDICATOR
(2) 015454 000240                64$:    NOP
(1)
(2)                                      ;;*****************************************************************************
(1)                                      ;SET VOLUME VALID FOR PORT A
(1)
(1) 015456 012760 000011 000000          MOV     #11,RHCS1(R0)   ;ISSUE A DRIVE CLEAR
(1) 015464 012760 000021 000000          MOV     #21,RHCS1(R0)   ;ISSUE A READIN PRESET
(1) 015472 012760 010000 000032          MOV     #FMT22,RHOF(R0) ;SET FMT22
(1)
(2)                                      ;;*****************************************************************************
(1)                                      ;CHECK THE DRIVE STATUS THROUGH PORT B; VERIFY THAT 'NED' DOES NOT
(1)                                      ;       SET WHEN THE DRIVE IS ACCESSED THROUGH PORT B.
(1)
(2) 015500 113760 001220 000010          MOVB    PORTB,RHCS2(R0)  ;SELECT PORT B
(2) 015506 013737 001220 001226          MOV     PORTB,PTNBR  ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
(2) 015514 005037 001236                 CLR     CKERR           ;CLEAR THE 'CHECK ERROR' INDICATOR
(2) 015520 016037 000012 001126          MOV     RHDS1(R0),$BDDAT  ;GET CONTENTS OF RHDS1
(2) 015526 012737 000012 001122          MOV     #RHDS1,$BDADR   ;FORM REGISTER ADDRESS OF ERROR MESSAGE
(2) 015534 060037 001122                 ADD     R0,$BDADR       ;ADD RH11 BASE ADDRESS
(2) 015540 005037 001124                 CLR     $GDDAT          ;WHAT REGISTER SHOULD BE
(2) 015544 013737 001126 001156          MOV     $BDDAT,$TMP0    ;MOVE REGISTER CONTENTS TO '$TMP0'
(2) 015552 042737 000077 001156          BIC     #^C177700,$TMP0  ;SAVE SPECIFIED BITS
(2) 015560 023737 001124 001156          CMP     $GDDAT,$TMP0    ;COMPARE THE BITS
(2) 015566 001414                        BEQ     65$             ;BR IF OK
(2) 015570 013737 001126 001166          MOV     $BDDAT,$TMP4    ;COPY 'BAD DATA'
(2) 015576 042737 177700 001166          BIC     #177700,$TMP4   ;CLEAR THE MASKED BITS
(2) 015604 053737 001166 001124          BIS     $TMP4,$GDDAT    ;'OR' WITH GOOD DATA FOR TYPEOUT
(2) 015612 104022                        ERROR   22              ;TYPE MESSAGE 22
(2) 015614 005137 001236                 COM     CKERR           ;SET THE REGISTER COMPARE ERROR INDICATOR
(2) 015620 000240                65$:    NOP
(2) 015622 005037 001236                 CLR     CKERR           ;CLEAR THE 'CHECK ERROR' INDICATOR
(2) 015626 016037 000010 001126          MOV     RHCS2(R0),$BDDAT  ;GET CONTENTS OF RHCS2
(2) 015634 012737 000010 001122          MOV     #RHCS2,$BDADR   ;FORM REGISTER ADDRESS OF ERROR MESSAGE
(2) 015642 060037 001122                 ADD     R0,$BDADR       ;ADD RH11 BASE ADDRESS
(2) 015646 005037 001124                 CLR     $GDDAT          ;WHAT REGISTER SHOULD BE
(2) 015652 013737 001126 001156          MOV     $BDDAT,$TMP0    ;MOVE REGISTER CONTENTS TO '$TMP0'
(2) 015660 042737 167777 001156          BIC     #^CNED,$TMP0  ;SAVE SPECIFIED BITS
(2) 015666 023737 001124 001156          CMP     $GDDAT,$TMP0    ;COMPARE THE BITS
(2) 015674 001414                        BEQ     66$             ;BR IF OK
(2) 015676 013737 001126 001166          MOV     $BDDAT,$TMP4    ;COPY 'BAD DATA'
(2) 015704 042737 010000 001166          BIC     #NED,$TMP4      ;CLEAR THE MASKED BITS
(2) 015712 053737 001166 001124          BIS     $TMP4,$GDDAT    ;'OR' WITH GOOD DATA FOR TYPEOUT
(2) 015720 104023                        ERROR   23              ;TYPE MESSAGE 23
(2) 015722 005137 001236                 COM     CKERR           ;SET THE REGISTER COMPARE ERROR INDICATOR
(2) 015726 000240                66$:    NOP
```

```
(1)  015730  005060  000012                    CLR     RHDS1(R0)       ;TRY TO SET REQUEST BY WRITING THROUGH
(1)                                                                      ;THE LOCKED OUT PORT (PORT 'B')
(1)
(2)                                     ;;*************************************************************************
(1)                                     ;VERIFY THAT DRIVE STAYS LOCKED ON PORT A
(1)
(2)  015734  113760  001216  000010          MOVB    PORTA,RHCS2(R0)  ;SELECT PORT A
(2)  015742  013737  001216  001226          MOV     PORTA,PTNBR  ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
(1)  015750  012760  000013  000012          MOV     #13,RHDS1(R0)    ;ISSUE A RELEASE THROUGH PORT A
(1)  015756  013737  001216  001230          MOV     PORTA,SEIZPT     ;ADDRESS OF 'LOCKED ON' PORT
(2)  015764  113760  001220  000010          MOVB    PORTB,RHCS2(R0)  ;SELECT PORT B
(2)  015772  013737  001220  001226          MOV     PORTB,PTNBR  ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
(2)  016000  005037  001236                  CLR     CKERR            ;CLEAR THE 'CHECK ERROR' INDICATOR
(2)  016004  016037  000012  001126          MOV     RHDS1(R0),$BDDAT  ;GET CONTENTS OF RHDS1
(2)  016012  012737  000012  001122          MOV     #RHDS1,$BDADR  ;FORM REGISTER ADDRESS OF ERROR MESSAGE
(2)  016020  060037  001122                  ADD     R0,$BDADR        ;ADD RH11 BASE ADDRESS
(2)  016024  005037  001124                  CLR     $GDDAT           ;WHAT REGISTER SHOULD BE
(2)  016030  013737  001126  001156          MOV     $BDDAT,$TMP0     ;MOVE REGISTER CONTENTS TO '$TMP0'
(2)  016036  042737  000077  001156          BIC     #^C177700,$TMP0  ;SAVE SPECIFIED BITS
(2)  016044  023737  001124  001156          CMP     $GDDAT,$TMP0     ;COMPARE THE BITS
(2)  016052  001414                          BEQ     67$              ;BR IF OK
(2)  016054  013737  001126  001166          MOV     $BDDAT,$TMP4     ;COPY 'BAD DATA'
(2)  016062  042737  177700  001166          BIC     #177700,$TMP4    ;CLEAR THE MASKED BITS
(2)  016070  053737  001166  001124          BIS     $TMP4,$GDDAT     ;'OR' WITH GOOD DATA FOR TYPEOUT
(2)  016076  104024                          ERROR   24               ;TYPE MESSAGE 24
(2)  016100  005137  001236                  COM     CKERR            ;SET THE REGISTER COMPARE ERROR INDICATOR
(2)  016104  000240                  67$:    NOP
(1)
(1)                                     ;IF ERROR OCCURED, CHECK FOR LOOP ON TEST
(1)
(1)  016106  105737  001103                  TSTB    $ERFLG           ;DID AN ERROR OCCUR
(1)  016112  001412                          BEQ     3$               ;BR IF NOT
(1)  016114  032737  001000  177570          BIT     #SW09,SWR        ;SEE IF LOOP ON ERROR (SWR9=1)
(1)  016122  001406                          BEQ     3$               ;BR IF NOT
(1)  016124  105037  001103                  CLRB    $ERFLG           ;CLEAR THE ERROR FLAG
(1)  016130  005037  001170                  CLR     $TIMES           ;CLEAR THE MAX ITERATION COUNT
(1)  016134  000177  162750                  JMP     @$LPERR          ;GO TO THE LOOP ADDRESS
(1)  016140  005737  001260          3$:     TST     KYBCTL           ;IN SINGLE TEST MODE ?
(1)  016144  001460                          BEQ     6$               ;BR IF NOT
(1)  016146  032737  040000  177570          BIT     #SW14,SWR        ;LOOP ON TEST ?
(1)  016154  001054                          BNE     6$               ;BR IF LOOPING
(1)  016156  104400  023213                  TYPE    ,CYCLED          ;TYPE 'CYCLE DOWN'
(1)  016162  104400  022707                  TYPE    ,SWTCHN          ;'SWITCH TO A/B'
(1)  016166  104400  023235                  TYPE    ,CYCLEU          ;'CYCLE THE DRIVE UP'
(2)  016172  113760  001216  000010          MOVB    PORTA,RHCS2(R0)  ;SELECT PORT A
(2)  016200  013737  001216  001226          MOV     PORTA,PTNBR  ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
(1)  016206  032760  010000  000012  4$:     BIT     #MOL,RHDS1(R0)   ;CHECK 'MOL'
(1)  016214  001374                          BNE     4$               ;BR IF SET (DRIVE NOT CYCLED DOWN)
(1)  016216  032760  010000  000012  5$:     BIT     #MOL,RHDS1(R0)   ;CHECK 'MOL' AGAIN
(1)  016224  001774                          BEQ     5$               ;BR IF NOT SET (DRIVE NOT CYCLED UP)
(1)
(2)                                     ;;*************************************************************************
(1)                                     ;SET VOLUME VALID FOR BOTH PORTS
(1)
```

```
(1)  016226  012760  000011  000000          MOV     #11,RHCS1(R0)   ;ISSUE A DRIVE CLEAR THROUGH PORT A
(1)  016234  012760  000021  000000          MOV     #21,RHCS1(R0)   ;ISSUE A READIN PRESET THROUGH PORT A
(1)  016242  012760  000013  000000          MOV     #13,RHCS1(R0)   ;RELEASE PORT A
(2)  016250  113760  001220  000010          MOVB    PORTB,RHCS2(R0) ;SELECT PORT B
(2)  016256  013737  001220  001226          MOV     PORTB,PTNBR  ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
(1)  016264  012760  000021  000000          MOV     #21,RHCS1(R0)   ;ISSUE A READIN PRESET THROUGH PORT B
(1)  016272  012760  010000  000032          MOV     #FMT22,RHOF(R0) ;SET FMT22
(1)  016300  012760  000013  000000          MOV     #13,RHCS1(R0)   ;RELEASE PORT B
(1)  016306  104400  001201          6$:    TYPE    ,$CRLF          ;CR-LF
(3)  016312  000400                         BR      TST14           ;;GO TO NEXT TEST
5367
(5)                                 ;*************************************************************
(4)                                 ;*TEST 14       TEST 'CONTROLLER SELECT' SWITCH LOCKED ON PORT B
(5)                                 ;*
(5)                                 ;*TEST THE OPERATION OF THE 'CONTROLLER SELECT' SWITCH (DRIVE CYCLED DOWN).
(5)                                 ;*
(5)                                 ;*  A.  CYCLE THE DRIVE DOWN.
(5)                                 ;*
(5)                                 ;*  B.  SWITCH TO CONTROLLER B POSITION.  VERIFY THAT THE DRIVE IS IN
(5)                                 ;*      NEUTRAL AND THAT THE STATUS BITS IN RHDS1, AS READ THROUGH BOTH
(5)                                 ;*      PORTS, ARE CORRECT.
(5)                                 ;*
(5)                                 ;*  C.  SWITCH THE 'CONTROLLER SELECT' SWITCH TO B; CYCLE THE DRIVE UP.
(5)                                 ;*
(5)                                 ;*  D.  WHEN THE DRIVE CYCLES UP, VERIFY THAT 'VV-B IS RESET, AND
(5)                                 ;*      THAT 'ATA-B IS SET.
(5)                                 ;*
(5)                                 ;*  E.  ISSUE A DRIVE CLEAR COMMAND AND A READIN PRESET COMMAND THROUGH
(5)                                 ;*      PORT B.
(5)                                 ;*
(5)                                 ;*  F.  VERIFY THAT THE DRIVE CANNOT BE ACCESSED THROUGH PORT A AND
(5)                                 ;*      'NED' DOES NOT SET WHEN ATEMPTING TO ACCESS THE DRIVE THROUGH
(5)                                 ;*      PORT A.  ATTEMPT TO SET PORT REQUEST BY WRITING 0'S
(5)                                 ;*      INTO RHDS1 THROUGH PORT A.
(5)                                 ;*
(5)                                 ;*  G.  ISSUE A RELEASE COMMAND THROUGH PORT B.  VERIFY THAT THE
(5)                                 ;*      DRIVE REMAINS LOCKED ON PORT B.
(5)                                 ;*
(5)                                 ;*  H.  CYCLE THE DRIVE DOWN.  CHANGE THE 'CONTROLLER SELECT' SWITCH TO
(5)                                 ;*      A/B; CYCLE THE DRIVE UP.
(5)                                 ;*
(5)                                 ;*  I.  VERIFY THAT BOTH PORTS CAN ACCESS THE DRIVE, THAT BOTH ATTENTION
(5)                                 ;*      BITS ARE SET, AND THAT BOTH 'VV' BITS ARE RESET.
(5)                                 ;*
(4)                                 ;*************************************************************
(4)  016314                         TST14:
(4)  016314  000004                         SCOPE                   ;INITIALIZE THE SCOPE HANDLER
(4)  016316  005737  001260                 TST     KYBCTL          ;PERFORMING ONLY SINGLE TESTS ?
(4)  016322  001406                         BEQ     2$              ;BR IF NOT
(4)  016324  100002                         BPL     1$              ;BR IF JUST ENTERED TEST
(4)  016326  000137  002262                 JMP     EXEC            ;RETURN & GET NEXT TEST NUMBER
(4)  016332  012737  177777  001260  1$:    MOV     #-1,KYBCTL      ;SET SINGLE TEST INDICATOR
(4)  016340  112737  000014  001102  2$:    MOVB    #14,$TSTNM      ;TEST NUMBER
(4)  016346  012737  016370  001106          MOV     #TEST14,$LPADR  ;LOAD LOOP ON TEST ADDRESS
```

```
(4)  016354  012737  016370  001110          MOV     #TEST14,$LPERR  ;LOAD LOOP ON ERROR ADDRESS
(2)  016362  012737  000001  001170          MOV     #1,$TIMES       ;;DO 1 ITERATION
(1)
(2)
(3)                                  ;;*************************************************************************
(2)                                  ;END OF 'SCOPE' SETUP - START OF MAIN TEST
(2)
(2)  016370                          TEST14:
(2)  016370  113760  001220  000010          MOVB    PORTB,RHCS2(R0)  ;SELECT PORT B
(2)  016376  013737  001220  001226          MOV     PORTB,PTNBR  ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
(1)  016404  104400  023213                  TYPE    ,CYCLED          ;'CYCLE DOWN THE DRIVE'
(1)
(2)                                  ;;*************************************************************************
(1)                                  ;WAIT FOR 'MOL' TO RESET
(1)
(1)  016410  032760  010000  000012  1$:     BIT     #MOL,RHDS1(R0)  ;TEST 'MOL'
(1)  016416  001374                          BNE     1$              ;BR IF IT IS STILL SET
(1)  016420  104400  023061                  TYPE    ,SWTCHB         ;SWITCH TO 'B'
(1)  016424  104400  023235                  TYPE    ,CYCLEU         ;'CYCLE UP THE DRIVE'
(1)
(2)                                  ;;*************************************************************************
(1)                                  ;WAIT FOR DRIVE TO CYCLE UP AFTER SWITCH CHANGED
(1)
(1)  016430  032760  010000  000012  2$:     BIT     #MOL,RHDS1(R0)  ;TEST 'MOL' AGAIN
(1)  016436  001774                          BEQ     2$              ;BR IF NOT SET
(1)
(2)                                  ;;*************************************************************************
(1)                                  ;DRIVE IS CYCLED UP, CHECK STATUS THROUGH PORT B
(1)
(2)  016440  005037  001236                  CLR     CKERR           ;CLEAR THE 'CHECK ERROR' INDICATOR
(2)  016444  016037  000012  001126          MOV     RHDS1(R0),$BDDAT  ;GET CONTENTS OF RHDS1
(2)  016452  012737  000012  001122          MOV     #RHDS1,$BDADR   ;FORM REGISTER ADDRESS OF ERROR MESSAGE
(2)  016460  060037  001122                  ADD     R0,$BDADR       ;ADD RH11 BASE ADDRESS
(2)  016464  012737  110600  001124          MOV     #ATA!MOL!DPR!DRY,$GDDAT  ;WHAT REGISTER SHOULD BE
(2)  016472  013737  001126  001156          MOV     $BDDAT,$TMP0    ;MOVE REGISTER CONTENTS TO '$TMP0'
(2)  016500  042737  066077  001156          BIC     #^C111700,$TMP0  ;SAVE SPECIFIED BITS
(2)  016506  023737  001124  001156          CMP     $GDDAT,$TMP0    ;COMPARE THE BITS
(2)  016514  001414                          BEQ     64$             ;BR IF OK
(2)  016516  013737  001126  001166          MOV     $BDDAT,$TMP4    ;COPY 'BAD DATA'
(2)  016524  042737  111700  001166          BIC     #111700,$TMP4   ;CLEAR THE MASKED BITS
(2)  016532  053737  001166  001124          BIS     $TMP4,$GDDAT    ;'OR' WITH GOOD DATA FOR TYPEOUT
(2)  016540  104021                          ERROR   21              ;TYPE MESSAGE 21
(2)  016542  005137  001236                  COM     CKERR           ;SET THE REGISTER COMPARE ERROR INDICATOR
(2)  016546  000240                  64$:    NOP
(1)
(2)                                  ;;*************************************************************************
(1)                                  ;SET VOLUME VALID FOR PORT B
(1)
(1)  016550  012760  000011  000000          MOV     #11,RHCS1(R0)   ;ISSUE A DRIVE CLEAR
(1)  016556  012760  000021  000000          MOV     #21,RHCS1(R0)   ;ISSUE A READIN PRESET
(1)  016564  012760  010000  000032          MOV     #FMT22,RHOF(R0) ;SET FMT22
(1)
(2)                                  ;;*************************************************************************
(1)                                  ;CHECK THE DRIVE STATUS THROUGH PORT A; VERIFY THAT 'NED' DOES NOT
```

```
(1)                                     ;       SET WHEN THE DRIVE IS ACCESSED THROUGH PORT A.
(1)
(2)  016572  113760  001216  000010             MOVB    PORTA,RHCS2(R0)  ;SELECT PORT A
(2)  016600  013737  001216  001226             MOV     PORTA,PTNBR  ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
(2)  016606  005037  001236                     CLR     CKERR           ;CLEAR THE 'CHECK ERROR' INDICATOR
(2)  016612  016037  000012  001126             MOV     RHDS1(R0),$BDDAT  ;GET CONTENTS OF RHDS1
(2)  016620  012737  000012  001122             MOV     #RHDS1,$BDADR  ;FORM REGISTER ADDRESS OF ERROR MESSAGE
(2)  016626  060037  001122                     ADD     R0,$BDADR       ;ADD RH11 BASE ADDRESS
(2)  016632  005037  001124                     CLR     $GDDAT          ;WHAT REGISTER SHOULD BE
(2)  016636  013737  001126  001156             MOV     $BDDAT,$TMP0    ;MOVE REGISTER CONTENTS TO '$TMP0'
(2)  016644  042737  000077  001156             BIC     #^C177700,$TMP0  ;SAVE SPECIFIED BITS
(2)  016652  023737  001124  001156             CMP     $GDDAT,$TMP0    ;COMPARE THE BITS
(2)  016660  001414                             BEQ     65$             ;BR IF OK
(2)  016662  013737  001126  001166             MOV     $BDDAT,$TMP4    ;COPY 'BAD DATA'
(2)  016670  042737  177700  001166             BIC     #177700,$TMP4   ;CLEAR THE MASKED BITS
(2)  016676  053737  001166  001124             BIS     $TMP4,$GDDAT    ;'OR' WITH GOOD DATA FOR TYPEOUT
(2)  016704  104022                             ERROR   22              ;TYPE MESSAGE 22
(2)  016706  005137  001236                     COM     CKERR           ;SET THE REGISTER COMPARE ERROR INDICATOR
(2)  016712  000240                     65$:    NOP
(2)  016714  005037  001236                     CLR     CKERR           ;CLEAR THE 'CHECK ERROR' INDICATOR
(2)  016720  016037  000010  001126             MOV     RHCS2(R0),$BDDAT  ;GET CONTENTS OF RHCS2
(2)  016726  012737  000010  001122             MOV     #RHCS2,$BDADR  ;FORM REGISTER ADDRESS OF ERROR MESSAGE
(2)  016734  060037  001122                     ADD     R0,$BDADR       ;ADD RH11 BASE ADDRESS
(2)  016740  005037  001124                     CLR     $GDDAT          ;WHAT REGISTER SHOULD BE
(2)  016744  013737  001126  001156             MOV     $BDDAT,$TMP0    ;MOVE REGISTER CONTENTS TO '$TMP0'
(2)  016752  042737  167777  001156             BIC     #^CNED,$TMP0  ;SAVE SPECIFIED BITS
(2)  016760  023737  001124  001156             CMP     $GDDAT,$TMP0    ;COMPARE THE BITS
(2)  016766  001414                             BEQ     66$             ;BR IF OK
(2)  016770  013737  001126  001166             MOV     $BDDAT,$TMP4    ;COPY 'BAD DATA'
(2)  016776  042737  010000  001166             BIC     #NED,$TMP4      ;CLEAR THE MASKED BITS
(2)  017004  053737  001166  001124             BIS     $TMP4,$GDDAT    ;'OR' WITH GOOD DATA FOR TYPEOUT
(2)  017012  104023                             ERROR   23              ;TYPE MESSAGE 23
(2)  017014  005137  001236                     COM     CKERR           ;SET THE REGISTER COMPARE ERROR INDICATOR
(2)  017020  000240                     66$:    NOP
(1)  017022  005060  000012                     CLR     RHDS1(R0)       ;TRY TO SET REQUEST BY WRITING THROUGH
(1)                                             ;THE LOCKED OUT PORT (PORT 'A')
(1)
(2)                                     ;;*************************************************************************
(1)                                     ;VERIFY THAT DRIVE STAYS LOCKED ON PORT B
(1)
(2)  017026  113760  001220  000010             MOVB    PORTB,RHCS2(R0)  ;SELECT PORT B
(2)  017034  013737  001220  001226             MOV     PORTB,PTNBR  ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
(1)  017042  012760  000013  000012             MOV     #13,RHDS1(R0)   ;ISSUE A RELEASE THROUGH PORT B
(1)  017050  013737  001220  001230             MOV     PORTB,SEIZPT    ;ADDRESS OF 'LOCKED ON' PORT
(2)  017056  113760  001216  000010             MOVB    PORTA,RHCS2(R0)  ;SELECT PORT A
(2)  017064  013737  001216  001226             MOV     PORTA,PTNBR  ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
(2)  017072  005037  001236                     CLR     CKERR           ;CLEAR THE 'CHECK ERROR' INDICATOR
(2)  017076  016037  000012  001126             MOV     RHDS1(R0),$BDDAT  ;GET CONTENTS OF RHDS1
(2)  017104  012737  000012  001122             MOV     #RHDS1,$BDADR  ;FORM REGISTER ADDRESS OF ERROR MESSAGE
(2)  017112  060037  001122                     ADD     R0,$BDADR       ;ADD RH11 BASE ADDRESS
(2)  017116  005037  001124                     CLR     $GDDAT          ;WHAT REGISTER SHOULD BE
(2)  017122  013737  001126  001156             MOV     $BDDAT,$TMP0    ;MOVE REGISTER CONTENTS TO '$TMP0'
(2)  017130  042737  000077  001156             BIC     #^C177700,$TMP0  ;SAVE SPECIFIED BITS
(2)  017136  023737  001124  001156             CMP     $GDDAT,$TMP0    ;COMPARE THE BITS
```

```
(2)  017144  001414                          BEQ     67$                ;BR IF OK
(2)  017146  013737  001126  001166          MOV     $BDDAT,$TMP4       ;COPY 'BAD DATA'
(2)  017154  042737  177700  001166          BIC     #177700,$TMP4      ;CLEAR THE MASKED BITS
(2)  017162  053737  001166  001124          BIS     $TMP4,$GDDAT       ;'OR' WITH GOOD DATA FOR TYPEOUT
(2)  017170  104024                          ERROR   24                 ;TYPE MESSAGE 24
(2)  017172  005137  001236                  COM     CKERR              ;SET THE REGISTER COMPARE ERROR INDICATOR
(2)  017176  000240                  67$:    NOP
(1)
(1)                                ;IF ERROR OCCURED, CHECK FOR LOOP ON TEST
(1)
(1)  017200  105737  001103                  TSTB    $ERFLG             ;DID AN ERROR OCCUR
(1)  017204  001412                          BEQ     3$                 ;BR IF NOT
(1)  017206  032737  001000  177570          BIT     #SW09,SWR          ;SEE IF LOOP ON ERROR (SWR9=1)
(1)  017214  001406                          BEQ     3$                 ;BR IF NOT
(1)  017216  105037  001103                  CLRB    $ERFLG             ;CLEAR THE ERROR FLAG
(1)  017222  005037  001170                  CLR     $TIMES             ;CLEAR THE MAX ITERATION COUNT
(1)  017226  000177  161656                  JMP     @$LPERR            ;GO TO THE LOOP ADDRESS
(1)  017232  032737  040000  177570  3$:     BIT     #SW14,SWR          ;LOOP ON TEST ?
(1)  017240  001054                          BNE     6$                 ;BR IF LOOPING
(1)  017242  104400  023213                  TYPE    ,CYCLED            ;TYPE 'CYCLE DOWN'
(1)  017246  104400  022707                  TYPE    ,SWTCHN            ;'SWITCH TO A/B'
(1)  017252  104400  023235                  TYPE    ,CYCLEU            ;'CYCLE THE DRIVE UP'
(2)  017256  113760  001220  000010          MOVB    PORTB,RHCS2(R0)    ;SELECT PORT B
(2)  017264  013737  001220  001226          MOV     PORTB,PTNBR    ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
(1)  017272  032760  010000  000012  4$:     BIT     #MOL,RHDS1(R0)     ;CHECK 'MOL'
(1)  017300  001374                          BNE     4$                 ;BR IF SET (DRIVE NOT CYCLED DOWN)
(1)  017302  032760  010000  000012  5$:     BIT     #MOL,RHDS1(R0)     ;CHECK 'MOL' AGAIN
(1)  017310  001774                          BEQ     5$                 ;BR IF NOT SET (DRIVE NOT CYCLED UP)
(1)
(2)                                ;;*************************************************************************
(1)                                ;SET VOLUME VALID FOR BOTH PORTS
(1)
(1)  017312  012760  000011  000000          MOV     #11,RHCS1(R0)    ;ISSUE A DRIVE CLEAR THROUGH PORT B
(1)  017320  012760  000021  000000          MOV     #21,RHCS1(R0)    ;ISSUE A READIN PRESET THROUGH PORT B
(1)  017326  012760  000013  000000          MOV     #13,RHCS1(R0)    ;RELEASE PORT B
(2)  017334  113760  001216  000010          MOVB    PORTA,RHCS2(R0)   ;SELECT PORT A
(2)  017342  013737  001216  001226          MOV     PORTA,PTNBR    ;MOVE PORT ADDRESS TO LOCATION FOR TYPEOUT
(1)  017350  012760  000021  000000          MOV     #21,RHCS1(R0)    ;ISSUE A READIN PRESET THROUGH PORT A
(1)  017356  012760  010000  000032          MOV     #FMT22,RHOF(R0)  ;SET FMT22
(1)  017364  012760  000013  000000          MOV     #13,RHCS1(R0)    ;RELEASE PORT A
(1)  017372  000137  017610          6$:     JMP     $EOP               ;GO TO THE END OF PASS ROUTINE
5368
5369                               ;;*************************************************************************
5370
5371                               .SBTTL  *** SUBROUTINES ***
5372
5373                               ;;*************************************************************************
5374
5375
5376                               ;ROUTINE TO CHECK FOR KW11-L OR KW11-P CLOCKS
5377
5378  017376  012737  017446  000004  CKCLK:  MOV     #CKCLK1,@#ERRVEC   ;SET UP VECTOR FOR CLOCK CHECK
5379  017404  005037  000006                  CLR     @#ERRVEC+2          ;NEW PSW
5380  017410  005777  161570                  TST     @$LKCSR            ;CHECK FOR KW11-P
```

```
5381  017414  013701  001210                  MOV     $LPVEC,R1       ;KW11-P VECTOR ADDRESS
5382  017420  012721  017530                  MOV     #CLOCK,(R1)+    ;SET UP KW11-P VECTOR
5383  017424  012711  000300                  MOV     #300,(R1)       ;PSW - PRI 6
5384  017430  012777  177777  161550          MOV     #-1,@$LKCSB     ;LOAD COUNTER BUFFER WITH 1'S
5385  017436  012777  000135  161540          MOV     #135$,@$LKCSR   ;SET CLOCK - CNT UP, 16MS, CONT INT
5386  017444  000425                          BR      CKCLK3
5387  017446  062706  000004          CKCLK1: ADD     #4,SP           ;RESTORE THE STACK POINTER
5388  017452  012737  017510  000004          MOV     #CKCLK2,@#ERRVEC  ;CHANGE ERROR VECTOR TO CHECK FOR KW11-L
5389  017460  005777  161526                  TST     @$LKS           ;LOOK FOR KW11-L
5390  017464  013701  001214                  MOV     $LLVEC,R1       ;KW11-L VECTOR ADDRESS
5391  017470  012721  017530                  MOV     #CLOCK,(R1)+    ;SET UP KW11-L VECTOR
5392  017474  012711  000300                  MOV     #300,(R1)       ;PSW - PRI 6
5393  017500  012777  000100  161504          MOV     #100,@$LKS      ;SET KW11-L INTERRUPT
5394  017506  000404                          BR      CKCLK3
5395  017510  062706  000004          CKCLK2: ADD     #4,SP           ;RESTORE THE STACK POINTER
5396  017514  062716  000002                  ADD     #2,(SP)         ;INCREMENT RETURN, NO CLOCK
5397  017520  012737  000006  000004  CKCLK3: MOV     #6,@#ERRVEC     ;RESTORE THE ERROR VECTOR
5398  017526  000207                          RTS     PC
5399
5400                                  ;ROUTINE TO COUNT CLOCK TICKS
5401
5402  017530  062737  000021  001244  CLOCK:  ADD     #17.,TIME       ;ADD 17 MS TO ELAPSED TIME COUNTER
5403  017536  005737  001246                  TST     WATCH           ;IS WATCH ALREADY ZERO ?
5404  017542  001406                          BEQ     1$              ;BR IF IT IS
5405  017544  162737  000021  001246          SUB     #17.,WATCH      ;SUBTRACT 17 MS FROM WATCH DOG COUNTER
5406  017552  100002                          BPL     1$              ;BR IF NOT MINUS
5407  017554  005037  001246                  CLR     WATCH           ;CLEAR WATCH DOG COUNTER
5408  017560  000002                  1$:     RTI                     ;RETURN
5409
5410                                  ;ROUTINE TO CALCULATE + 25% TIME TOLERANCE VALUES
5411
5412  017562  005746                  TOLER:  TST     -(SP)           ;MAKE ROOM ON THE STACK
5413  017564  016616  000002                  MOV     2(SP),(SP)      ;SAVE STACK
5414  017570  013546                          MOV     @(R5)+,-(SP)    ;GET TIME VALUE
5415  017572  011666  000004                  MOV     (SP),4(SP)      ;MOVE TIME VALUE
5416  017576  006216                          ASR     (SP)            ;DIVIDE BY 2
5417  017600  006216                          ASR     (SP)            ;DIVIDE BY 2 AGAIN (FOR A TOTAL OF 4)
5418  017602  062666  000002                  ADD     (SP)+,2(SP)     ;CALCULATE UPPER LIMIT FOR TIMEOUT
5419  017606  000205                          RTS     R5              ;RETURN WITH TOLERANCES ON THE STACK
5420
5421                                  ;;******************************************************************************
5422
5423                                  .SBTTL  'SYSMAC' UTILITY ROUTINES
5424
5425                                  ;;******************************************************************************
5426
5433                                  ;*****************************************************************
  (1)
  (1)                                 .SBTTL  END OF PASS ROUTINE
  (1)
  (1)                                 ;*INCREMENT THE PASS NUMBER ($PASS)
  (1)                                 ;*INDICATE END-OF-PROGRAM AFTER 1 PASSES THRU THE PROGRAM
  (1)                                 ;*TYPE "END PASS #XXXXX" (WHERE XXXXX IS A DECIMAL NUMBER)
  (1)                                 ;*IF THERES A MONITOR GO TO IT
```

```
(1)                                     ;*IF THERE ISN'T JUMP TO TST1AA
(1)
(1)  017610                             $EOP:
(3)  017610  000004                             SCOPE
(3)  017612  005737  001260                     TST     KYBCTL          ;ENTERED TEST VIA KEYBOARD COMMAND ?
(3)  017616  001402                             BEQ     .+6             ;BR IF NOT
(3)  017620  000137  002262                     JMP     EXEC            ;RETURN TO KEYBOARD CONTROL
(1)  017624  005037  001102                     CLR     $TSTNM          ;;ZERO THE TEST NUMBER
(1)  017630  005037  001170                     CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
(1)  017634  005237  001100                     INC     $PASS           ;;INCREMENT THE PASS NUMBER
(1)  017640  042737  100000  001100             BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
(1)  017646  005327                             DEC     (PC)+           ;;LOOP?
(1)  017650  000001             $EOPCT: .WORD   1
(1)  017652  003031                             BGT     $DOAGN          ;;YES
(1)  017654  012737                             MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
(1)  017656  000001             $ENDCT: .WORD   1
(1)  017660  017650                             $EOPCT
(1)  017662  104400  017742                     TYPE    ,$ENDMG         ;;TYPE "END PASS #"
(2)  017666  013746  001100                     MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
(2)  017672  104410                             TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
(1)  017674  104400  017757                     TYPE    ,$ENULL         ;;TYPE A NULL CHARACTER
(1)  017700  013700  000042             $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
(1)  017704  001414                             BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
(1)  017706  022700  017726                     CMP     #$ENDAD,R0      ;;IS MONITOR ACT11?
(1)  017712  001005                             BNE     $ENDAD          ;;NO--BRANCH
(1)  017714  022760  177777  000002             CMP     #-1,2(R0)       ;;YES--IS THIS THE LAST PASS?
(1)  017722  001005                             BNE     $DOAGN          ;;NO--MAKE ANOTHER PASS
(1)  017724  000005                             RESET                   ;;CLEAR THE WORLD
(1)  017726  004710             $ENDAD: JSR     PC,(R0)                 ;;GO TO MONITOR
(1)  017730  000240                             NOP                     ;;SAVE ROOM
(1)  017732  000240                             NOP                     ;;FOR
(1)  017734  000240                             NOP                     ;;ACT11
(1)  017736  000137  002554             $DOAGN: JMP     @#TST1AA                ;;RETURN
(1)  017742  005015  047105  020104     $ENDMG: .ASCIZ  <15><12>/END PASS #/
(1)  017750  040520  051523  021440
(1)  017756     000
(1)  017757     377     377     000     $ENULL: .BYTE   -1,-1,0          ;;NULL CHARACTER STRING
5437                                    ;*****************************************************************
(1)
(1)                                     .SBTTL  SCOPE HANDLER ROUTINE
(1)
(1)                                     ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
(1)                                     ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
(1)                                     ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
(1)                                     ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
(1)                                     ;*SW14=1        LOOP ON TEST
(1)                                     ;*SW11=1        INHIBIT ITERATIONS
(1)                                     ;*CALL
(1)                                     ;*      SCOPE           ;;SCOPE=IOT
(1)
(1)  017762                             $SCOPE:
(1)  017762  006137  177570                     ROL     @#SWR           ;;LOOP ON PRESENT TEST?
(1)  017766  100455                             BMI     $OVER           ;;YES IF SW14=1
(1)                                     ;*****START OF CODE FOR THE XOR TESTER*****
```

```
(1)  017770  000416                    $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
(1)                                                                     ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
(1)  017772  013746  000004                    MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
(1)  017776  012737  020016  000004            MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
(1)  020004  005737  177060                    TST     @#177060        ;;TIME OUT ON XOR?
(1)  020010  012637  000004                    MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
(1)  020014  000436                            BR      $SVLAD          ;;GO TO THE NEXT TEST
(1)  020016  022626                    5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
(1)  020020  012637  000004                    MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
(1)  020024  000436                            BR      $OVER           ;;LOOP ON THE PRESENT TEST
(1)  020026                            6$:;#####END OF CODE FOR THE XOR TESTER#####
(1)  020026  105737  001103            2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
(1)  020032  001404                            BEQ     3$              ;;BR IF NO
(1)  020034  105037  001103            4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
(1)  020040  005037  001170                    CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
(1)  020044  032737  004000  177570    3$:     BIT     #BIT11,@#SWR    ;;INHIBIT ITERATIONS?
(1)  020052  001011                            BNE     1$              ;;BR IF YES
(1)  020054  005737  001100                    TST     $PASS           ;;IF FIRST PASS OF PROGRAM
(1)  020060  001406                            BEQ     1$              ;;      INHIBIT ITERATIONS
(1)  020062  005237  001104                    INC     $ICNT           ;;INCREMENT ITERATION COUNT
(1)  020066  023737  001170  001104            CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
(1)  020074  002012                            BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
(1)  020076  012737  000001  001104    1$:     MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
(1)  020104  013737  020136  001170            MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
(1)  020112  105237  001102            $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
(1)  020116  011637  001106                    MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
(1)  020122  013737  001102  177570    $OVER:  MOV     $TSTNM,@#DISPLAY;;DISPLAY TEST NUMBER
(1)  020130  013716  001106                    MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
(1)  020134  000002                            RTI                     ;;FIXES PS
(1)  020136  000004                    $MXCNT: 4                       ;;MAX. NUMBER OF ITERATIONS
5438                                   ;*****************************************************************
(1)
(1)                                    .SBTTL  ERROR HANDLER ROUTINE
(1)
(1)                                    ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
(1)                                    ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
(1)                                    ;*AND GO TO $ERRTYP ON ERROR
(1)                                    ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
(1)                                    ;*SW15=1        HALT ON ERROR
(1)                                    ;*SW13=1        INHIBIT ERROR TYPEOUTS
(1)                                    ;*SW10=1        BELL ON ERROR
(1)                                    ;*CALL
(1)                                    ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER
(1)
(1)  020140                            $ERROR:
(2)  020140  113737  001102  001234            MOVB    $TSTNM,TSTNUM
(1)  020146  105237  001103            7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
(1)  020152  001775                            BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
(1)  020154  013737  001102  177570            MOV     $TSTNM,@#DISPLAY;;DISPLAY TEST NUMBER AND ERROR FLAG
(1)  020162  032737  002000  177570            BIT     #BIT10,@#SWR    ;;BELL ON ERROR?
(1)  020170  001402                            BEQ     1$              ;;NO - SKIP
(1)  020172  104400  001174                    TYPE    ,$BELL          ;;RING BELL
(1)  020176  005237  001112            1$:     INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
(1)  020202  011637  001116                    MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
```

```
(1)  020206  162737  000002  001116          SUB     #2,$ERRPC
(1)  020214  117737  160676  001114          MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
(1)  020222  032737  020000  177570          BIT     #BIT13,@#SWR    ;;SKIP TYPEOUT IF SET
(1)  020230  001004                          BNE     2$              ;;SKIP TYPEOUTS
(1)  020232  004737  020254                  JSR     PC,@#$ERRTYP    ;;GO TO USER ERROR ROUTINE
(1)  020236  104400  001201                  TYPE    ,$CRLF
(1)  020242  005737  177570          2$:     TST     @#SWR           ;;HALT ON ERROR
(1)  020246  100001                          BPL     3$              ;;SKIP IF CONTINUE
(1)  020250  000000                          HALT                    ;;HALT ON ERROR!
(1)  020252                          3$:
(1)  020252  000002                          RTI                     ;;RETURN
5439                                 ;*****************************************************************
(1)
(1)                                  .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE
(1)
(1)                                  ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
(1)                                  ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
(1)                                  ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
(1)
(1)  020254                          $ERRTYP:
(1)  020254  104400  001201                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
(1)  020260  010046                          MOV     R0,-(SP)        ;;SAVE R0
(1)  020262  005000                          CLR     R0              ;;PICKUP THE ITEM INDEX
(1)  020264  153700  001114                  BISB    @#$ITEMB,R0
(1)  020270  001004                          BNE     1$              ;;IF ITEM NUMBER IS ZERO, JUST
(1)                                                                  ;;TYPE THE PC OF THE ERROR
(2)  020272  013746  001116                  MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
(2)                                                                  ;;ERROR ADDRESS
(2)  020276  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
(1)  020300  000445                          BR      10$             ;;GET OUT
(1)  020302  005300                  1$:     DEC     R0              ;;ADJUST THE INDEX SO THAT IT WILL
(1)  020304  006300                          ASL     R0              ;;      WORK FOR THE ERROR TABLE
(1)  020306  006300                          ASL     R0
(1)  020310  006300                          ASL     R0
(1)  020312  062700  001266                  ADD     #$ERRTB,R0      ;;FORM TABLE POINTER
(1)  020316  012037  020326                  MOV     (R0)+,2$        ;;PICKUP "ERROR MESSAGE" POINTER
(1)  020322  001404                          BEQ     3$              ;;SKIP TYPEOUT IF NO POINTER
(1)  020324  104400                          TYPE                    ;;TYPE THE "ERROR MESSAGE"
(1)  020326  000000                  2$:     .WORD   0               ;;"ERROR MESSAGE" POINTER GOES HERE
(1)  020330  104400  001201                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
(1)  020334  012037  020344          3$:     MOV     (R0)+,4$        ;;PICKUP "DATA HEADER" POINTER
(1)  020340  001404                          BEQ     5$              ;;SKIP TYPEOUT IF 0
(1)  020342  104400                          TYPE                    ;;TYPE THE "DATA HEADER"
(1)  020344  000000                  4$:     .WORD   0               ;;"DATA HEADER" POINTER GOES HERE
(1)  020346  104400  001201                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
(1)  020352  010146                  5$:     MOV     R1,-(SP)        ;;SAVE R1
(1)  020354  012001                          MOV     (R0)+,R1        ;;PICKUP "DATA TABLE" POINTER
(1)  020356  001415                          BEQ     9$              ;;BR IF NO DATA TO BE TYPED
(1)  020360  012000                          MOV     (R0)+,R0        ;;PICKUP "DATA FORMAT" POINTER
(1)  020362  105720                  6$:     TSTB    (R0)+           ;;"OCTAL" OR "DECIMAL"
(1)  020364  001003                          BNE     7$              ;;BR IF DECIMAL
(2)  020366  013146                          MOV     @(R1)+,-(SP)    ;;SAVE @(R1)+ FOR TYPEOUT
(2)  020370  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
(1)  020372  000402                          BR      8$
```

```
(2)  020374                          7$:
(2)  020374  013146                          MOV     @(R1)+,-(SP)    ;;SAVE @(R1)+ FOR TYPEOUT
(2)  020376  104410                          TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
(1)  020400  005711                  8$:     TST     (R1)            ;;IS THERE ANOTHER NUMBER?
(1)  020402  001403                          BEQ     9$              ;;BR IF NO
(1)  020404  104400  020424                  TYPE    ,11$            ;;TYPE TWO(2) SPACES
(1)  020410  000764                          BR      6$              ;;LOOP
(1)
(1)  020412  012601                  9$:     MOV     (SP)+,R1        ;;RESTORE R1
(1)  020414  012600                  10$:    MOV     (SP)+,R0        ;;RESTORE R0
(1)  020416  104400  001201                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
(1)  020422  000207                          RTS     PC              ;;RETURN
(1)  020424  020040      000         11$:    .ASCIZ  /  /            ;;TWO(2) SPACES
(1)          020430                          .EVEN
5440                                 ;*****************************************************************
(1)
(1)                                  .SBTTL  TYPE ROUTINE
(1)
(1)                                  ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
(1)                                  ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
(1)                                  ;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
(1)                                  ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
(1)                                  ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
(1)                                  ;*
(1)                                  ;*CALL:
(1)                                  ;*1) USING A TRAP INSTRUCTION
(1)                                  ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
(1)                                  ;*OR
(1)                                  ;*      TYPE
(1)                                  ;*      MESADR
(1)                                  ;*
(1)                                  ;*2) USING A JSR INSTRUCTION
(1)                                  ;*      MOV     PS,-(SP)        ;;PUSH PROCESSOR STATUS WORD ON THE STACK
(1)                                  ;*      JSR     PC,$TYPE        ;;CALL TYPE ROUTINE
(1)                                  ;*      MESADDR                 ;;FIRST ADRESS OF MESSAGE
(1)
(1)  020430  105737  001151          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
(1)  020434  100002                          BPL     1$              ;;BR IF YES
(1)  020436  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
(1)  020440  000407                          BR      3$              ;;LEAVE
(1)  020442  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
(1)  020444  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
(1)  020450  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
(1)  020452  001005                          BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
(1)  020454  005726                          TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
(1)  020456  012600                          MOV     (SP)+,R0        ;;RESTORE R0
(1)  020460  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
(1)  020464  000002                          RTI                     ;;RETURN
(1)  020466  004737  020520          4$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
(1)  020472  123726  001150          5$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
(1)  020476  001364                          BNE     2$              ;;IF NO GO GET NEXT CHAR.
(1)  020500  013746  001146                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
(1)                                                                  ;;AND THE NULL CHAR.
(1)  020504  105366  000001          6$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
```

```
(1)  020510  002770                        BLT     5$                ;;BR IF NO--GO POP THE NULL OFF OF STACK
(1)  020512  004737  020520                JSR     PC,$TYPEC         ;;GO TYPE A NULL
(1)  020516  000772                        BR      6$                ;;LOOP
(1)  020520  105777  160416          $TYPEC: TSTB    @$TPS             ;;WAIT UNTIL PRINTER IS READY
(1)  020524  100375                        BPL     $TYPEC
(1)  020526  116677  000002  160410        MOVB    2(SP),@$TPB       ;;LOAD CHAR TO BE TYPED INTO DATA REG.
(1)  020534  000207                        RTS     PC
5441                                ;*****************************************************************
(1)
(1)                                 .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
(1)
(1)                                 ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
(1)                                 ;*OCTAL (ASCII) NUMBER AND TYPE IT.
(1)                                 ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
(1)                                 ;*CALL:
(1)                                 ;*      MOV     NUM,-(SP)         ;;NUMBER TO BE TYPED
(1)                                 ;*      TYPOS                     ;;CALL FOR TYPEOUT
(1)                                 ;*      .BYTE   N                 ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
(1)                                 ;*      .BYTE   M                 ;;M=1 OR 0
(1)                                 ;*                                ;;1=TYPE LEADING ZEROS
(1)                                 ;*                                ;;0=SUPPRESS LEADING ZEROS
(1)                                 ;*
(1)                                 ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
(1)                                 ;*$TYPOS OR $TYPOC
(1)                                 ;*CALL:
(1)                                 ;*      MOV     NUM,-(SP)         ;;NUMBER TO BE TYPED
(1)                                 ;*      TYPON                     ;;CALL FOR TYPEOUT
(1)                                 ;*
(1)                                 ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
(1)                                 ;*CALL:
(1)                                 ;*      MOV     NUM,-(SP)         ;;NUMBER TO BE TYPED
(1)                                 ;*      TYPOC                     ;;CALL FOR TYPEOUT
(1)
(1)  020536  017646  000000        $TYPOS: MOV     @(SP),-(SP)       ;;PICKUP THE MODE
(1)  020542  116637  000001  020761        MOVB    1(SP),$OFILL      ;;LOAD ZERO FILL SWITCH
(1)  020550  112637  020763                MOVB    (SP)+,$OMODE+1    ;;NUMBER OF DIGITS TO TYPE
(1)  020554  062716  000002                ADD     #2,(SP)           ;;ADJUST RETURN ADDRESS
(1)  020560  000406                        BR      $TYPON
(1)  020562  112737  000001  020761  $TYPOC: MOVB    #1,$OFILL         ;;SET THE ZERO FILL SWITCH
(1)  020570  112737  000006  020763        MOVB    #6,$OMODE+1       ;;SET FOR SIX(6) DIGITS
(1)  020576  112737  000005  020760  $TYPON: MOVB    #5,$OCNT          ;;SET THE ITERATION COUNT
(1)  020604  010346                        MOV     R3,-(SP)          ;;SAVE R3
(1)  020606  010446                        MOV     R4,-(SP)          ;;SAVE R4
(1)  020610  010546                        MOV     R5,-(SP)          ;;SAVE R5
(1)  020612  113704  020763                MOVB    $OMODE+1,R4       ;;GET THE NUMBER OF DIGITS TO TYPE
(1)  020616  005404                        NEG     R4
(1)  020620  062704  000006                ADD     #6,R4             ;;SUBTRACT IT FOR MAX. ALLOWED
(1)  020624  110437  020762                MOVB    R4,$OMODE         ;;SAVE IT FOR USE
(1)  020630  113704  020761                MOVB    $OFILL,R4         ;;GET THE ZERO FILL SWITCH
(1)  020634  016605  000012                MOV     12(SP),R5         ;;PICKUP THE INPUT NUMBER
(1)  020640  005003                        CLR     R3                ;;CLEAR THE OUTPUT WORD
(1)  020642  006105                  1$:   ROL     R5                ;;ROTATE MSB INTO "C"
(1)  020644  000404                        BR      3$                ;;GO DO MSB
(1)  020646  006105                  2$:   ROL     R5                ;;FORM THIS DIGIT
```

```
(1)  020650  006105                        ROL     R5
(1)  020652  006105                        ROL     R5
(1)  020654  010503                        MOV     R5,R3
(1)  020656  006103                3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
(1)  020660  105337  020762                DECB    $OMODE          ;;TYPE THIS DIGIT?
(1)  020664  100016                        BPL     7$              ;;BR IF NO
(1)  020666  042703  177770                BIC     #177770,R3      ;;GET RID OF JUNK
(1)  020672  001002                        BNE     4$              ;;TEST FOR 0
(1)  020674  005704                        TST     R4              ;;SUPPRESS THIS 0?
(1)  020676  001403                        BEQ     5$              ;;BR IF YES
(1)  020700  005204                4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
(1)  020702  052703  000060                BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
(1)  020706  052703  000040                BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
(1)  020712  110337  020756                MOVB    R3,8$           ;;SAVE FOR TYPING
(1)  020716  104400  020756                TYPE    ,8$             ;;GO TYPE THIS DIGIT
(1)  020722  105337  020760                DECB    $OCNT           ;;COUNT BY 1
(1)  020726  003347                        BGT     2$              ;;BR IF MORE TO DO
(1)  020730  002402                        BLT     6$              ;;BR IF DONE
(1)  020732  005204                        INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
(1)  020734  000744                        BR      2$              ;;GO DO THE LAST DIGIT
(1)  020736  012605                6$:     MOV     (SP)+,R5        ;;RESTORE R5
(1)  020740  012604                        MOV     (SP)+,R4        ;;RESTORE R4
(1)  020742  012603                        MOV     (SP)+,R3        ;;RESTORE R3
(1)  020744  016666  000002  000004        MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
(1)  020752  012616                        MOV     (SP)+,(SP)
(1)  020754  000002                        RTI                     ;;RETURN
(1)  020756     000                8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
(1)  020757     000                        .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
(1)  020760     000                $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
(1)  020761     000                $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
(1)  020762  000000                $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
5442                               ;*****************************************************************
(1)
(1)                                .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
(1)
(1)                                ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
(1)                                ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
(1)                                ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
(1)                                ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
(1)                                ;*REPLACED WITH SPACES.
(1)                                ;*CALL:
(1)                                ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
(1)                                ;*      TYPDS                   ;;GO TO THE ROUTINE
(1)
(2)  020764                        $TYPDS:
(3)  020764  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
(3)  020766  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
(3)  020770  010246                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
(3)  020772  010346                        MOV     R3,-(SP)        ;;PUSH R3 ON STACK
(3)  020774  010546                        MOV     R5,-(SP)        ;;PUSH R5 ON STACK
(1)  020776  012746  020200                MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
(1)  021002  016605  000020                MOV     20(SP),R5       ;;GET THE INPUT NUMBER
(1)  021006  100004                        BPL     1$              ;;BR IF INPUT IS POS.
(1)  021010  005405                        NEG     R5              ;;MAKE THE BINARY NUMBER POS.
```

```
(1)  021012  112766  000055  000001          MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
(1)  021020  005000                    1$:    CLR     R0              ;;ZERO THE CONSTANTS INDEX
(1)  021022  012703  021200                    MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
(1)  021026  112723  000040                    MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
(1)  021032  005002                    2$:    CLR     R2              ;;CLEAR THE BCD NUMBER
(1)  021034  016001  021170                    MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
(1)  021040  160105                    3$:    SUB     R1,R5           ;;FORM THIS BCD DIGIT
(1)  021042  002402                            BLT     4$              ;;BR IF DONE
(1)  021044  005202                            INC     R2              ;;INCREASE THE BCD DIGIT BY 1
(1)  021046  000774                            BR      3$
(1)  021050  060105                    4$:    ADD     R1,R5           ;;ADD BACK THE CONSTANT
(1)  021052  005702                            TST     R2              ;;CHECK IF BCD DIGIT=0
(1)  021054  001002                            BNE     5$              ;;FALL THROUGH IF 0
(1)  021056  105716                            TSTB    (SP)            ;;STILL DOING LEADING 0'S?
(1)  021060  100407                            BMI     7$              ;;BR IF YES
(1)  021062  106316                    5$:    ASLB    (SP)            ;;MSD?
(1)  021064  103003                            BCC     6$              ;;BR IF NO
(1)  021066  116663  000001  177777            MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
(1)  021074  052702  000060                6$: BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
(1)  021100  052702  000040                7$: BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
(1)  021104  110223                            MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
(1)  021106  005720                            TST     (R0)+           ;;JUST INCREMENTING
(1)  021110  020027  000010                    CMP     R0,#10          ;;CHECK THE TABLE INDEX
(1)  021114  002746                            BLT     2$              ;;GO DO THE NEXT DIGIT
(1)  021116  003002                            BGT     8$              ;;GO TO EXIT
(1)  021120  010502                            MOV     R5,R2           ;;GET THE LSD
(1)  021122  000764                            BR      6$              ;;GO CHANGE TO ASCII
(1)  021124  105726                    8$:    TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
(1)  021126  100003                            BPL     9$              ;;BR IF NO
(1)  021130  116663  177777  177776            MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
(1)  021136  105013                    9$:    CLRB    (R3)            ;;SET THE TERMINATOR
(3)  021140  012605                            MOV     (SP)+,R5        ;;POP STACK INTO R5
(3)  021142  012603                            MOV     (SP)+,R3        ;;POP STACK INTO R3
(3)  021144  012602                            MOV     (SP)+,R2        ;;POP STACK INTO R2
(3)  021146  012601                            MOV     (SP)+,R1        ;;POP STACK INTO R1
(3)  021150  012600                            MOV     (SP)+,R0        ;;POP STACK INTO R0
(1)  021152  104400  021200                    TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
(1)  021156  016666  000002  000004            MOV     2(SP),4(SP)     ;;ADJUST THE STACK
(1)  021164  012616                            MOV     (SP)+,(SP)
(1)  021166  000002                            RTI                     ;;RETURN TO USER
(1)  021170  023420                    $DTBL:  10000.
(1)  021172  001750                            1000.
(1)  021174  000144                            100.
(1)  021176  000012                            10.
(1)  021200  000004                    $DBLK:  .BLKW   4
5443                                    ;*****************************************************************
 (1)
 (1)                                    .SBTTL  TTY INPUT ROUTINE
 (1)
 (1)                                    ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
 (1)                                    ;*CALL:
 (1)                                    ;*      RDCHR                   ;;INPUT A SINGLE CHARACTER FROM THE TTY
 (1)                                    ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
 (1)                                    ;
```

```
(1)
(1)  021210  011646                    $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
(1)  021212  016666  000004  000002            MOV     4(SP),2(SP)     ;;SAVE THE PS
(1)  021220  105777  157712            1$:     TSTB    @$TKS           ;;WAIT FOR
(1)  021224  100375                            BPL     1$              ;;A CHARACTER
(1)  021226  117766  157706  000004            MOVB    @$TKB,4(SP)     ;;READ THE TTY
(1)  021234  042766  177600  000004            BIC     #^C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
(1)  021242  000002                            RTI                     ;;GO BACK TO USER
(2)                                     ;*****************************************************************
(1)                                     ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
(1)                                     ;*CALL:
(1)                                     ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
(1)                                     ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
(1)                                     ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
(1)
(1)  021244  010346                    $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
(1)  021246  005046                            CLR     -(SP)           ;;CLEAR THE RUBOUT KEY
(1)  021250  012703  021505            1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
(1)  021254  022703  021514            2$:     CMP     #$TTYIN+7,R3    ;;BUFFER FULL?
(1)  021260  101456                            BLOS    4$              ;;BR IF YES
(1)  021262  104412                            RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
(1)  021264  112613                            MOVB    (SP)+,(R3)      ;;GET CHARACTER
(1)  021266  122713  000177                    CMPB    #177,(R3)       ;;IS IT A RUBOUT
(1)  021272  001022                            BNE     5$              ;;BR IF NO
(1)  021274  005716                            TST     (SP)            ;;IS THIS THE FIRST RUBOUT?
(1)  021276  001007                            BNE     6$              ;;BR IF NO
(1)  021300  112737  000134  021476            MOVB    #'\,9$          ;;TYPE A BACK SLASH
(1)  021306  104400  021476                    TYPE    ,9$
(1)  021312  012716  177777                    MOV     #-1,(SP)        ;;SET THE RUBOUT KEY
(1)  021316  005303                    6$:     DEC     R3              ;;BACKUP BY ONE
(1)  021320  020327  021505                    CMP     R3,#$TTYIN      ;;STACK EMPTY?
(1)  021324  103434                            BLO     4$              ;;BR IF YES
(1)  021326  111337  021476                    MOVB    (R3),9$         ;;SETUP TO TYPEOUT THE DELETED CHAR.
(1)  021332  104400  021476                    TYPE    ,9$             ;;GO TYPE
(1)  021336  000746                            BR      2$              ;;GO READ ANOTHER CHAR.
(1)  021340  005716                    5$:     TST     (SP)            ;;RUBOUT KEY SET?
(1)  021342  001406                            BEQ     7$              ;;BR IF NO
(1)  021344  112737  000134  021476            MOVB    #'\,9$          ;;TYPE A BACK SLASH
(1)  021352  104400  021476                    TYPE    ,9$
(1)  021356  005016                            CLR     (SP)            ;;CLEAR THE RUBOUT KEY
(1)  021360  122713  000025            7$:     CMPB    #25,(R3)        ;;IS CHARACTER A CTRL U?
(1)  021364  001003                            BNE     8$              ;;BR IF NO
(1)  021366  104400  021500                    TYPE    ,$CNTLU         ;;TYPE A CONTROL "U"
(1)  021372  000726                            BR      1$              ;;GO START OVER
(1)  021374  122713  000012            8$:     CMPB    #12,(R3)        ;;IS CHARACTER A "LF"?
(1)  021400  001011                            BNE     3$              ;;BRANCH IF NO
(1)  021402  105013                            CLRB    (R3)            ;;CLEAR THE CHARACTER
(1)  021404  104400  001201                    TYPE    ,$CRLF          ;;TYPE A "CR" & "LF"
(1)  021410  104400  021505                    TYPE    ,$TTYIN         ;;TYPE THE INPUT STRING
(1)  021414  000717                            BR      2$              ;;GO PICKUP ANOTHER CHACTER
(1)  021416  104400  001200            4$:     TYPE    ,$QUES          ;;TYPE A '?'
(1)  021422  000712                            BR      1$              ;;CLEAR THE BUFFER AND LOOP
(1)  021424  111337  021476            3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
(1)  021430  104400  021476                    TYPE    ,9$
```

```
(1)  021434  122723  000015                    CMPB    #15,(R3)+       ;;CHECK FOR RETURN
(1)  021440  001305                            BNE     2$              ;;LOOP IF NOT RETURN
(1)  021442  105063  177777                    CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
(1)  021446  104400  001202                    TYPE    $LF             ;;TYPE A LINE FEED
(1)  021452  005726                            TST     (SP)+           ;;CLEAN RUBOUT KEY FROM THE STACK
(1)  021454  012603                            MOV     (SP)+,R3        ;;RESTORE R3
(1)  021456  011646                            MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
(1)  021460  016666  000004  000002            MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
(1)  021466  012766  021505  000004            MOV     #$TTYIN,4(SP)
(1)  021474  000002                            RTI                     ;;RETURN
(1)  021476     000                 9$:        .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
(1)  021477     000                            .BYTE   0               ;;TERMINATOR
(1)  021500  052536  005015     000 $CNTLU:    .ASCIZ  /↑U/<15><12>    ;;CONTROL "U"
(1)  021505  000007                 $TTYIN:    .BLKB   7               ;;RESERVE 7 BYTES FOR TTY INPUT
5444                                ;*****************************************************************
(1)
(1)                                 .SBTTL  READ AN OCTAL NUMBER FROM THE TTY
(1)
(1)                                 ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
(1)                                 ;*CHANGE IT TO BINARY.
(1)                                 ;*THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL
(1)                                 ;*OCTAL DIGITS. IF AN ILLEGAL CHARACTER IS READ A "?" WILL BE TYPED
(1)                                 ;*FOLLOWED BY A CARRIAGE RETURN-LINE FEED. THE COMPLETE NUMBER MUST
(1)                                 ;*THEN BE RETYPED. THE INPUT IS TERMINATED BY TYPING A CARRIAGE RETURN.
(1)                                 ;*CALL:
(1)                                 ;*      RDOCT                   ;;READ AN OCTAL NUMBER
(1)                                 ;*      RETURN HERE             ;;LOW ORDER BITS ARE ON TOP OF THE STACK
(1)                                 ;*                              ;;HIGH ORDER BITS ARE IN $HIOCT
(1)
(1)  021514  011646                 $RDOCT: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR THE
(1)  021516  016666  000004  000002        MOV     4(SP),2(SP)     ;;INPUT NUMBER
(3)  021524  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
(3)  021526  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
(3)  021530  010246                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
(1)  021532  104414                 1$:    RDLIN                   ;;READ AN ASCIZ LINE
(1)  021534  012600                        MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
(1)  021536  010037  021642                MOV     R0,5$           ;;AND SAVE IT
(1)  021542  005001                        CLR     R1              ;;CLEAR DATA WORD
(1)  021544  005002                        CLR     R2
(1)  021546  112046                 2$:    MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
(1)  021550  001420                        BEQ     3$              ;;IF ZERO GET OUT
(1)  021552  122716  000060                CMPB    #'0,(SP)        ;;MAKE SURE THIS CHARACTER
(1)  021556  003026                        BGT     4$              ;;IS AN OCTAL DIGIT
(1)  021560  122716  000067                CMPB    #'7,(SP)
(1)  021564  002423                        BLT     4$
(1)  021566  006301                        ASL     R1              ;;*2
(1)  021570  006102                        ROL     R2
(1)  021572  006301                        ASL     R1              ;;*4
(1)  021574  006102                        ROL     R2
(1)  021576  006301                        ASL     R1              ;;*8
(1)  021600  006102                        ROL     R2
(1)  021602  042716  177770                BIC     #↑C7,(SP)       ;;STRIP THE ASCII JUNK
(1)  021606  062601                        ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
(1)  021610  000756                        BR      2$              ;;LOOP
```

```
(1)  021612  005726                3$:     TST     (SP)+           ;;CLEAN TERMINATOR FROM STACK
(1)  021614  010166  000012              MOV     R1,12(SP)       ;;SAVE THE RESULT
(1)  021620  010237  021652              MOV     R2,$HIOCT
(3)  021624  012602                      MOV     (SP)+,R2        ;;POP STACK INTO R2
(3)  021626  012601                      MOV     (SP)+,R1        ;;POP STACK INTO R1
(3)  021630  012600                      MOV     (SP)+,R0        ;;POP STACK INTO R0
(1)  021632  000002                      RTI                     ;;RETURN
(1)  021634  005726                4$:     TST     (SP)+           ;;CLEAN PARTIAL FROM STACK
(1)  021636  105010                      CLRB    (R0)            ;;SET A TERMINATOR
(1)  021640  104400                      TYPE                    ;;TYPE UP THRU THE BAD CHAR.
(1)  021642  000000                5$:     .WORD   0
(1)  021644  104400  001200              TYPE    ,$QUES          ;;"?" "CR" & "LF"
(1)  021650  000730                      BR      1$              ;;TRY AGAIN
(1)  021652  000000                $HIOCT: .WORD   0               ;;HIGH ORDER BITS GO HERE
5445                               ;*******************************************************************
(1)
(1)                                .SBTTL  SAVE AND RESTORE R0-R5 ROUTINES
(1)
(1)                                ;*SAVE R0-R5
(1)                                ;*CALL:
(1)                                ;*      SAVREG
(1)                                ;*UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
(1)                                ;*
(1)                                ;*TOP---(+16)
(1)                                ;* +2---(+18)
(1)                                ;* +4---R5
(1)                                ;* +6---R4
(1)                                ;* +8---R3
(1)                                ;*+10---R2
(1)                                ;*+12---R1
(1)                                ;*+14---R0
(1)
(1)  021654                        $SAVREG:
(3)  021654  010046                      MOV     R0,-(SP)        ;;PUSH R0 ON STACK
(3)  021656  010146                      MOV     R1,-(SP)        ;;PUSH R1 ON STACK
(3)  021660  010246                      MOV     R2,-(SP)        ;;PUSH R2 ON STACK
(3)  021662  010346                      MOV     R3,-(SP)        ;;PUSH R3 ON STACK
(3)  021664  010446                      MOV     R4,-(SP)        ;;PUSH R4 ON STACK
(3)  021666  010546                      MOV     R5,-(SP)        ;;PUSH R5 ON STACK
(1)  021670  016646  000022              MOV     22(SP),-(SP)    ;;SAVE PS OF MAIN FLOW
(1)  021674  016646  000022              MOV     22(SP),-(SP)    ;;SAVE PC OF MAIN FLOW
(1)  021700  016646  000022              MOV     22(SP),-(SP)    ;;SAVE PS OF CALL
(1)  021704  016646  000022              MOV     22(SP),-(SP)    ;;SAVE PC OF CALL
(1)  021710  000002                      RTI
(1)
(1)                                ;*RESTORE R0-R5
(1)                                ;*CALL:
(1)                                ;*      RESREG
(1)  021712                        $RESREG:
(1)  021712  012666  000022              MOV     (SP)+,22(SP)    ;;RESTORE PC OF CALL
(1)  021716  012666  000022              MOV     (SP)+,22(SP)    ;;RESTORE PS OF CALL
(1)  021722  012666  000022              MOV     (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
(1)  021726  012666  000022              MOV     (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
(3)  021732  012605                      MOV     (SP)+,R5        ;;POP STACK INTO R5
```

```
(3)  021734  012604                        MOV     (SP)+,R4        ;;POP STACK INTO R4
(3)  021736  012603                        MOV     (SP)+,R3        ;;POP STACK INTO R3
(3)  021740  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
(3)  021742  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
(3)  021744  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
(1)  021746  000002                        RTI
5446                            ;*****************************************************************
(1)
(1)                             .SBTTL  TRAP DECODER
(1)
(1)                             ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
(1)                             ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
(1)                             ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
(1)                             ;*GO TO THAT ROUTINE.
(1)
(1)  021750  010046             $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
(1)  021752  016600  000002             MOV     2(SP),R0        ;;GET TRAP ADDRESS
(1)  021756  005740                     TST     -(R0)           ;;BACKUP BY 2
(1)  021760  111000                     MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
(1)  021762  016000  021770             MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
(1)  021766  000200                     RTS     R0              ;;GO TO ROUTINE
(1)
(3)
(3)                             .SBTTL  TRAP TABLE
(3)
(3)                             ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
(3)                             ;*BY THE "TRAP" INSTRUCTION.
(3)
(3)                             ;       ROUTINE
(3)                             ;       -------
(3)  021770                     $TRPAD:
(3)  021770  020430                     $TYPE   ;;CALL=TYPE     TRAP+0(104400)  TTY TYPEOUT ROUTINE
(3)  021772  020562                     $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
(3)  021774  020536                     $TYPOS  ;;CALL=TYPOS    TRAP+4(104404)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
(3)  021776  020576                     $TYPON  ;;CALL=TYPON    TRAP+6(104406)  TYPE OCTAL NUMBER (AS PER LAST CALL)
(3)  022000  020764                     $TYPDS  ;;CALL=TYPDS    TRAP+10(104410) TYPE DECIMAL NUMBER (WITH SIGN)
(3)  022002  021210                     $RDCHR  ;;CALL=RDCHR    TRAP+12(104412) TTY TYPEIN CHARACTER ROUTINE
(3)  022004  021244                     $RDLIN  ;;CALL=RDLIN    TRAP+14(104414) TTY TYPEIN STRING ROUTINE
(3)  022006  021514                     $RDOCT  ;;CALL=RDOCT    TRAP+16(104416) READ AN OCTAL NUMBER FROM TTY
(3)  022010  021654                     $SAVREG ;;CALL=SAVREG   TRAP+20(104420) SAVE R0-R5 ROUTINE
(3)  022012  021712                     $RESREG ;;CALL=RESREG   TRAP+22(104422) RESTORE R0-R5 ROUTINE
5447
5448                            ;;*****************************************************************
5449
5450                            .SBTTL  TELETYPE MESSAGES
5451
5452                            ;;*****************************************************************
5453
5454  022014  005015  040515  047111  TITLE:  .ASCII  <15><12>/MAINDEC-11-DZRPQ-A/<15><12><12>
      022022  042504  026503  030461
      022030  042055  051132  050520
      022036  040455  005015     012
5455  022043     122  030120  020064          .ASCIZ  /RP04 DUAL CONTROLLER LOGIC TEST - PART 2/<15><12><12>
      022050  052504  046101  041440
```

```
                022056  047117  051124  046117
                022064  042514  020122  047514
                022072  044507  020103  042524
                022100  052123  026440  050040
                022106  051101  020124  006462
                022114  005012      000
5456            022117     015  042412  052116  ENTERA: .ASCIZ  <15><12>/ENTER DRIVE ADDRESS: /
                022124  051105  042040  044522
                022132  042526  040440  042104
                022140  042522  051523  020072
                022146     000
5457            022147     111  053116  046101  ADRERR: .ASCIZ  /INVALID ADDRESS/<15><12>
                022154  042111  040440  042104
                022162  042522  051523  005015
                022170     000
5458            022171     015  050012  051117  PORTAIS: .ASCIZ <15><12>/PORT A ADDRESS IS: /
                022176  020124  020101  042101
                022204  051104  051505  020123
                022212  051511  020072      000
5459            022217     015  050012  051117  PORTBIS: .ASCIZ <15><12>/PORT B ADDRESS IS: /
                022224  020124  020102  042101
                022232  051104  051505  020123
                022240  051511  020072      000
5460            022245     015  051412  051531  NOCLOCK: .ASCIZ <15><12>/SYSTEM MUST HAVE 'L' OR 'P' CLOCK/<15><12><12>
                022252  042524  020115  052515
                022260  052123  044040  053101
                022266  020105  046047  020047
                022274  051117  023440  023520
                022302  041440  047514  045503
                022310  005015  000012
5461            022314  042412  052116  051105  TESTNO: .ASCIZ  <12>/ENTER TEST #: /
                022322  052040  051505  020124
                022330  035043  000040
5462            022334  047111  040526  044514  BADNO:  .ASCIZ  /INVALID TEST NUMBER/<15><12>
                022342  020104  042524  052123
                022350  047040  046525  042502
                022356  006522  000012
5463            022362  005015  052012  042510  ADDRIS: .ASCIZ  <15><12><12>/THE PRESENT ADDRESS OF THE RH11 (RHCS1) IS: /
                022370  050040  042522  042523
                022376  052116  040440  042104
                022404  042522  051523  047440
                022412  020106  044124  020105
                022420  044122  030461  024040
                022426  044122  051503  024461
                022434  044440  035123  000040
5464            022442  042412  052116  051105  NTRH11: .ASCIZ  <12>/ENTER NEW RH11 ADDRESS: /
                022450  047040  053505  051040
                022456  030510  020061  042101
                022464  051104  051505  035123
                022472  000040
5465            022474  005015  044124  020105  NORESP: .ASCIZ  <15><12>/THE RH11 DID NOT RESPOND WHEN RHWC ACCESSED AT ADDR: /
                022502  044122  030461  042040
                022510  042111  047040  052117
                022516  051040  051505  047520
```

```
                022524  042116  053440  042510
                022532  020116  044122  041527
                022540  040440  041503  051505
                022546  042523  020104  052101
                022554  040440  042104  035122
                022562  000040
 5466           022564  005015  050012  042522  STANDBY: .ASCII <15><12><12>/PRESS 'STANDBY' ON DRIVE/
                022572  051523  023440  052123
                022600  047101  041104  023531
                022606  047440  020116  051104
                022614  053111     105
 5467           022617     015  050012  047522          .ASCIZ  <15><12>/PROGRAM WILL LOOP WAITING FOR 'MOL' TO SET FROM PORT /
                022624  051107  046501  053440
                022632  046111  020114  047514
                022640  050117  053440  044501
                022646  044524  043516  043040
                022654  051117  023440  047515
                022662  023514  052040  020117
                022670  042523  020124  051106
                022676  046517  050040  051117
                022704  020124     000
 5468           022707     015  005012  042522  SWTCHN: .ASCIZ  <15><12><12>@RETURN 'CONTROLLER SELECT' SWITCH ON DRIVE TO 'A/B'@
                022714  052524  047122  023440
                022722  047503  052116  047522
                022730  046114  051105  051440
                022736  046105  041505  023524
                022744  051440  044527  041524
                022752  020110  047117  042040
                022760  044522  042526  052040
                022766  020117  040447  041057
                022774  000047
 5469           022776  005015  052012  051125  SWTCHA: .ASCIZ  <15><12><12>/TURN 'CONTROLLER SELECT' SWITCH ON DRIVE TO 'A'/
                023004  020116  041447  047117
                023012  051124  046117  042514
                023020  020122  042523  042514
                023026  052103  020047  053523
                023034  052111  044103  047440
                023042  020116  051104  053111
                023050  020105  047524  023440
                023056  023501     000
 5470           023061     015  005012  052524  SWTCHB: .ASCIZ  <15><12><12>/TURN 'CONTROLLER SELECT' SWITCH ON DRIVE TO 'B'/
                023066  047122  023440  047503
                023074  052116  047522  046114
                023102  051105  051440  046105
                023110  041505  023524  051440
                023116  044527  041524  020110
                023124  047117  042040  044522
                023132  042526  052040  020117
                023140  041047  000047
 5471           023144  005015  044124  047105  CONTUE: .ASCIZ  <15><12>/THEN PRESS 'CONT' ON THE PROCESSOR/<15><12>
                023152  050040  042522  051523
                023160  023440  047503  052116
                023166  020047  047117  052040
                023174  042510  050040  047522
```

```
        023202  042503  051523  051117
        023210  005015     000
 5472   023213     015  005012  052123  CYCLED: .ASCIZ  <15><12><12>/STOP THE DRIVE/
        023220  050117  052040  042510
        023226  042040  044522  042526
        023234     000
 5473   023235     015  005012  052123  CYCLEU: .ASCIZ  <15><12><12>/START THE DRIVE - THE PROGRAM WILL WAIT FOR 'MOL' TO SET/
        023242  051101  020124  044124
        023250  020105  051104  053111
        023256  020105  020055  044124
        023264  020105  051120  043517
        023272  040522  020115  044527
        023300  046114  053440  044501
        023306  020124  047506  020122
        023314  046447  046117  020047
        023322  047524  051440  052105
        023330     000
 5474
 5475                           ;;*****************************************************************************
 5476
 5477                           .SBTTL  TEST ERROR MESSAGES
 5478
 5479                           ;;*****************************************************************************
 5480
 5481   023331     104  044522  042526  EM1:    .ASCIZ  /DRIVE IS NON-EXISTANT ('NED' BIT SET)/
        023336  044440  020123  047516
        023344  026516  054105  051511
        023352  040524  052116  024040
        023360  047047  042105  020047
        023366  044502  020124  042523
        023374  024524     000
 5482
 5483   023377     127  047522  043516  EM2:    .ASCIZ  /WRONG DRIVE TYPE/
        023404  042040  044522  042526
        023412  052040  050131  000105
 5484
 5485   023420  047503  052116  047522  EM3:    .ASCIZ  @CONTROLLER SELECT SWITCH ON DRIVE NOT IN 'A/B'@
        023426  046114  051105  051440
        023434  046105  041505  020124
        023442  053523  052111  044103
        023450  047440  020116  051104
        023456  053111  020105  047516
        023464  020124  047111  023440
        023472  027501  023502     000
 5486
 5487   023477     104  044522  042526  EM4:    .ASCIZ  /DRIVE NOT ON LINE/
        023504  047040  052117  047440
        023512  020116  044514  042516
        023520     000
 5488
 5489   023521     123  051105  040511  EM5:    .ASCIZ  /SERIAL NUMBER READ THROUGH EACH PORT NOT THE SAME/
        023526  020114  052516  041115
        023534  051105  051040  040505
        023542  020104  044124  047522
```

```
                023550  043525  020110  040505
                023556  044103  050040  051117
                023564  020124  047516  020124
                023572  044124  020105  040523
                023600  042515     000
5490
5491    023603     124  046511  047505  EM6:    .ASCIZ  /TIMEOUT HAS NOT OCCURED WITHIN 2 SECONDS/
                023610  052125  044040  051501
                023616  047040  052117  047440
                023624  041503  051125  042105
                023632  053440  052111  044510
                023640  020116  020062  042523
                023646  047503  042116  000123
5492
5493    023654  044524  042515  052517  EM7:    .ASCIZ  /TIMEOUT ONE-SHOT IS LESS THAN 500 MS/
                023662  020124  047117  026505
                023670  044123  052117  044440
                023676  020123  042514  051523
                023704  052040  040510  020116
                023712  030065  020060  051515
                023720     000
5494
5495    023721     122  040505  044504  EM10:   .ASCIZ  /READIN PRESET DOES NOT SET VOLUME VALID FOR THE PORT/
                023726  020116  051120  051505
                023734  052105  042040  042517
                023742  020123  047516  020124
                023750  042523  020124  047526
                023756  052514  042515  053040
                023764  046101  042111  043040
                023772  051117  052040  042510
                024000  050040  051117  000124
5496
5497    024006  043447  023517  041040  EM11:   .ASCIZ  /'GO' BIT RESET DURING UNLOAD COMMAND/
                024014  052111  051040  051505
                024022  052105  042040  051125
                024030  047111  020107  047125
                024036  047514  042101  041440
                024044  046517  040515  042116
                024052     000
5498
5499    024053     111  041516  051117  EM12:   .ASCIZ  /INCORRECT STATUS DURING UNLOAD COMMAND/
                024060  042522  052103  051440
                024066  040524  052524  020123
                024074  052504  044522  043516
                024102  052440  046116  040517
                024110  020104  047503  046515
                024116  047101  000104
5500
5501    024122  051104  053111  020105  EM13:   .ASCIZ  /DRIVE DID NOT RETURN TO NEUTRAL AFTER UNLOAD COMMAND/
                024130  044504  020104  047516
                024136  020124  042522  052524
                024144  047122  052040  020117
                024152  042516  052125  040522
                024160  020114  043101  042524
```

```
                024166  020122  047125  047514
                024174  042101  041440  046517
                024202  040515  042116     000
5502
5503    024207     101  052124  047105  EM14:   .ASCIZ  /ATTENTION BIT SET ON 'OPPOSITE PORT' AFTER UNLOAD/
        024214  044524  047117  041040
        024222  052111  051440  052105
        024230  047440  020116  047447
        024236  050120  051517  052111
        024244  020105  047520  052122
        024252  020047  043101  042524
        024260  020122  047125  047514
        024266  042101     000
5504
5505    024271     101  052124  047105  EM15:   .ASCIZ  /ATTENTION BIT NOT SET ON PORT WHICH ISSUED 'UNLOAD'/
        024276  044524  047117  041040
        024304  052111  047040  052117
        024312  051440  052105  047440
        024320  020116  047520  052122
        024326  053440  044510  044103
        024334  044440  051523  042525
        024342  020104  052447  046116
        024350  040517  023504     000
5506
5507    024355     104  044522  042526  EM16:   .ASCII  /DRIVE NOT IN NEUTRAL AFTER UNLOAD WITH 'CONTROLLER/<15><12>
        024362  047040  052117  044440
        024370  020116  042516  052125
        024376  040522  020114  043101
        024404  042524  020122  047125
        024412  047514  042101  053440
        024420  052111  020110  041447
        024426  047117  051124  046117
        024434  042514  006522     012
5508    024441     123  046105  041505          .ASCIZ  @SELECT' SWITCH MOVED FROM 'A/B'@
        024446  023524  051440  044527
        024454  041524  020110  047515
        024462  042526  020104  051106
        024470  046517  023440  027501
        024476  023502     000
5509
5510    024501     104  044522  042526  EM17:   .ASCIZ  /DRIVE LOCKED ON PORT 'A' BY SWITCH WHILE CYCLED UP/
        024506  046040  041517  042513
        024514  020104  047117  050040
        024522  051117  020124  040447
        024530  020047  054502  051440
        024536  044527  041524  020110
        024544  044127  046111  020105
        024552  054503  046103  042105
        024560  052440  000120
5511
5512    024564  051104  053111  020105  EM20:   .ASCIZ  /DRIVE LOCKED ON PORT 'B' BY SWITCH WHILE CYCLED UP/
        024572  047514  045503  042105
        024600  047440  020116  047520
        024606  052122  023440  023502
```

```
                024614  041040  020131  053523
                024622  052111  044103  053440
                024630  044510  042514  041440
                024636  041531  042514  020104
                024644  050125     000
5513
5514    024647     123  040524  052524  EM21:   .ASCIZ  /STATUS INCORRECT FOR PORT AFTER CYCLE UP/
                024654  020123  047111  047503
                024662  051122  041505  020124
                024670  047506  020122  047520
                024676  052122  040440  052106
                024704  051105  041440  041531
                024712  042514  052440  000120
5515
5516    024720  042522  044507  052123  EM22:   .ASCIZ  /REGISTER CONTENTS SEEN WHEN DRIVE SWITCHED ON 'OPPOSITE' PORT/
                024726  051105  041440  047117
                024734  042524  052116  020123
                024742  042523  047105  053440
                024750  042510  020116  051104
                024756  053111  020105  053523
                024764  052111  044103  042105
                024772  047440  020116  047447
                025000  050120  051517  052111
                025006  023505  050040  051117
                025014  000124
5517
5518    025016  047047  042105  020047  EM23:   .ASCIZ  /'NED' SET WHEN RHDS1 ACCESSED THROUGH PORT NOT SWITCHED/
                025024  042523  020124  044127
                025032  047105  051040  042110
                025040  030523  040440  041503
                025046  051505  042523  020104
                025054  044124  047522  043525
                025062  020110  047520  052122
                025070  047040  052117  051440
                025076  044527  041524  042510
                025104  000104
5519
5520    025106  051104  053111  020105  EM24:   .ASCIZ  /DRIVE SWITCHED TO LOCKED OUT PORT WHEN RELEASED/
                025114  053523  052111  044103
                025122  042105  052040  020117
                025130  047514  045503  042105
                025136  047440  052125  050040
                025144  051117  020124  044127
                025152  047105  051040  046105
                025160  040505  042523  000104
5521
5522    025166  051104  053111  020105  EM30:   .ASCIZ  /DRIVE NOT SEIZED BY PORT/
                025174  047516  020124  042523
                025202  055111  042105  041040
                025210  020131  047520  052122
                025216     000
5523
5524    025217     127  047522  043516  EM31:   .ASCIZ  /WRONG STATUS SEEN BY THE SEIZING PORT/
                025224  051440  040524  052524
```

```
                025232  020123  042523  047105
                025240  041040  020131  044124
                025246  020105  042523  055111
                025254  047111  020107  047520
                025262  052122     000
5525
5526            025265     122  043505  051511  EM32:   .ASCIZ  /REGISTER CONTENTS WRONG/
                025272  042524  020122  047503
                025300  052116  047105  051524
                025306  053440  047522  043516
                025314     000
5527
5528            025315     103  047117  051124  EM33:   .ASCIZ  /CONTROL BUS PARITY ERROR READING INDICATED REGISTER/
                025322  046117  041040  051525
                025330  050040  051101  052111
                025336  020131  051105  047522
                025344  020122  042522  042101
                025352  047111  020107  047111
                025360  044504  040503  042524
                025366  020104  042522  044507
                025374  052123  051105     000
5529
5530            025401     103  047101  052047  EM34:   .ASCIZ  /CAN'T ACCESS DRIVE THROUGH EITHER PORT/
                025406  040440  041503  051505
                025414  020123  051104  053111
                025422  020105  044124  047522
                025430  043525  020110  044505
                025436  044124  051105  050040
                025444  051117  000124
5531
5532            025450  051104  053111  020105  EM35:   .ASCIZ  /DRIVE NOT IN NEUTRAL AFTER RELEASE - REQUEST NOT SET/
                025456  047516  020124  047111
                025464  047040  052505  051124
                025472  046101  040440  052106
                025500  051105  051040  046105
                025506  040505  042523  026440
                025514  051040  050505  042525
                025522  052123  047040  052117
                025530  051440  052105     000
5533
5534            025535     104  044522  042526  EM36:   .ASCIZ  /DRIVE NOT IN NEUTRAL AFTER TIMEOUT - REQUEST NOT SET/
                025542  047040  052117  044440
                025550  020116  042516  052125
                025556  040522  020114  043101
                025564  042524  020122  044524
                025572  042515  052517  020124
                025600  020055  042522  052521
                025606  051505  020124  047516
                025614  020124  042523  000124
5535
5536            025622  042522  044507  052123  EM37:   .ASCIZ  /REGISTER CONTENTS WRONG AFTER RELEASE OR TIMEOUT/
                025630  051105  041440  047117
                025636  042524  052116  020123
                025644  051127  047117  020107
```

```
                025652  043101  042524  020122
                025660  042522  042514  051501
                025666  020105  051117  052040
                025674  046511  047505  052125
                025702     000
5537
5538    025703     104  044522  042526  EM40:   .ASCIZ  /DRIVE IN NEUTRAL AFTER RELEASE - REQUEST SET/
        025710  044440  020116  042516
        025716  052125  040522  020114
        025724  043101  042524  020122
        025732  042522  042514  051501
        025740  020105  020055  042522
        025746  052521  051505  020124
        025754  042523  000124
5539
5540    025760  042522  044507  052123  EM41:   .ASCIZ  /REGISTER WRONG AFTER RELEASE WITH REQUEST SET/
        025766  051105  053440  047522
        025774  043516  040440  052106
        026002  051105  051040  046105
        026010  040505  042523  053440
        026016  052111  020110  042522
        026024  052521  051505  020124
        026032  042523  000124
5541
5542
5543    026036  042524  052123  021440  DH1:    .ASCIZ  /TEST #  ERR PC  PORT #  REG ADR CONTENTS/
        026044  020040  051105  020122
        026052  041520  020040  047520
        026060  052122  021440  020040
        026066  042522  020107  042101
        026074  020122  047503  052116
        026102  047105  051524     000
5544    026107     124  051505  020124  DH2:    .ASCIZ  /TEST #  ERR PC  PORT #  REG ADR GOOD    BAD/
        026114  020043  042440  051122
        026122  050040  020103  050040
        026130  051117  020124  020043
        026136  051040  043505  040440
        026144  051104  043440  047517
        026152  020104  020040  041040
        026160  042101     000
5545    026163     124  051505  020124  DH5:    .ASCIZ  /TEST #  ERR PC  REG ADR PORT A  PORT B/
        026170  020043  042440  051122
        026176  050040  020103  051040
        026204  043505  040440  051104
        026212  050040  051117  020124
        026220  020101  050040  051117
        026226  020124  000102
5546    026232  042524  052123  021440  DH6:    .ASCIZ  /TEST #  ERR PC  PORT #/
        026240  020040  051105  020122
        026246  041520  020040  047520
        026254  052122  021440     000
5547    026261     124  051505  020124  DH7:    .ASCIZ  /TEST #  ERR PC  PORT #  TIME (IN MS)/
        026266  020043  042440  051122
        026274  050040  020103  050040
```

```
                026302  051117  020124  020043
                026310  052040  046511  020105
                026316  044450  020116  051515
                026324  000051
 5548           026326  020040  020040  020040  DH13:   .ASCII  /              SEIZE/<15><12>
                026334  020040  020040  020040
                026342  020040  020040  042523
                026350  055111  006505     012
 5549           026355     124  051505  020124          .ASCIZ  /TEST #  ERR PC  PORT #/
                026362  020043  042440  051122
                026370  050040  020103  050040
                026376  051117  020124  000043
 5550           026404  020040  020040  020040  DH14:   .ASCII  /              SEIZE   ERROR/<15><12>
                026412  020040  020040  020040
                026420  020040  020040  042523
                026426  055111  020105  020040
                026434  051105  047522  006522
                026442     012
 5551           026443     124  051505  020124          .ASCIZ  /TEST #  ERR PC  PORT #  PORT #  REG ADR CONTENTS/
                026450  020043  042440  051122
                026456  050040  020103  050040
                026464  051117  020124  020043
                026472  050040  051117  020124
                026500  020043  051040  043505
                026506  040440  051104  041440
                026514  047117  042524  052116
                026522  000123
 5552           026524  042524  052123  021440  DH17:   .ASCIZ  /TEST #  ERR PC/
                026532  020040  051105  020122
                026540  041520     000
 5553           026543     040  020040  020040  DH24:   .ASCII  /              LOCKED  SWITCHED TO/<15><12>
                026550  020040  020040  020040
                026556  020040  020040  046040
                026564  041517  042513  020104
                026572  051440  044527  041524
                026600  042510  020104  047524
                026606  005015
 5554           026610  042524  052123  021440          .ASCIZ  /TEST #  ERR PC  PORT #  PORT #/
                026616  020040  051105  020122
                026624  041520  020040  047520
                026632  052122  021440  020040
                026640  047520  052122  021440
                026646     000
 5555           026647     040  020040  020040  DH30:   .ASCII  /              SEIZE   ERROR/<15><12>
                026654  020040  020040  020040
                026662  020040  020040  051440
                026670  044505  042532  020040
                026676  042440  051122  051117
                026704  005015
 5556           026706  042524  052123  021440          .ASCIZ  /TEST #  ERR PC  PORT #  PORT #  REG ADR GOOD    BAD/
                026714  020040  051105  020122
                026722  041520  020040  047520
                026730  052122  021440  020040
                026736  047520  052122  021440
```

```
                026744  020040  042522  020107
                026752  042101  020122  047507
                026760  042117  020040  020040
                026766  040502  000104
5557            026772  020040  020040  020040  DH34:   .ASCII  /              PORT A  PORT B/<15><12>
                027000  020040  020040  020040
                027006  020040  020040  047520
                027014  052122  040440  020040
                027022  047520  052122  041040
                027030  005015
5558            027032  042524  052123  021440          .ASCIZ  /TEST #  ERR PC  RHDS1   RHDS1/
                027040  020040  051105  020122
                027046  041520  020040  044122
                027054  051504  020061  020040
                027062  044122  051504  000061
5559            027070  020040  020040  020040  DH35:   .ASCII  /              RELSNG  ERROR/<15><12>
                027076  020040  020040  020040
                027104  020040  020040  042522
                027112  051514  043516  020040
                027120  051105  047522  006522
                027126     012
5560            027127     124  051505  020124          .ASCIZ  /TEST #  ERR PC  PORT #  PORT #/
                027134  020043  042440  051122
                027142  050040  020103  050040
                027150  051117  020124  020043
                027156  050040  051117  020124
                027164  000043
5561            027166  020040  020040  020040  DH37:   .ASCII  /              RELSNG  ERROR/<15><12>
                027174  020040  020040  020040
                027202  020040  020040  042522
                027210  051514  043516  020040
                027216  051105  047522  006522
                027224     012
5562            027225     124  051505  020124          .ASCIZ  /TEST #  ERR PC  PORT #  PORT #  REG ADR GOOD    BAD/
                027232  020043  042440  051122
                027240  050040  020103  050040
                027246  051117  020124  020043
                027254  050040  051117  020124
                027262  020043  051040  043505
                027270  040440  051104  043440
                027276  047517  020104  020040
                027304  041040  042101     000
5563            027311     040  020040  020040  DH40:   .ASCII  /              RELSNG  RQSTNG/<15><12>
                027316  020040  020040  020040
                027324  020040  020040  051040
                027332  046105  047123  020107
                027340  051040  051521  047124
                027346  006507     012
5564            027351     124  051505  020124          .ASCIZ  /TEST #  ERR PC  PORT #  PORT #/
                027356  020043  042440  051122
                027364  050040  020103  050040
                027372  051117  020124  020043
                027400  050040  051117  020124
                027406  000043
```

```
5565
5566                                        .EVEN
5567
5568  027410  001234  001116  001226  DT1:    .WORD   TSTNUM,$ERRPC,PTNBR,$BDADR,$BDDAT,0
      027416  001122  001126  000000
5569  027424  001234  001116  001226  DT2:    .WORD   TSTNUM,$ERRPC,PTNBR,$BDADR,$GDDAT,$BDDAT,0
      027432  001122  001124  001126
      027440  000000
5570  027442  001234  001116  001122  DT5:    .WORD   TSTNUM,$ERRPC,$BDADR,$GDDAT,$BDDAT,0
      027450  001124  001126  000000
5571  027456  001234  001116  001226  DT6:    .WORD   TSTNUM,$ERRPC,PTNBR,0
      027464  000000
5572  027466  001234  001116  001226  DT7:    .WORD   TSTNUM,$ERRPC,PTNBR,TIME,0
      027474  001244  000000
5573  027500  001234  001116  001230  DT13:   .WORD   TSTNUM,$ERRPC,SEIZPT,0
      027506  000000
5574  027510  001234  001116  001230  DT14:   .WORD   TSTNUM,$ERRPC,SEIZPT,PTNBR,$BDADR,$BDDAT,0
      027516  001226  001122  001126
      027524  000000
5575  027526  001234  001116  000000  DT17:   .WORD   TSTNUM,$ERRPC,0
5576  027534  001234  001116  001230  DT24:   .WORD   TSTNUM,$ERRPC,SEIZPT,PTNBR,0
      027542  001226  000000
5577  027546  001234  001116  001230  DT30:   .WORD   TSTNUM,$ERRPC,SEIZPT,PTNBR,$BDADR,$GDDAT,$BDDAT,0
      027554  001226  001122  001124
      027562  001126  000000
5578  027566  001234  001116  001162  DT34:   .WORD   TSTNUM,$ERRPC,$TMP2,$TMP3,0
      027574  001164  000000
5579  027600  001234  001116  001230  DT35:   .WORD   TSTNUM,$ERRPC,SEIZPT,PTNBR,0
      027606  001226  000000
5580  027612  001234  001116  001230  DT40:   .WORD   TSTNUM,$ERRPC,SEIZPT,OPPRT,0
      027620  001232  000000
5581
5582  027624      000     000     001  DF1:    .BYTE   0,0,1,0,0
      027627      000     000
5583  027631      000     000     001  DF2:    .BYTE   0,0,1,0,0,0
      027634      000     000     000
5584  027637      000     000     000  DF5:    .BYTE   0,0,0,0,0
      027642      000     000
5585  027644      000     000     001  DF6:    .BYTE   0,0,1
5586  027647      000     000     001  DF7:    .BYTE   0,0,1,1
      027652      001
5587  027653      000     000     001  DF14:   .BYTE   0,0,1,1,0,0
      027656      001     000     000
5588  027661      000     000          DF17:   .BYTE   0,0
5589  027663      000     000     001  DF30:   .BYTE   0,0,1,1,0,0,0
      027666      001     000     000
      027671      000
5590  027672      000     000     000  DF34:   .BYTE   0,0,0,0
      027675      000
5591
5592                                        .EVEN
5593
5594
5595                                ;;**********************************************************************
```

```
5596
5597                                    .SBTTL  CONSTANTS, TABLES, ETC
5598
5599                                    ;;**************************************************************************
5600
5601                                    ;TABLE OF TEST STARTING ADDRESSES
5602
5603   027676  002562                   TSTADR: .WORD   TST1+2          ;STARTING ADDRESS OF TEST 1
5606   027700  004114                           .WORD   TST2+2          ;STARTING ADDRESS OF TEST 2
  (1)  027702  004724                           .WORD   TST3+2          ;STARTING ADDRESS OF TEST 3
  (1)  027704  005534                           .WORD   TST4+2          ;STARTING ADDRESS OF TEST 4
  (1)  027706  006476                           .WORD   TST5+2          ;STARTING ADDRESS OF TEST 5
  (1)  027710  007440                           .WORD   TST6+2          ;STARTING ADDRESS OF TEST 6
  (1)  027712  011026                           .WORD   TST7+2          ;STARTING ADDRESS OF TEST 7
  (1)  027714  012414                           .WORD   TST10+2         ;STARTING ADDRESS OF TEST 10
  (1)  027716  013304                           .WORD   TST11+2         ;STARTING ADDRESS OF TEST 11
  (1)  027720  014166                           .WORD   TST12+2         ;STARTING ADDRESS OF TEST 12
  (1)  027722  015224                           .WORD   TST13+2         ;STARTING ADDRESS OF TEST 13
  (1)  027724  016316                           .WORD   TST14+2         ;STARTING ADDRESS OF TEST 14
5607
5608                                    ;ATTENTION BIT TABLE
5609
5610   027726     001                   ATABIT: .BYTE   1               ;ATTENTION BIT FOR DRIVE 0
5611   027727     002                           .BYTE   2               ;ATTENTION BIT FOR DRIVE 1
5612   027730     004                           .BYTE   4               ;ATTENTION BIT FOR DRIVE 2
5613   027731     010                           .BYTE   10              ;ATTENTION BIT FOR DRIVE 3
5614   027732     020                           .BYTE   20              ;ATTENTION BIT FOR DRIVE 4
5615   027733     040                           .BYTE   40              ;ATTENTION BIT FOR DRIVE 5
5616   027734     100                           .BYTE   100             ;ATTENTION BIT FOR DRIVE 6
5617   027735     200                           .BYTE   200             ;ATTENTION BIT FOR DRIVE 7
5618
5619   027736  000014                   MAXTN:  .WORD   $TN-1           ;MAXIMUM TEST NUMBER
5620
5621           000001                           .END
```

```
ACL    = 000040          4433#
ACU    = 100000          4403#
ADDRIS   022362          4831    5463#
ADRERR   022147          4764    5457#
AOE    = 001000          4333#
ASR1     001224          4517#   4783*
ATA    = 100000          4320#   4957    4958    5020    5021    5123    5124    5199    5200    5226    5230    5366    5367
ATABIT   027726          4783    5610#
AT0    = 000001          4353#
AT1    = 000002          4354#
AT2    = 000004          4355#
AT3    = 000010          4356#
AT4    = 000020          4357#
AT5    = 000040          4358#
AT6    = 000100          4359#
AT7    = 000200          4360#
A16    = 000400          4295#
A17    = 001000          4296#
BADNO    022334          4814    5462#
BAI    = 000010          4266#
BIT0   = 000001          4247#
BIT00  = 000001          4247#
BIT01  = 000002          4247#
BIT02  = 000004          4247#
BIT03  = 000010          4247#
BIT04  = 000020          4247#
BIT05  = 000040          4247#
BIT06  = 000100          4247#
BIT07  = 000200          4247#
BIT08  = 000400          4247#
BIT09  = 001000          4247#
BIT1   = 000002          4247#
BIT10  = 002000          4247#   5438
BIT11  = 004000          4247#   5437
BIT12  = 010000          4247#
BIT13  = 020000          4247#   5438
BIT14  = 040000          4247#
BIT15  = 100000          4247#
BIT2   = 000004          4247#
BIT3   = 000010          4247#
BIT4   = 000020          4247#
BIT5   = 000040          4247#
BIT6   = 000100          4247#
BIT7   = 000200          4247#
BIT8   = 000400          4247#
BIT9   = 001000          4247#
BPTVEC= 000014           4247#
CHANGE   002430          4477    4826#   4848
CKCLK    017376          4787    4800    5378#
CKCLK1   017446          5378    5387#
CKCLK2   017510          5388    5395#
CKCLK3   017520          5386    5394    5397#
CKERR    001236          4517#   4885*   4893*   4902*   4957*   4958*   5123*   5124*   5366*   5367*
CLOCK    017530          5382    5391    5402#
```

```
CLR    = 000040          4268#   4885
CONTUE   023144          5224    5228    5471#
CSF    = 000002          4390#
CSU    = 000010          4392#
CYCLED   023213          5366    5367    5472#
CYCLEU   023235          5366    5367    5473#
DCK    = 100000          4339#
DCL    = 000100          4434#
DE1    = 000040          4310#
DFF20  = 000002          4306#
DF1      027624          4525    4539    4574    4581    4588    4609    4637    4644    4651    4695    5582#
DF14     027653          4602    5587#
DF17     027661          4623    4630    5588#
DF2      027631          4532    4546    4681    4688    5583#
DF30     027663          4674    4723    4737    5589#
DF34     027672          4702    5590#
DF5      027637          4553    5584#
DF6      027644          4560    4595    4617    5585#
DF7      027647          4567    4658    4709    4716    4730    5586#
DH1      026036          4523    4537    4572    4579    4586    4607    4635    4642    4649    4693    5543#
DH13     026326          4593    4615    5548#
DH14     026404          4600    5550#
DH17     026524          4621    4628    5552#
DH2      026107          4530    4544    4679    4686    5544#
DH24     026543          4656    5553#
DH30     026647          4672    4735    5555#
DH34     026772          4700    5557#
DH35     027070          4707    4714    5559#
DH37     027166          4721    5561#
DH40     027311          4728    5563#
DH5      026163          4551    5545#
DH6      026232          4558    5546#
DH7      026261          4565    5547#
DIGB   = 000004          4307#
DISPLA= 177570           4247#   5437*   5438*
DLT    = 100000          4278#
DL64   = 000020          4309#
DMD    = 000001          4343#
DPR    = 000400          4313#   4902    4957    4958    5020    5021    5123    5124    5199    5200    5226    5230    5366
                         5367
DRY    = 000200          4312#   4902    4957    4958    5020    5021    5123    5124    5199    5200    5226    5230    5366
                         5367
DTE    = 010000          4336#
DTSY   = 000200          4349#
DT1      027410          4524    4538    4573    4580    4587    4608    4636    4643    4650    4694    5568#
DT13     027500          4594    4616    5573#
DT14     027510          4601    5574#
DT17     027526          4622    4629    5575#
DT2      027424          4531    4545    4680    4687    5569#
DT24     027534          4657    5576#
DT30     027546          4673    4722    4736    5577#
DT34     027566          4701    5578#
DT35     027600          4708    4715    5579#
DT40     027612          4729    5580#
```

```
DT5       027442            4552    5570#
DT6       027456            4559    5571#
DT7       027466            4566    5572#
DVA    =  004000            4298#   5123    5124
ECH    =  000100            4330#
ECI    =  004000            4415#
EMTVEC=   000030            4247#   4754*
EM1       023331            4522    5481#
EM10      023721            4571    5495#
EM11      024006            4578    5497#
EM12      024053            4585    5499#
EM13      024122            4592    5501#
EM14      024207            4599    5503#
EM15      024271            4606    5505#
EM16      024355            4613    5507#
EM17      024501            4620    5510#
EM2       023377            4529    5483#
EM20      024564            4627    5512#
EM21      024647            4634    5514#
EM22      024720            4641    5516#
EM23      025016            4648    5518#
EM24      025106            4655    5520#
EM3       023420            4536    5485#
EM30      025166            4671    5522#
EM31      025217            4678    5524#
EM32      025265            4685    5526#
EM33      025315            4692    5528#
EM34      025401            4699    5530#
EM35      025450            4706    5532#
EM36      025535            4713    5534#
EM37      025622            4720    5536#
EM4       023477            4543    5487#
EM40      025703            4727    5538#
EM41      025760            4734    5540#
EM5       023521            4550    5489#
EM6       023603            4557    5491#
EM7       023654            4564    5493#
ENTERA    022117            4759    5456#
ERR    =  040000            4319#
ERRVEC=   000004            4247#   5378*   5379*   5388*   5397*   5437*
EXEC      002262            4474    4788    4795#   4815    4872    4957    4958    5020    5021    5123    5124    5199    5200
                            5219    5366    5367    5433
EXT1   =  000001            4370#
EXT10  =  000010            4373#
EXT2   =  000002            4371#
EXT20  =  000020            4374#
EXT4   =  000004            4372#
EXT40  =  000040            4375#
FEN    =  000200            4396#
FER    =  000020            4328#
FMT22  =  010000            4416#   4957    4958    5123    5124    5199    5200    5366    5367
GNS    =  ****** U          4471    5446
GO     =  000001            4292#   5123    5124
GRV    =  000010            4308#
```

```
HCE    = 000200          4331#
HCI    = 002000          4414#
HCRC   = 000400          4332#
IAE    = 002000          4334#
IE     = 000100          4293#
ILF    = 000001          4324#
ILR    = 000002          4325#
IOTVEC= 000020           4247#   4754*
IR     = 000100          4269#
IXE    = 004000          4400#
KYBCTL  001260           4517#   4802*  4819*  4872*  4957*  4958*  5020*  5021*  5123*  5124*  5199*  5200*  5219*
                         5366*   5367*  5433
LST    = 002000          4315#
MAXTN   027736           4812    5619#
MCLK   = 000002          4344#
MCPE   = 020000          4299#
MHS    = 001000          4398#
MINX   = 000004          4345#
MOL    = 010000          4317#   4902   4957   4958   5020   5021   5123   5124   5199   5200   5226   5230   5366
                         5367
MPE    = 000400          4271#
MRD    = 000020          4347#
MSE    = 000020          4393#
MSTCK  = 000010          4346#
MWR    = 000040          4348#
MXF    = 001000          4272#
NED    = 010000          4275#   4885   5366   5367
NEM    = 004000          4274#
NHS    = 002000          4399#
NOATA  = 000001          4745#   4957   4958   5020   5021   5123#  5124#  5199   5200   5226   5230
NOCLOC  022245           4789    5460#
NORESP  022474           4845    5465#
NOSEIZ  001240           4517#
NTRH11  022442           4835    5464#
OCYL   = 100000          4436#
OFREV  = 000200          4413#
OF100  = 000004          4409#
OF200  = 000010          4410#
OF25   = 000001          4407#
OF400  = 000020          4411#
OF50   = 000002          4408#
OF800  = 000040          4412#
OPI    = 020000          4337#
OPPRT   001232           4517#   5020*  5021*  5580
OR     = 000200          4270#
PAR    = 000010          4327#
PAT    = 000020          4267#
PC     =%000007          4247#   4787*  4800*  5398*  5433*  5438*  5439*  5440*
PGE    = 002000          4273#
PGM    = 001000          4314#   4902   4957   4958   5020   5021   5123   5124   5199   5200   5226   5230
PIP    = 020000          4318#   5123   5124
PIRQ   = 177772          4247#
PIRQVE= 000240           4247#
PLU    = 020000          4402#
```

```
PORTA   001216          4517# 4761* 4762  4766  4769  4773  4779  4782  4885  4893  4902  4907  4957
                        4958  5020  5021  5123  5124  5199  5200  5222  5226  5230  5366  5367
PORTAI  022171          4772  5458#
PORTB   001220          4517# 4766* 4767* 4768* 4771* 4776  4885  4893  4902  4909  4957  4958  5020
                        5021  5123  5124  5199  5200  5222  5226  5230  5366  5367
PORTBI  022217          4775  5459#
PORTC   001222          4517# 4779* 4780* 4781*
PRE   = 000020          4432#
PRO   = 000000          4247#
PR1   = 000040          4247#
PR2   = 000100          4247#
PR3   = 000140          4247#
PR4   = 000200          4247#
PR5   = 000240          4247#
PR7   = 000340          4247#
PS    = 177776          4247# 4754* 4796* 4827*
PSEL  = 002000          4297#
PSU   = 000001          4429#
PSW   = 177776          4247#
PTNBR   001226          4517# 4885* 4893* 4902* 4957* 4958* 5020* 5021* 5123* 5124* 5199* 5200* 5226*
                        5230* 5366* 5367* 5568  5569  5571  5572  5574  5576  5577  5579
PWRVEC= 000024          4247#
RDCHR = 104412          5443  5446#
RDLIN = 104414          5444  5446#
RDOCT = 104416          4760  4808  4836  5446#
RDY   = 000200          4294#
RELERR  001242          4517# 4957* 4958* 5020* 5021* 5123* 5124* 5199* 5200* 5226* 5230*
RELOK = 000001          4744# 4957  4958  4961# 5020  5021  5123  5124  5199  5200  5226  5230
RESREG= 104422          5446#
RESVEC= 000010          4247#
RHAS  = 000016          4457#
RHBA  = 000004          4452#
RHCA  = 000034          4464#
RHCC  = 000036          4465#
RHCS1 = 000000          4450# 4957* 4958* 5020* 5021* 5123* 5124* 5199* 5200* 5222* 5226* 5230* 5366*
                        5367#
RHCS2 = 000010          4454# 4885* 4893* 4902* 4907* 4909* 4957* 4958* 5020* 5021* 5123* 5124* 5199*
                        5200* 5222* 5226* 5230* 5366* 5367*
RHDA  = 000006          4453#
RHDB  = 000022          4459#
RHDS1 = 000012          4455# 4885  4902  4957  4958  5020* 5021* 5123* 5124* 5199* 5200* 5222* 5226
                        5230  5366* 5367*
RHDT  = 000026          4461# 4893
RHEC1 = 000044          4468#
RHEC2 = 000046          4469#
RHER1 = 000014          4456#
RHER2 = 000040          4466#
RHER3 = 000042          4467#
RHLA  = 000020          4458#
RHMR  = 000024          4460#
RHOF  = 000032          4463# 4957* 4958* 5123* 5124* 5199* 5200* 5366* 5367*
RHSN  = 000030          4462# 4908  4910
RHWC  = 000002          4451#
RMR   = 000004          4326#
```

```
RP6    = 000300          4247#
R0     =%000000          4247#  4798*  4841*  4842*  4843   4846   4870*  4885*  4893*  4902*  4907*  4908   4909*
                         4910   4957*  4958*  5020*  5021*  5123*  5124*  5199*  5200*  5222*  5226*  5230*  5366*
                         5367*  5433*  5439*  5440*  5442*  5444*  5445*  5446*
R1     =%000001          4247#  4782*  4783   4809*  4812   4816*  4817*  4818   5381*  5382*  5383*  5390*  5391*
                         5392*  5439*  5442*  5444*  5445*
R2     =%000002          4247#  5442*  5444*  5445*
R3     =%000003          4247#  5441*  5442*  5443*  5445*
R4     =%000004          4247#  5441*  5445*
R5     =%000005          4247#  5020*  5021*  5414   5419*  5441*  5442*  5445*
R6     =%000006          4247#  4754*  4799*
R7     =%000007          4247#
SAVREG= 104420           5446#
SC     = 100000          4301#
SC1    = 000100          4376#
SC10   = 001000          4379#
SC2    = 000200          4377#
SC20   = 002000          4380#
SC4    = 000400          4378#
SEIZPT  001230           4517#  5020*  5021*  5123*  5124*  5199*  5200*  5366*  5367*  5573   5574   5576   5577
                         5579   5580
SKI    = 040000          4435#
SP     =%000006          4247#  4754*  4761   4769*  4770*  4771   4773*  4776*  4809   4828*  4832*  4837   4839
                         4846*  5020   5021   5123*  5124*  5199*  5200*  5387*  5395*  5396*  5412   5413*  5414*
                         5415*  5416*  5417*  5418*  5433*  5437*  5438   5439*  5440*  5441*  5442*  5443*  5444*
                         5445*  5446*
STACK  = 001100          4247#  4754   4799   4828
STANDB  022564           5123   5124   5199   5200   5466#
START   001676           4471   4754#  4849
START1  002032           4756#  4757*  4758*
STKLMT= 177774           4247#
SWR    = 177570          4247#  4914   4917   4957   4958   5020   5021   5123   5124   5199   5200   5232   5366
                         5367   5437*  5438
SWTCHA  022776           5199   5223   5366   5469#
SWTCHB  023061           5200   5227   5367   5470#
SWTCHN  022707           5199   5200   5231   5366   5367   5468#
SW0    = 000001          4247#
SW00   = 000001          4247#
SW01   = 000002          4247#
SW02   = 000004          4247#
SW03   = 000010          4247#
SW04   = 000020          4247#
SW05   = 000040          4247#
SW06   = 000100          4247#
SW07   = 000200          4247#
SW08   = 000400          4247#
SW09   = 001000          4247#  4917   4957   4958   5020   5021   5123   5124   5199   5200   5232   5366   5367
SW1    = 000002          4247#
SW10   = 002000          4247#
SW11   = 004000          4247#
SW12   = 010000          4247#
SW13   = 020000          4247#
SW14   = 040000          4247#  5199   5200   5366   5367
SW15   = 100000          4247#  4914
```

```
SW2    = 000004          4247#
SW3    = 000010          4247#
SW4    = 000020          4247#
SW5    = 000040          4247#
SW6    = 000100          4247#
SW7    = 000200          4247#
SW8    = 000400          4247#
SW9    = 001000          4247#
TBITVE= 000014           4247#
TDF    = 000040          4394#
TESTNO  022314           480?    5461#
TEST1   002634           4872    4873#
TEST10  012466           5199#
TEST11  013356           5200#
TEST12  014240           5219    5221#
TEST13  015276           5366#
TEST14  016370           5367#
TEST2   004166           4957#
TEST3   004776           4958#
TEST4   005606           5020#
TEST5   006550           5021#
TEST6   007512           5123#
TEST7   011100           5124#
TIME    001244           4517#   5020#   5021#   5402#   5572
TIMEA   001250           4517#   4786#   5020#
TIMEAP  001252           4517#   4786#   5020#   5123
TIMEB   001254           4517#   4786#   5021#
TIMEBP  001256           4517#   4786#   5021#   5124
TITLE   022014           4756    5454#
TKVEC  = 000060          4247#
TOLER   017562           5020    5021    5412#
TPVEC  = 000064          4247#
TRAPVE= 000034           4247#   4754*   4829*   4830*
TRE    = 040000          4300#
TRK1   = 004000          4381#
TRK10  = 040000          4384#
TRK2   = 010000          4382#
TRK20  = 100000          4385#
TRK4   = 020000          4383#
TRTVEC= 000014           4247#
TSTADR  027676           4818    5603#
TSTNUM  001234           4517#   5438*   5568    5569    5570    5571    5572    5573    5574    5575    5576    5577    5578
                         5579    5580
TST1    002560           4811    4872#   5603
TST1AA  002554           4870#   5433
TST10   012412           5124    5199#   5606
TST11   013302           5199    5200#   5606
TST12   014164           5200    5219#   5606
TST13   015222           5232    5366#   5606
TST14   016314           5366    5367#   5606
TST2    004112           4917    4957#   5606
TST3    004722           4957    4958#   5606
TST4    005532           4958    5020#   5606
TST5    006474           5020    5021#   5606
```

```
TST6    007436          5021    5123#   5606
TST7    011024          5123    5124#   5606
TUF   = 000100          4395#
TYPDS = 104410          4774    4777    5123    5124    5199    5200    5433    5439    5446#
TYPE  = 104400          4756    4759    4764    4772    4775    4778    4789    4797    4807    4814    4831    4834    4835
                        4845    5123    5124    5199    5200    5223    5224    5227    5228    5231    5366    5367    5433
                        5438    5439    5441    5442    5443    5444    5446#
TYPOC = 104402          4833    4847    5439    5446#
TYPON = 104406          5446#
TYPOS = 104404          5446#
UNS   = 040000          4338#
UPE   = 020000          4276#
US1   = 000001          4263#
US2   = 000002          4264#
US4   = 000004          4265#
UWR   = 000010          4431#
VUF   = 000002          4430#
VU30  = 010000          4401#
VV    = 000100          4311#   4957    4958    5020    5021    5123    5124    5199    5200    5226    5230
VVSET = 000001          4743#   4957    4958    4962#   5020    5021    5123#   5124#   5199#   5200#   5226    5230
WATCH   001246          4517#   5020*   5021*   5123*   5124*   5403    5405*   5407*
WCE   = 040000          4277#
WCF   = 000040          4329#
WCU   = 000001          4389#
WLE   = 004000          4335#
WRL   = 004000          4316#
WRU   = 000400          4397#
WSU   = 000004          4391#
$BDADR  001122          4517#   4885*   4893*   4902*   4957*   4958*   5020*   5021*   5123*   5124*   5199*   5200*   5226*
                        5230#   5366*   5367*   5568    5569    5570    5574    5577
$BDDAT  001126          4517#   4885*   4893*   4902*   4910*   4911    4957*   4958*   5020*   5021*   5123*   5124*   5199*
                        5200*   5226*   5230*   5366*   5367*   5568    5569    5570    5574    5577
$BELL   001174          4517#   5438
$CMTAG  001100          4517#   4754
$CM1  = 000001          4517#
$CM2  = 000002          4517#
$CM3  = 000001          4517#
$CM4  = 000005          4517#
$CNTLU  021500          5443#
$CRLF   001201          4517#   4778    4797    4834    5123    5124    5199    5200    5366    5438    5439    5443    5444
$DBLK   021200          5442#
$DOAGN  017736          5433#
$DTBL   021170          5442#
$ENDAD  017726          4517    5433#
$ENDCT  017656          4754    5433#
$ENDMG  017742          5433#
$ENULL  017757          5433#
$EOP    017610          5367    5433#
$EOPCT  017650          4754*   5433#
$ERFLG  001103          4517#   4917*   4957*   4958*   5020*   5021*   5123*   5124*   5199*   5200*   5232*   5366*   5367*
                        5437*   5438*
$ERMAX  001115          4517#   4754*   4804*   5437
$ERROR  020140          4754    5438#
$ERRPC  001116          4517#   5438*   5439    5568    5569    5570    5571    5572    5573    5574    5575    5576    5577
```

```
                        5578    5579    5580
$ERRTB  001266          4517#   5439
$ERRTY  020254          5438    5439#
$ERTTL  001112          4517#   5438*
$ESCAP  001172          4517#   4754*
$FILLC  001150          4517#   5440
$FILLS  001147          4517#   5440
$GDADR  001120          4517#
$GDDAT  001124          4517#   4885*   4893*   4902*   4908*   4911    4957*   4958*   5020*   5021*   5123*   5124*   5199*
                        5200*   5226*   5230*   5366*   5367*   5569    5570    5577
$GET42  017700          5433#
$HD   = 000000          4242
$HIOCT  021652          5444#*
$ICNT   001104          4517#   4820*   5437*
$ITEMB  001114          4517#   5438*   5439
$LF     001202          4517#   5438    5443    5444
$LKCSB  001206          4517#   5384*
$LKCSR  001204          4517#   5380    5385*
$LKS    001212          4517#   5389    5393*
$LLVEC  001214          4517#   5390
$LPADR  001106          4517#   4754*   4805*   4872*   4957*   4958*   5020*   5021*   5123*   5124*   5199*   5200*   5219*
                        5366*   5367*   5437*
$LPERR  001110          4517#   4754*   4806*   4872*   4917    4957*   4958*   5020*   5021*   5123*   5124*   5199*   5200*
                        5219*   5232    5366*   5367*
$LPVEC  001210          4517#   5381
$MXCNT  020136          5437#
$NULL   001146          4517#   5440
$NWTST= 000001          4872#   4957#   4958#   5020#   5021#   5123#   5124#   5199#   5200#   5219#   5366#   5367#
$OCNT   020760          5441#*
$OMODE  020762          5441#*
$OVER   020122          5437#
$PASS   001100          4517#   4803*   5433*   5437
$QUES   001200          4517#   5438    5443    5444
$RDCHR  021210          5443#   5446
$RDDEC= ****** U        5446
$RDLIN  021244          5443#   5446
$RDOCT  021514          5444#   5446
$RDSZ = 000007          5443#
$REGAD  001152          4517#
$REGO   001154          4517#
$RESRE  021712          5445#   5446
$RPADR  001262          4517#   4798    4832    4839*   4841    4870
$RPVEC  001264          4517#
$SAVRE  021654          5445#   5446
$SCOPE  017762          4754    5437#
$SETUP= 000027          4741#   4754    5433
$STUP = 177777          4741#
$SVLAD  020112          5437#
$SWR  = 166000          4238#   4242    4244    4517    4754    4872    4957    4958    5020    5021    5123    5124    5199
                        5200    5219    5366    5367    5433    5435#   5437    5438
$SWRMK= 000000          5437
$TIMES  001170          4517#   4754*   4872*   4917*   4957*   4958*   5020*   5021*   5123*   5124*   5199*   5200*   5219*
                        5232*   5366*   5367*   5433*   5437*
$TKB    001140          4517#   5443
```

```
$TKS    001136          4517#  4754   5443
$TMP0   001156          4517#  4885*  4902*  4957*  4958*  5020*  5021*  5123*  5124*  5199*  5200*  5226*  5230*
                        5366*  5367*
$TMP1   001160          4517#  4957*  4958*  5020*  5021*  5123*  5124*  5199*  5200*  5226*  5230*
$TMP2   001162          4517#  4957*  4958*  5020*  5021*  5123*  5124*  5199*  5200*  5226*  5230*  5578
$TMP3   001164          4517#  4957*  4958*  5020*  5021*  5123*  5124*  5199*  5200*  5226*  5230*  5578
$TMP4   001166          4517#  4885*  4902*  4957*  4958*  5123*  5124*  5366*  5367*
$TN   = 000015          4239#  4242   4872#  4873   4917   4957#  4958#  5020#  5021#  5123#  5124#  5199#  5200#
                        5219#  5221   5232   5366#  5367#  5619
$TPB    001144          4517#  5440*
$TPFLG  001151          4517#  5440
$TPS    001142          4517#  5440
$TRAP   021750          4754   4829   5446#
$TRP  = 000024          5446#
$TRPAD  021770          5446#
$TSTNM  001102          4517#  4872*  4957*  4958*  5020*  5021*  5123*  5124*  5199*  5200*  5219*  5366*  5367*
                        5433*  5437*  5438
$TTYIN  021505          5443#
$TYPBN= ****** U        5446
$TYPDS  020764          5442#  5446
$TYPE   020430          5440#  5446
$TYPEC  020520          5440#
$TYPOC  020562          5441#  5446
$TYPON  020576          5441#  5446
$TYPOS  020536          5441#  5446
$XTSTR  017770          5437#
$STRP = 000002          5446#
$OFILL  020761          5441#*
.     = 027740          4471#  4517#  4754   4791   4805   4806   4810   4885   5020   5021   5433   5437   5438
                        5439#  5440   5442#  5443#  5444
```

```
AA          4965#  5020   5021
BB          4919#  4957   4958
CC          5023#  5123   5124
CHECK       3783#  4885   4893   4902   4957   4958   5123   5124   5366   5367
CLRATA      4068#  5123   5124   5199   5200   5222
COMMEN       904#  4247#
DD          5126#  5199   5200
ENDCOM       916#  4247#
ERROR       4247#  4885   4893   4902   4913   4957   4958   5020   5021   5123   5124   5199   5200   5226   5230
            5366   5367
ESCAPE      1018#  4247#
FF          5234#  5366   5367
HEAD        4130#  4872   4957   4958   5020   5021   5123   5124   5199   5200   5219   5366   5367
LOOP        4176#  4917   4957   4958   5020   5021   5123   5124   5232
MORETA      4481#  4517
MSG         4857#  4872   4957#  4958#  5020#  5021#  5123#  5124#  5199#  5200#  5202#  5219   5366#  5367#
MULT        3201#  4247#
NEUTRA      3923#  4957   4958   5020   5021   5123   5124   5199   5200   5226   5230
NEWTST       956#  4247#  4872   4957   4958   5020   5021   5123   5124   5199   5200   5219   5366   5367
POP         1351#  4247#  5442   5444   5445
PUSH        1343#  4247#  5442   5444   5445
RELEAS      3854#  4957   4958
SCOPE       4247#  4872   4957   4958   5020   5021   5123   5124   5199   5200   5219   5366   5367   5433
SEIZE       3693#  5020   5021
SELECT      3897#  4885   4893   4902   4957   4958   5020   5021   5123   5124   5199   5200   5366   5367
SETATA      4102#
SETTRA      5446#
SETUP        738#  4247#  4754
SKIP        1052#  4247#  4917   4957   4958   5020   5021   5123   5124   5199   5200   5232   5366
SLASH        856#  4247#
SPACE       4247#
STARS        825#  4247#  4517   4872   4957   4958   5020   5021   5123   5124   5199   5200   5219   5366   5367
            5433   5437   5438   5439   5440   5441   5442   5443   5444   5445   5446
STARSS      4199#  4249   4253   4284   4288   4444   4448   4517   4747   4751   4851   4855   4873   4875   4887
            4895   4904   4957   4958   5020   5021   5123   5124   5199   5200   5221   5366   5367   5369   5373
            5421   5425   5448   5452   5475   5479   5595   5599
SUBTAG      4217#  4873   4957   4958   5020   5021   5123   5124   5199   5200   5221   5366   5367
SWITCH      4024#
TIMER       4154#  5020   5021
TRMTRP      5446#
TYPBIN      1287#  4247#
TYPDEC      1257#  4247#  5123   5124   5199   5200   5433   5439
TYPNUM      1224#  4247#
TYPOCS      1177#  4247#
TYPOCT      1140#  4247#  5439
TYPTXT      1094#  4247#
USEREQ      5427#  5433
$$CMRE      4517#
$$CMTM      4517#
$$ESCA      1031#  4247#
$$NEWT       985#  4247#  4872   4957   4958   5020   5021   5123   5124   5199   5200   5219   5366   5367
$$SET       5446#
$$SKIP      1065#  4247#  4917   4957   4958   5020   5021   5123   5124   5199   5200   5232   5366
.EQUAT       164#  4234#  4247
```

```
.HEADE     39#   4234#   4242
.KT11     300#
.SETUP    684#   4234#   4741
.SWRHI     81#   4234#   4244
.SWRLO   4234#   4244#
.$CATC    478#   4235#   4471
.$CMTA    535#   4235#   4517
.$DB2D   3401#
.$DB20   3525#
.$DIV    3303#
.$EOP    1399#   4235#   5433
.$ERRO   1773#   4235#   5438
.$ERRT   1928#   4234#   5439
.$MULT   3239#
.$POWE   2964#
.$RAND   3021#
.$RDDE   2660#
.$RDOC   2568#   4236#   5444
.$READ   2354#   4236#   5443
.$SAVE   2736#   4236#   5445
.$SB2D   3486#
.$SB20   3588#
.$SCOP   1575#   4235#   5437
.$SIZE   3083#
.$SUPR   3626#
.$TRAP   2837#   4236#   5446
.$TYPB   2269#
.$TYPD   2191#   4236#   5442
.$TYPE   2016#   4235#   5440
.$TYPO   2094#   4236#   5441
```

CROSS REFERENCE TABLE

```
NOP     4757  4758  4801  4885  4893  4902  4957  4958  5020  5021  5123  5124  5199  5200  5226
        5230  5366  5367  5433
RESET   4755  4795  4826  5433
ROL     5437  5441  5444
RTI     5408  5437  5438  5440  5441  5442  5443  5444  5445
RTS     5398  5419  5439  5440  5446
SUB     5405  5438  5442
TRAP    5446
TST     4837  4843  4872  4885  4957  4958  5020  5021  5123  5124  5199  5200  5219  5226  5230
        5366  5367  5380  5389  5403  5412  5433  5437  5438  5439  5440  5441  5442  5443  5444
        5446
TSTB    4917  4957  4958  5020  5021  5123  5124  5199  5200  5232  5366  5367  5437  5439  5440
        5442  5443
.ASCII  4517  5454  5466  5507  5548  5550  5553  5555  5557  5559  5561  5563
.ASCIZ  4517  5433  5439  5443  5455  5456  5457  5458  5459  5460  5461  5462  5463  5464  5465
        5467  5468  5469  5470  5471  5472  5473  5481  5483  5485  5487  5489  5491  5493  5495
        5497  5499  5501  5503  5505  5508  5510  5512  5514  5516  5518  5520  5522  5524  5526
        5528  5530  5532  5534  5536  5538  5540  5543  5544  5545  5546  5547  5549  5551  5552
        5554  5556  5558  5560  5562  5564
.BLKB   5443
.BLKW   5442
.BYTE   4517  5433  5441  5443  5582  5583  5584  5585  5586  5587  5588  5589  5590  5610  5611
        5612  5613  5614  5615  5616  5617
.ENABL     4  4231
.END    5621
.ENDC   4242  4244  4247  4471  4517  4741  4754  4872  4885  4893  4902  4917  4957  4958  5020
        5021  5123  5124  5199  5200  5219  5222  5226  5230  5232  5366  5367  5433  5437  5438
        5439  5440  5441  5442  5443  5444  5445  5446
.EQUIV  4247
.EVEN   5439  5566  5592
.IF     4242  4244  4247  4471  4517  4741  4754  4872  4885  4893  4902  4917  4957  4958  5020
        5021  5123  5124  5199  5200  5219  5222  5226  5230  5232  5366  5367  5433  5437  5438
        5439  5440  5441  5442  5443  5444  5445  5446
.IFF    4244  4247  4517  4754  4872  4885  4893  4902  4917  4957  4958  5020  5021  5123  5124
        5199  5200  5219  5222  5226  5230  5232  5366  5367  5433  5437  5438  5439  5440  5441
        5442  5443  5444  5445  5446
.IFT    5020  5021  5123  5124  5199  5200  5222  5437  5438  5443  5444
.IFTF   5020  5021  5123  5124  5199  5200  5222  5437  5438  5443  5444
.IIF    4242  4244  4471  4517  4754  5123  5124  5199  5200  5433  5437  5439  5439  5440  5443
        5444  5446
.IRP    4517  4660  4741  4784  4872  4878  4890  4898  4957  4958  5020  5021  5123  5124  5199
        5200  5219  5366  5367  5433  5438  5442  5444  5445  5604
.LIST      2  3664  4226  4229  4240  4247  4471  4517  4741  4753  4872  4957  4958  4963  5020
        5021  5123  5124  5199  5200  5219  5366  5367  5436  5437  5443  5446
.MACRO    39    81   164   300   478   535   684   738   825   856   904   916   956   985  1018
        1031  1052  1065  1094  1140  1177  1224  1257  1287  1343  1351  1399  1575  1773  1928
        2016  2094  2191  2269  2354  2568  2660  2736  2837  2964  3021  3083  3201  3239  3303
        3401  3486  3525  3588  3626  3693  3783  3854  3897  3923  4024  4068  4102  4130  4154
        4176  4199  4217  4244  4481  4517  4857  4919  4957  4958  4965  5020  5021  5023  5123
        5124  5126  5199  5200  5202  5234  5366  5367  5427  5446
.MCALL  4234  4235  4236  4247
.NLIST     1     3  3666  4228  4230  4247  4471  4517  4741  4742  4872  4957  4958  4960  5020
        5021  5123  5124  5199  5200  5219  5366  5367  5434  5437  5443  5446
.PAGE   4517
```

DZRPQA.P11 CROSS REFERENCE TABLE

```
.REPT    4471    4517
.SBTTL   4244    4247    4251    4286    4446    4471    4517    4749    4853    4872    4957    4958    5020    5021    5123
         5124    5199    5200    5219    5366    5367    5371    5423    5433    5437    5438    5439    5440    5441    5442
         5443    5444    5445    5446    5450    5477    5597
.TITLE   4242
.WORD    4471    4517    4822    5020    5021    5433    5439    5441    5444    5568    5569    5570    5571    5572    5573
         5574    5575    5576    5577    5578    5579    5580    5603    5606    5619


 ERRORS DETECTED:  0

 *DZRPQA,DZRPQA/CRF=SYSMAC.SMA,DZRPQA
 RUN-TIME:  64 62 5 SECONDS
 CORE USED:  29K
```

E09

Spooler runtime 3 Seconds, 36 KCS, 461 disk reads, 1 disk writes, 115 pages printed

[redacted]