RP04
RP05, RP06

RP04/5/6 MLT-DR LGC
CZRJDE0

AH-9197E-MC
FICHE 1 OF 2

APR 1982

RP04 RP04/5/6 MLT-DR LGC AH-9197E-MC APR 1982
RP05,RP06 CZRJDEO FICHE 2 OF 2

digital

```
                .REM    @











                                                IDENTIFICATION
                                                --------------

                                PRODUCT CODE:   AC-9195E-MC

                                PRODUCT NAME:   CZRJDEO  RP04/5/6 MULTI-DRIVE LOGIC TEST

                                PRODUCT DATE:   NOVEMBER 1981

                                MAINTAINER:     CX DIAGNOSTIC GROUP

                                AUTHOR:

                                REVISED BY:     MIKE LEAVITT


                THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT NOTICE AND
                SHOULD  NOT  BE  CONSTRUED  AS  A  COMMITMENT  BY  DIGITAL  EQUIPMENT
                CORPORATION.  DIGITAL EQUIPMENT CORPORATION  ASSUMES  NO  RESPONSIBILITY
                FOR ANY ERRORS THAT MAY APPEAR IN THIS MANUAL.

                THE SOFTWARE DESCRIBED IN THIS DOCUMENT IS FURNISHED UNDER A LICENSE AND
                MAY ONLY BE USED OR COPIED IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE.

                DIGITAL  EQUIPMENT  CORPORATION ASSUMES NO RESPONSIBILITY FOR THE USE OR
                RELIABILITY OF ITS  SOFTWARE  ON  EQUIPMENT  THAT  IS  NOT  SUPPLIED  BY
                DIGITAL.

                    COPYRIGHT (C) 1976,1979,1982 DIGITAL EQUIPMENT CORPORATION
                    CONTENTS
                    --------
```

1. INTRODUCTION

1.1 REVISION HISTORY

1. CHANGE TITLE FROM CZRJDCOTO CZRJDDO

2. MODIFY STARTING ADDRESSES TO SAFEGUARD DRIVES IN MULTI-PROCESSOR ENVIRONMENT. SEE PARAGRAPH 2.4 BELOW.

3. MODIFY CODE IN ''CLEAR DRIVE PARAMETER BLOCK'' SUBROUTINE (CLRDPB) TO PREVENT CLEARING LOCATION $RPDT.

4. MODIFY CODE IN ''DRIVE INITIALIZATION'' ROUTINE (DRVINT) TO STORE DRIVE TYPE IN APPROPRIATE DRIVE PARAMETER BLOCK.

5. INSTALL XON/XOFF FIXES FOR HIGH TERMINALS AND THE DIANA BOX USED BY REMOTE DIAGNOSTICS (RD). (REV E)

1.2 ABSTRACT

THE RP04/5/6 MULTIDRIVE EXERCISER PROGRAM IS DESIGNED TO PERFORM AN INTERACTIVE TEST ON RP04/5/6 DISK DRIVES CONNECTED TO A MASSBUS SUBSYSTEM. THE SUBSYSTEM MAY BE COMPOSED OF INTERMIXED RP04, RP05 OR RP06 DISK DRIVES CONTROLLED BY EITHER AN RH11 OR AN RH70. IN ADDITION TO PERFORMING AN INTERACTIVE TEST OF THE DISK DRIVES ON THE SUBSYSTEM, THE PROGRAM IS INTENDED TO BE USED TO VERIFY THAT THE DRIVES UNDER TEST ARE PERFORMING TO THEIR DATA ERROR RATE AND SEEK ERROR RATE SPECIFICATIONS.

THE RP04/5/6 MULTIDRIVE EXERCISER PROGRAM WILL EXERCISE DRIVES CONNECTED AS EITHER SINGLE OR DUAL PORT UNITS. DUAL PORT DRIVES ARE TESTED BY LOADING AND RUNNING THE PROGRAM FROM BOTH CONTROLLING SYSTEMS. THE PROGRAM WILL EXERCISE A MIXED SYSTEM OF DUAL PORT AND NON DUAL PORT DRIVES. PROGRAMMABLE OPERATION IS INHIBITED WITH STARTING ADDRESS 200-SEE SEC 2.4. THEREFORE, USE STARTING ADDRESS 204 OR 220 FOR DUAL PORT TESTING.

TO OBTAIN INTERACTIVE TESTING, OPERATIONS ON THE DRIVES ARE OVERLAPPED (OTHER DRIVES ARE PERFORMING SEEK/SEARCH OPERATIONS WHILE ONE DRIVE IS PERFORMING A DATA TRANSFER OR WRITE CHECK OPERATION). OPERATIONS AMONG THE DRIVES ARE OPTIMIZED SO THAT A HIGH SUBSYSTEM DATA TRANSFER RATE OR A HIGH POSITIONING OPERATION RATE IS MAINTAINED.

THE PERFORMANCE OF EACH DRIVE IS MONITORED BY THE PROGRAM. IF A DRIVE EXCEEDS A PRESET NUMBER OF ERRORS IN ANY OF SEVERAL CATEGORIES, THAT DRIVE IS AUTOMATICALLY DEASSIGNED. (THE OPERATOR MAY OVERRIDE THE AUTOMATIC DEASSIGNMENT FEATURE.) THE PROGRAM REPORTS PERFORMANCE STATISTICS FOR EACH DRIVE BEING EXERCISED ON REQUEST FROM THE OPERATOR OR AUTOMATICALLY AT AN INTERVAL DETERMINED BY THE OPERATOR.

ALL DATA TRANSFER COMMANDS ARE USED (I.E., WRITE DATA, WRITE HEADER & DATA, READ DATA, AND READ HEADER & DATA) AS WELL AS WRITE CHECK DATA AND WRITE CHECK HEADER & DATA COMMANDS. RECALIBRATE AND READ-IN PRESET COMMANDS ARE USED AT STARTUP AND DRIVE INITIALIZATION. RECALIBRATE, OFFSET, AND RETURN-TO-CENTERLINE COMMANDS ARE USED DURING ERROR PROCESSING.

```

                THE DATA TRANSFER COMMANDS ARE SELECTED RANDOMLY EXCEPT FOR THE
                WRITE CHECK COMMANDS.  THE WRITE CHECK COMMANDS ARE USED TO VERIFY
                A PREVIOUS WRITE OPERATION.  THUS, WHEN A WRITE COMMAND IS SELECTED,
                THE DATA WRITTEN IS VERIFIED BY THE APPROPRIATE WRITE CHECK
                COMMAND.

                PROGRAM/OPERATOR COMMUNICATIONS ARE THROUGH THE KEYBOARD; DYNAMIC
                PROGRAM OPTIONS ARE SELECTED BY SWITCH REGISTER SETTINGS.  ERRORS
                ARE REPORTED ON THE TELETYPE.

                ALL COMMANDS, DATA PATTERNS, AND DATA BUFFER SIZES ARE SELECTED
                RANDOMLY BY THE PROGRAM.  ADDITIONALLY THE ADDRESSES (EG, CYLINDER,
                TRACK, AND SECTOR) FOR EACH OPERATION ARE SELECTED RANDOMLY.


        2.      REQUIREMENTS
                ------------

        2.1     EQUIPMENT

                               REQUIRED
                               --------

                        PDP-11 PROCESSOR
                        16K MEMORY (20K IF THE PROGRAM IS INCLUDED IN AN 'XXDP' CHAIN
                        TELETYPE
                        PROGRAM LOADING DEVICE
                        KW11-L OR KW11-P CLOCK
                        RH11 OR RH70 WITH 1 RP04, RP05, OR RP06 DISK DRIVE

                               OPTIONAL
                               --------

                        ADDITIONAL MEMORY TO A MAXIMUM OF 28K
                        1 TO 7 ADDITIONAL RP04/5/6'S ON THE SAME RH11 OR RH70

        2.2     MEDIA

                THE RP04/5/6 MULTIDRIVE EXERCISER PROGRAM REQUIRES FORMATTED DISK PACKS
                WHICH CONTAIN RANDOM OR PATTERNED DATA RECOGNIZED BY THE EXERCISER.
                DISK PACKS USED BY THE PROGRAM MAY BE GENERATED BY THE RP04/5/6 FORMATTER
                PROGRAM (DZRJB) OR BY THE 'W' COMMAND OF THE RP04/5/6 MULTIDRIVE
                EXERCISER (SEE SECTION 4.4).  THE PACKS MUST BE FORMATTED IN 22
                SECTOR (16 BIT) MODE; THE ALTERNATE (20 SECTOR - 18 BIT) MODE IS
                NOT SUPPORTED.


        2.3     PRELIMINARY PROGRAMS

                RP04/5/6 DISKLESS CONTROLLER TEST
                        PART 1 (CZRJG)
                        PART 2 (CZRJH)

                RP04/5/6 FUNCTIONAL CONTROLLER TEST
                        PART 1 (CZRJI)
                        PART 2 (CZRJJ)
                RP04/5/6 DUAL CONTROLLER LOGIC TEST (FOR DUAL PORT DRIVE TESTING)
                        PART 1 (CZRJE)
                        PART 2 (CZRJF)

        2.4     PROGRAMMABLE DRIVES (DUAL PORT ENABLED)

                THIS REV INCORPORATES A SAFEGUARD TO PREVENT INADVERTENT
                CORRUPTION OF DISK PACKS IN PROGRAMMABLE DRIVES.
                THIS IS A POTENTIAL HAZARD IN RUNNING THIS PROGRAM IN A
                MULTIPROCESSOR SYSTEM.  FOR THE STANDARD STARTING ADDRESS OF 200
                THE PROGRAM HAS BEEN MODIFIED TO PREVENT INITIALIZING DRIVES
                FOUND TO BE PROGRAMMABLE.  THIS MODIFICATION APPLIES
                ONLY TO THE FIELD ENVIRONMENT (XXDP CHAIN, STANDALONE)
                WHERE LOCATION 42 DOES NOT EQUAL LOCATION 46.  FOR THE
                MANUFACTURING ENVIRONMENT (WHERE LOCATION 42 EQUALS LOCATION 46) PROGRAMMABLE
                DRIVES WILL NOT BE INHIBITED.  IF THE OPERATOR DESIRES TO RUN THIS PROGRAM USING
                PROGRAMMABLE DRIVES IN A FIELD ENVIRONMENT USE STARTING ADDRESS 220, WHERE 220 IS
                THE SAME AS 200 WITHOUT INHIBITING PROGRAMMABLE DRIVES.
                SEE SECTION 3.2,3.3 FOR A SUMMARY OF ALL STARTING ADDRESSES.




        3.      OPERATING THE PROGRAM
                ---------------------

        3.1     THE PROGRAM MAY BE LOADED FROM PAPER TAPE USING THE ABSOLUTE LOADER OR
                IT MAY BE LOADED FROM THE APPROPRIATE 'XXDP' MEDIA USING THE ASSOCIATED
                LOADER.  THE PROGRAM MAY BE INCLUDED IN AN 'XXDP' CHAIN.  IF THE
                PROGRAM IS BEING RUN ON A PROCESSOR WITH 16K, THE 'XXDP' LOADER WILL
                NOT BE PRESERVED.  THE PROGRAM MUST BE RUN ON A SYSTEM WITH 20K OR
                MORE TO PRESERVE THE 'XXDP' LOADER.  THE 'ABSOLUTE' LOADER WILL BE
                PRESERVED IN A 16K SYSTEM, HOWEVER.

        3.2     THE PROGRAM STARTS AT LOCATION 200(8).  PARAMETERS NOT INCLUDED IN
                THE TELETYPE DIAGLOGUE GROUP MUST BE CHANGED BEFORE THE PROGRAM
                IS STARTED. PROGRAMMABLE DRIVES ARE INHIBITED STANDALONE AND
                XXDP CHAIN-SEE SEC  2.4.

        3.3     START THE PROGRAM AT LOCATION 204(8) IF THE RH11 OR THE RH70 IS
                NOT AT ADDRESS 176700. (NO INHIBITIONS)

                STARTING THE PROGRAM AT 220 IS THE SAME AS 200 BUT PROGRAMMABLE
                DRIVES ARE NOT INHIBITED.

        3.4     ONCE THE PROGRAM IS LOADED AND STARTED, OPERATIONS ARE INITIATED AND
                CONTROLLED BY KEYBOARD COMMANDS AND SWITCH REGISTER SWITCH SETTINGS.

                IF THIS IS THE PROGRAM'S FIRST START, THE STATUS OF THE DRIVES ON
                THE SELECTED MASSBUS SUBSYSTEM WILL BE TYPED OUT.  ON SUBSEQUENT
                STARTS, THIS TYPEOUT MAY BE INHIBITED BY SETTING SW<02>.

        3.5     PASS/TEST TERMINATION

                A PASS IS DETERMINED BY EITHER BITS READ OR SEEKS PERFORMED.  THE
                NUMBER OF BITS OR SEEKS REQUIRED FOR A PASS IS DERIVED FROM EITHER
```

```
            THE SOFT ERROR RATE SPECIFICATION OR THE SEEK ERROR RATE SPECIFICATION.
            THE SPECIFICATIONS FOR RP04'S, RP05'S, AND RP06'S SPECIFY NO MORE
            THAN 1 SOFT ERROR (NON-PACK RELATED) IN 1 X 10^9 BITS READ OR NO
            MORE THAN 1 SEEK ERROR IN 1 X 10^6 SEEKS.  THE NUMBER OF BITS OR
            SEEKS DETERMINING A PASS WERE SELECTED TO PROVIDE A 90% CONFIDENCE
            LEVEL THAT THE DRIVE IS PERFORMING TO THE APPLICABLE SPECIFICATION.

3.5.1       PASS TERMINATION

            END OF PASS MAY BE DETERMINED BY EITHER OF THE FOLLOWING CONDITIONS.
            THE END OF PASS CONDITION USED IS DETERMINED BY PARAMETER 'ENDET'.

            A.  IF PARAMETER 'ENDET' IS 1, END OF PASS OCCURS WHEN THE DRIVE
                HAS READ 1.875 X 10^8 WORDS (3 X 10^9 BITS).
            B.  IF PARAMETER 'ENDET' IS 0, END OF PASS OCCURS WHEN THE DRIVE
                HAS PERFORMED 3 X 10^6 SEEKS.

3.5.2       TEST TERMINATION

            THE TEST FOR A DRIVE IS TERMINATED (SW<04> = 0) WHEN:

            A.  THE DRIVE HAS COMPLETED THE NUMBER OF PASSES SPECIFIED IN
                PARAMETER 'PASCNT'.
            B.  THE TOTAL ERRORS ACCUMULATED EXCEED 100.
            C.  A FATAL ERROR OCCURS: EM12 OR EM14.

3.6         RUN TIME

            THE EXERCISER PROGRAM MAY BE RUN IN TWO MODES.  THE MODE IS
            DETERMINED BY THE VALUE IN PARAMETER 'MAXDL'.  IF 'MAXDL' IS ONE
            SECTOR, THE PROGRAM RUNS IN A SEEK HEAVY MODE; IF 'MAXDL' APPROACHES
            ONE TRACK IN SIZE (5720 DECIMAL) THE PROGRAM RUNS IN A DATA TRANSFER
            HEAVY MODE.  THE PROGRAM RUN TIME VARIES GREATLY DEPENDING ON
            THE OPERATION MODE SELECTED, THE MEMORY AVAILABLE OVER 16K, THE
            READ/WRITE RATIO PARAMETER - 'RATIO', AND BY SWITCHES 0, 1, & 2.

3.6.1       DATA TRANSFER MODE

            1 DRIVE - APPROXIMATELY 2.5 HRS (TO REACH 1.875 X 10^8 WORDS)
              TO
            8 DRIVES - APPROXIMATELY 11 HRS (FOR ALL DRIVES TO REACH 1.875 X 10^8 WORDS)

            NOTE:  IF SW<01> = 1 (NO SOFTWARE DATA COMPARSIONS), THE RUN TIMES
                   ARE THE FOLLOWING VALUES, APPROXIMATELY:

                     1 DRIVE - 1.7 HRS (1.875 X 10^8 WORDS READ)
                         ADD 1/2 HOUR FOR EACH ADDITIONAL DRIVE TESTED.

                     IF THE PROGRAM IS RUN WITH BOTH SW<00> AND SW<01>
                     SET, THE RUN TIMES SHOULD BE ABOUT 20% FASTER.

            NOTE:    ON THE FIRST PASS (QUICK VERIFY) THE TIMES ARE
                     APPROXIMATELY ONE EIGHTH OF THE ABOVE VALUES.

3.6.2       SEEK VERIFICATION MODE

            PARAMETER 'MAXDL' = 1 SECTOR (256 WORDS)
```

PARAMETER 'MAXTRK' = 'MINTRK'
PARAMETER 'MAXSEC' = 'MINSEC'
SW<00> = 1 (READ ONLY MODE)

1 DRIVE - APPROXIMATELY 25 HRS (3 X 10^6 SEEKS)
  TO
8 DRIVES - APPROXIMATELY 40 HRS (3 X 10^6 SEEKS FOR ALL DRIVES)

3.7 UNIBUS & VECTOR ADDRESSES

THE PROGRAM ASSUMES THE FOLLOWING UNIBUS AND VECTOR ADDRESSES. (REFER TO SECTION 4.2.2 FOR THE LOCATIONS AT WHICH TO CHANGE THESE ADDRESSES.)

UNIT	UNIBUS ADDRESS	VECTOR ADDRESS
RH11 OR RH70	176700	254
TTY PRINTER	177564	NOT USED
TTY KEYBOARD	177560	60
KW11-L	177546	100
KW11-P	172542	104

3.8 DUAL PORT OPERATION

A. LOAD THE RP04/5/6 MULTIDRIVE EXERCISER PROGRAM INTO BOTH PROCESSORS.

B. SWITCH THE 'CONTROLLER SELECT' SWITCH TO 'A/B' ON EACH DRIVE WHICH IS TO BE TESTED AS A DUAL PORT DRIVE; CYCLE THE DRIVES UP.

C. START THE PROGRAM IN EACH PROCESSOR. RUN THE PROGRAM AS THROUGH EACH PROCESSOR WERE RUNNING INDEPENDENTLY OF THE OTHER.

D. PROGRAMMABLE (DUAL PORT) OPERATION IS INHIBITED WITH STARTING ADDRESS 200-SEE SE THEREFORE, USE STARTING ADDRESS 204 OR 220 FOR DUAL PORT TESTING.

4. CONTROLLING THE PROGRAM

THE FOLLOWING KEYBOARD CONVENTIONS ARE USED BY THE KEYBOARD ENTRY ROUTINES IN THE PROGRAM:

A. TO DELETE AN INCORRECT CHARACTER FROM AN ENTRY STRING, TYPE A 'RUBOUT' ('RO'). TYPING A 'RO' WILL DELETE SUCESSIVE CHARACTERS FROM THE INPUT.

B. TO DELETE AN ENTIRE LINE, TYPE A 'CONTROL U' ('^U').

C. AN ENTRY MUST BE TERMINATED BY EITHER A 'CARRIAGE RETURN' OR

A 'PERIOD'. THE 'PERIOD' TERMINATION IS RECOGNIZED BY THE PROGRAM AS A DEFAULT ENTRY REQUEST. WHEN A LINE IS TERMINATED BY A 'PERIOD' INSTEAD OF A 'CARRIAGE RETURN', THE PROGRAM WILL ACCEPT THE ENTERED VALUE AND WILL DEFAULT TO THE PRELOADED VALUES FOR ANY REMAINING ENTRIES.

D. IF A 'CONTROL C' IS TYPED DURING KEYBOARD ENTRY, THE PROGRAM WILL RETURN TO THE BEGINNING OF THE GROUP BEING ENTERED.

4.1 DATE & OPERATOR IDENTIFICATION

WHEN THE PROGRAM IS INITIALLY STARTED, IT WILL ASK FOR DATE AND OPERATOR I.D. ENTRIES. (THE REQUEST FOR THESE ENTRIES OCCURS ONLY WHEN THE PROGRAM IS FIRST STARTED AND WILL NOT APPEAR WHEN THE PROGRAM IS RESTARTED.) THESE ENTRIES ARE OPTIONAL AND MAY BE BYPASSED BY ENTERING A 'CARRIAGE RETURN' IN RESPONSE TO THE REQUEST. THE PROGRAM DOES NOT EDIT OR CHECK EITHER ENTRY. UP TO 8 CHARACTERS OF DATE INFORMATION AND UP TO 6 CHARACTERS OF OPERATOR IDENTIFICATION MAY BE ENTERED. BOTH THE DATE AND THE OPERATOR I.D. WILL BE TYPED WHEN THE 'SA' COMMAND IS PERFORMED (SEE SECTION 4.4.3).

4.2 PARAMETERS

WHEN THE PROGRAM IS STARTED FROM LOCATION 200, THE OPERATOR WILL BE ASKED TO ENTER PARAMETERS. THE FOLLOWING MESSAGE WILL BE DISPLAYED:

```
ENTER PARAMETERS:
```

THE OPERATOR MUST ENTER A 'Y' TERMINATED BY A CARRIAGE RETURN <CR> IF PARAMETER ENTRIES ARE TO BE MADE. ANY OTHER CHARACTER IS ACCEPTED AS A 'NO' ENTRY. THE PROGRAM WILL IDENTIFY THE PARAMETER BY THE NAME GIVEN BELOW, DISPLAY THE CURRENT VALUE OF THE PARAMETER AND WAIT FOR THE ENTRY. THE PROGRAM WILL TYPE 'INVALID ENTRY' IF THE ENTRY IS NOT CORRECT AND WAIT FOR A CORRECT ENTRY TO BE TYPED.

4.2.1 KEYBOARD ENTRY PARAMETERS

NAME	BASE	DEFAULT VALUE	VALUE RANGE	FUNCTION
MAXDL	10	(SEE NOTE)		CONTROLS THE MAXIMUM BUFFER SIZE USED FOR DATA TRANSFERS
PASCNT	10	1	1 - 999	NUMBER OF PASSES TO END OF TEST.
INTRVL	10	5	0 - 256	DETERMINES THE INTERVAL (IN MINUTES) BETWEEN AUTOMATIC PERFORMANCE SUMMARY TYPEOUTS ERRORS PRINTED OUT IF SW<07>=0
CMPLMT	10	3	0 - 'MAXDL'	THE NUMBER OF DATA COMPARSION

WCSEL	8	000000	0 OR 1	IF PARAMETER = 0, THE DATA TRANSFER WORD COUNT IS RANDOMLY SELECTED BETWEEN 4 (10) AND THE VALUE IN 'MAXDL'.
				IF PARAMETER = 1, THE DATA TRANSFER WORD COUNT WILL

```
                                                          BE THE VALUE IN 'MAXDL'
          ENDET     8      000001      0 OR 1             IF PARAMETER = 1, END OF PASS
                                                          DETERMINED BY THE 'WORDS READ'
                                                          COUNT.
                                                          IF PARAMETER = 0, END OF PASS
                                                          IS DETERMINED BY THE NUMBER
                                                          OF SEEKS.
          FORMAT    8      000001      0 OR 1             IF PARAMETER = 0; DO NOT
                                                          PERFORM WRITE HEADER & DATA
                                                          ORDERS; IF PARAMETER > 0,
                                                          PERFORM WRITE HEADER & DATA
                                                          ORDERS
          RATIO     8      000003      0 - 7              CONTROLS THE APPROXIMATE
                                                          RATIO OF READ TO WRITE
                                                          ORDERS.

                                                          VALUE   R/W RATIO

                                                            0        15/1
                                                            1        7/1
                                                            2        6/2
                                                            3        5/3
                                                            4        4/4
                                                            5        3/5
                                                            6        2/6
                                                            7        1/7
          AUTOCK    8      000001      0 OR 1             IF PARAMETER = 1, THE PROGRAM
                                                          PERFORM WRITE CHECKS AFTER
                                                          EACH WRITE COMMAND.
                                                          IF PARAMETER = 0, THE PROGRAM
                                                          WILL PERFORM WRITE CHECKS
                                                          RANDOMLY.
          NOTPRT    8      000001      0 OR 1             IF PARAMETER = 1, DO NOT PRINT
                                                          ERROR MESSAGES FOR DATA ERRORS
                                                          OCCURING AT LOCATIONS DEFINED
                                                          BY THE OPERATOR AS BAD PACK
                                                          LOCATION.
                                                          IF PARAMETER = 0, PRINT ERROR
                                                          MESSAGES ASSOCIATED WITH BAD
                                                          PACK LOCATIONS.



          NOTE:  THE PROGRAM WILL SELECT A MAXIMUM BUFFER SIZE WHICH
                  IS DETERMINED BY THE MEMORY AVAILABLE.  THE MAXIMUM BUFFER
                  SIZE ASSIGNED BY THE PROGRAM IS 5980 (10) WORDS.  THE
                  OPERATOR MAY SPECIFIY ANY OTHER MAXIMUM SIZE AS LONG AS THE
                  VALUE SPECIFIED IS AT LEAST 4 WORDS BUT NO LARGER THAN
                  THE INITIAL VALUE OF 'MAXDL' DETERMINED BY THE PROGRAM.


  4.2.2   PERIPHERAL ADDRESSES AND OTHER LOCATIONS OF INTEREST

          TO ALTER THESE LOCATIONS, THE OPERATOR MUST MAKE MANUAL ENTRIES
          BEFORE THE PROGRAM IS STARTED.  THE KEYBOARD ENTRY ROUTINE DOES
          NOT PROVIDE ACCESS TO THESE LOCATIONS.
```

```
        LOC     TAG     CONTENTS        FUNCTION
        ---     ---     --------        --------

        1144    $TKS    177560          TTY KEYBOARD STATUS REGISTER
        1146    $TKB    177562          TTY KEYBOARD BUFFER REGISTER
        1150    $TPS    177564          TTY PRINTER STATUS REGISTER
        1152    $TPB    177566          TTY PRINTER BUFFER REGISTER
        1174    $LKCSR  172540          ADDRESS OF KW11-P STATUS REGISTER
        1176    $LKCSB  172542          ADDRESS OF KW11-P COUNTER BUFFER
        1200    $LPVEC     104          KW11-P VECTOR ADDRESS
        1202    $LKS    177546          ADDRESS OF KW11-L STATUS REGISTER
        1204    $LLVEC     100          KW11-L VECTOR ADDRESS
        1212    HZ          74          74 (60 DECIMAL) IF SYSTEM IS 60 HZ;
                                        62 (50 DECIMAL) IF SYSTEM IS 50 HZ.

        THE RH11-RH70 ADDRESS AND VECTOR MAY BE CHANGED WHEN THE PROGRAM
        IS STARTED FROM LOCATION 204(8) OR IF THE PROGRAM DOES NOT RECEIVE
        A RESPONSE WHEN IT ACCESSES THE DEFAULT RH11-RH70 ADDRESS.

4.2.3   PARAMETERS FOR THE FIRST OPERATION

        THE FOLLOWING PARAMETERS ARE USED FOR THE INITIAL OPERATION (IN
        ADDITION TO THE 'MINIMUM' ADDRESS VALUES).

                        INITIAL   VALUE
        LOC     TAG      VALUE    RANGE                   FUNCTION
        ---     ---     -------   -----                   --------

        1412    BEGPAT     10      1 - 15         THE CODE FOR THE STARTING
                                                  PATTERN.  (IF A WRITE ORDER
                                                  OR A WRITE CHECK ORDER IS
                                                  SPECIFIED IN 'BEGCOD')
        1414    BEGCOD      5      0 - 5          THE INITIAL COMMAND FOR EACH DRIVE
                                                  EXERCISED.
                                                  0 = WRITE CHECK DATA
                                                  1 = WRITE CHECK HEADER & DATA
                                                  2 = WRITE DATA
                                                  3 = WRITE HEADER & DATA
                                                  4 = READ DATA
                                                  5 = READ HEADER & DATA
        1416    BEGSIZ    404      4 - MAXDL      THE BUFFER SIZE FOR THE FIRST
                                                  DATA TRANSFER OPERATION.

4.3     SWITCH REGISTER SETTINGS

        SW <15> = 1     HALT ON ERROR
        SW <13> = 1     INHIBIT ERROR TYPEOUT
        SW <10> = 1     RING THE TELETYPE BELL IF ERROR
        SW <7>  = 1     DISPLAY ALL DATA COMPARE ERRORS
        SW <6>  = 1     DO NOT ALTER THE CURRENT OPERATION PARAMETERS
        SW <5>  = 1     PARTIAL REGISTER DISPLAY IF ERROR; DO NOT DISPLAY
                        ECC CORRECTION RESULTS
        SW <4>  = 1     INHIBIT MAXIMUM ERROR COUNT CHECK; DO NOT DEASSIGN
                        DRIVES WHEN NORMAL END OF TEST REACHED.
        SW <3>  = 1     DISPLAY THE SECTOR IN ERROR (BEFORE RETRY ATTEMPTS)
                        IF 'DCK', 'DTE', OR 'WCF' ERRORS OR AFTER THE 28TH
                        RETRY IF UNCORRECTABLE 'DCK' ERROR.
                        IF DATA COMPARE ERRORS & SW<7> SET, DISPLAY REST
                        OF BUFFER
        SW <2>  = 1     INHIBIT SUBSYSTEM STATUS TYPEOUT DURING STARTUP.
                        INHIBIT PERFORMANCE SUMMARY TYPEOUTS.
        SW <1>  = 1     INHIBIT DATA COMPARSION AFTER READ ORDERS
        SW <0>  = 1     READ ONLY MODE

        IF THE PROGRAM IS BEING RUN ON A SWITCHLESS PROCESSOR (I.E. AN 11/34)
        THE PROGRAM WILL DETERMINE THAT THE HARDWARE SWITCH REGISTER IS
        NOT PRESENT AND WILL USE A 'SOFTWARE' SWITCH REGISTER.  THE
        'SOFTWARE' SWITCH REGISTER IS LOCATED AT LOCATION 176 (8).  THE
        SETTINGS OF THE 'SOFTWARE' SWITCHES ARE CONTROLLED THROUGH A KEYBOARD
        ROUTINE WHICH IS CALLED BY TYPING A 'CONTROL G'.  THE PROGRAM WILL
        RECOGNIZE THE 'CONTROL G' AT ANY TIME EXCEPT WHEN THE PROGRAM IS
        IN KEYBOARD ENTRY MODE, OR IS AT A HIGHER PRIORITY PROCESSING AN
        RP04/5/6 INTERRUPT.  THE 'SOFTWARE' SWITCH VALUES ARE ENTERED
        AS AN OCTAL NUMBER IN RESPONSE TO THE PROMPT FROM THE SWITCH
        ENTRY ROUTINE:

                'SWR = NNNNNN    NEW ='

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

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

4.4     KEYBOARD COMMANDS

        THROUGH THE KEYBOARD COMMANDS, THE OPERATOR MAY ASSIGN DRIVES
        FOR TEST ('T' COMMAND), WRITE AND CHECK DATA PACKS ('W' COMMAND),
        PERFORM A SEQUENTIAL READ OF A PACK ('R' COMMAND), REQUEST A DRIVE
        PERFORMANCE SUMMARY ('S' COMMAND), OR DEASSIGN A DRIVE WHICH IS
        BEING TESTED, READING, OR WRITING ('D' COMMAND).

        AFTER THE PROGRAM HAS BEEN INITIALIZED, THE FOLLOWING MESSAGE
        WILL BE TYPED:

                'PROGRAM INITIALIZATION COMPLETE
                'TYPE A CONTROL C TO ENTER COMMANDS'

        KEYBOARD ENTRIES WILL NOT BE RECOGNIZED UNTIL THE OPERATOR TYPES
        A 'CONTROL C'.  WHEN THE PROGRAM SEES A 'CONTROL C' ENTRY, IT WILL
        SUSPEND THE SCHEDULING OF FURTHER DEVICE OPERATIONS AND WAIT UNTIL
        ALL OUTSTANDING ORDERS HAVE TERMINATED.  THE PROGRAM WILL ENTER
        COMMAND ENTRY MODE AND TYPE THE FOLLOWING PROMPTING MESSAGE:

                'HH:MM:SS
                'ENTER COMMANDS:'

        THE PROGRAM WILL THEN ACCEPT ANY OF THE VALID COMMANDS.  AT THE
        COMPLETION OF A COMMAND, THE PROGRAM WILL EXIT COMMAND MODE; THE
```

OPERATOR MUST TYPE ANOTHER 'CONTROL C' TO RETURN THE PROGRAM TO COMMAND MODE. IF THE COMMAND ENTERED SPECIFIED AN 'A' DRIVE NUMBER, THE PROGRAM WILL REMAIN IN COMMAND MODE UNTIL ALL AVAILABLE DRIVES HAVE BEEN PROCESSED.

THE 'T', 'W', AND 'R' COMMANDS REQUIRE ADDRESS LIMITS, DRIVE I.D. AND BAD LOCATION ADDRESS ENTRIES FOR THE DRIVE BEING REFERENCED.

THE PROGRAM WILL FIRST ASK FOR ADDRESS LIMITS WITH THE FOLLOWING TYPEOUT:

'ENTER ADDRESS LIMITS FOR DRV #N / RP04' (OR RP05 OR RP06)

THE PROGRAM WILL REQUEST VALUES FOR THE FOLLOWING ADDRESS LIMIT PARAMETERS.

NAME	BASE	DEFAULT VALUE	VALUE RANGE	FUNCTION
MINCYL	10		(SEE NOTE)	THE MINIMUM CYLINDER ADDRESS
MAXCYL	10		(SEE NOTE)	THE MAXIMUM CYLINDER ADDRESS
MINTRK	10	0	0 - 18	THE MINIMUM TRACK ADDRESS
MAXTRK	10	18	0 - 18	THE MAXIMUM TRACK ADDRESS
MINSEC	10	0	0 - 21	THE MINIMUM SECTOR ADDRESS
MAXSEC	10	21	0 - 21	THE MAXIMUM SECTOR ADDRESS

NOTE: 1. THE ADDRESS LIMITS ARE IN DECIMAL

2. THE MAXIMUM VALUES OF 'MINCYL' AND 'MAXCYL' WILL BE EITHER 410 (10) OR 814 (10) DEPENDING ON THE TYPE OF DRIVE.

3. THE MINIMUM CYLINDER, TRACK, OR SECTOR ADDRESS MAY BE SPECIFIED AS BEING LARGER THAN THE MAXIMUM ADDRESS. WHEN THESE VALUES ARE INVERTED, THE PROGRAM WILL SELECT ADDRESSES BETWEEN THE 'MIN' ADDRESS AND THE UPPER PHYSICAL LIMIT FOR THAT ADDRESS AND BETWEEN '0' AND THE VALUE IN 'MAX'.

EACH COMMAND, EXCEPT THE 'S' AND THE 'D' COMMANDS, WILL ASK FOR A DRIVE IDENTIFICATION ENTRY. THE DRIVE IDENTIFICATION ENTRY REQUEST IS MADE BY THE FOLLOWING MESSAGE:

'ENTER I.D. FOR DRV #N:'

THE OPERATOR MAY ENTER AN I.D. NUMBER FOR THE DRIVE OF UP TO 6 CHARACTERS IN LENGTH. THIS I.D. WILL BE DISPLAYED, ALONG WITH THE DATE AND OPERATOR I.D. ENTRIES (SEE SECTION 4.1), WHEN THE 'SA' COMMAND IS EXECUTED. THE OPERATOR MAY ENTER ANY CHARACTER STRING, TERMINATED BY A 'CARRIAGE RETURN', OR A 'PERIOD' ONLY (NULL ENTRY) IN RESPONSE TO THE I.D. REQUEST.

THE PROGRAM WILL THEN ASK FOR THE ADDRESSES OF KNOWN BAD SPOTS ON THE DISK PACK USED, UP TO 16 ADDRESSES MAY BE ENTERED:

'BAD TRK/SEC ADRS FOR DRV #N ?'

```
        THE OPERATOR MAY DECLARE UP TO 16 BAD LOCATIONS ON THE PACK BEING
        USED.  THE LOCATION MAY BE AN ENTIRE CYLINDER, A BAD TRACK, OR A  SINGLE
        SECTOR.  THE FORMATS USED ARE AS FOLLOW:

        FORMAT 1:       C,T,S<CR>

                A.  THE PROGRAM WILL INHIBIT DATA ERROR MESSAGES OR WILL
                    IDENTIFY DATA ERRORS WHICH OCCUR AT THE SPECIFIED ADDRESS,
                    DEPENDING ON THE VALUE OF PARAMETER 'NOTPRT'.
                B.  LEADING ZEROS ARE NOT REQUIRED.

        FORMAT 2:       C,T<CR>

                A.  WHEN THIS FORMAT IS USED, THE ENTIRE TRACK WILL BE
                    CONSIDERED BAD; DATA ERRORS WILL BE HANDLED AS IN
                    'FORMAT 1', ABOVE.
                B.  LEADING ZEROS ARE NOT REQUIRED.

        FORMAT 3:       C<CR>

                A.  WHEN THIS FORMAT IS USED, THE ENTIRE CYLINDER WILL
                    BE CONSIDERED BAD; DATA ERRORS WILL BE HANDLED AS IN
                    'FORMAT 1' ABOVE.
                B.  LEADING ZEROS ARE NOT REQUIRED.

        NOTE:  CYLINDER, TRACK, AND SECTOR ENTRIES ARE IN DECIMAL.

        THE OPERATOR MAY TERMINATE THE BAD ADDRESS ENTRY BY ENTERING A
        'PERIOD' IN RESPONSE TO THE ENTRY REQUEST OR BY TERMINATING AN ENTRY
        WITH A 'PERIOD' INSTEAD OF A 'CARRIAGE RETURN'.

4.4.1   'T' COMMAND

        USED TO ASSIGN A DRIVE(S) FOR TEST.  THIS COMMAND IS REQUIRED
        PERFORM THE TEST OF THE DRIVE(S).  THE OTHER COMMANDS ARE CONVIENCE
        COMMANDS OR SUPPORT COMMANDS.

        FORMAT: TN<CR>

        N = DRIVE NUMBER.  MAY BE 0 TO 7 OR 'A'.  ENTRY MUST BE
            TERMINIATED BY A CARRIAGE RETURN <CR>.

        EXAMPLE:  T0<CR> - ASSIGN DRIVE 0 FOR TEST
                  TA<CR> - ASSIGN ALL AVAILABLE DRIVES FOR TEST

        NOTE:  DRIVE OPERATION BEGINS IMMEDIATELY
               AFTER COMMAND IS ENTERED.


4.4.2   'D' COMMAND

        USED TO DEASSIGN A DRIVE(S) BEING EXERCISED.

        FORMAT:  DN<CR>

        N = DRIVE NUMBER.  MAY BE 0 TO 7 OR 'A'.  ENTRY MUST BE
            TERMINIATED BY A CARRIAGE RETURN <CR>.
```

```
                EXAMPLE:  D0<CR> - DEASSIGN DRIVE 0
                          DA<CR> - DEASSIGN ALL DRIVES BEING
                                   TESTED.

                NOTES:  1. IF THE 'D' COMMAND REFERENCES A
                           DRIVE NOT ASSIGNED THE PROGRAM
                           WILL TYPEOUT '?DRIVE NOT ASSIGNED'

                        2. THE DRIVES WILL BE DEASSIGNED AS THEIR
                           OPERATIONS COMPLETE.

                        3. IF 'DA' IS USED, ONLY DRIVES BEING
                           TESTED WILL BE DEASSIGNED - THE
                           ERROR MESSAGE IN (1) ABOVE WILL NOT
                           BE DISPLAYED.

        4.4.3   'S' COMMAND

                USED TO REQUEST A PERFORMANCE SUMMARY TYPEOUT FOR THE REFERENCED
                DRIVE(S).

                FORMAT:  SN<CR>

                N = DRIVE NUMBER.  MAY BE 0 TO 7 OR 'A'.  ENTRY MUST BE
                    TERMINIATED BY A CARRIAGE RETURN <CR>.

                EXAMPLE:  S0<CR> - TYPEOUT PERFORMANCE SUMMARY FOR DRIVE 0
                          SA<CR> - TYPEOUT PERFORMANCE SUMMARY
                                   FOR ALL DRIVES BEING TESTED.

                NOTES:  1. IF 'SA' IS USED ONLY DRIVES BEING TESTED WILL
                           BE DISPLAYED.

                        2.  IF PARAMETER 'INTRVL' IS NOT ZERO, THE PROGRAM
                            WILL AUTOMATICALLY DISPLAY A PERFORMANCE SUMMARY
                            FOR EACH DRIVE BEING TESTED AT A RATE DETERMINED BY
                            'INTRVL'.

                        3.  IF THE 'SA' COMMAND IS USED, THE PROGRAM WILL TYPEOUT
                            THE OPERATOR ENTERED DATE, OPERATOR I.D., AND THE DRIVE
                            I.D. FOR EACH DRIVE BEING TESTED.  THE DATE AND OPERATOR
                            I.D. WILL NOT BE TYPED OUT IF NO DRIVES ARE BEING TESTED.

        4.4.4   'W' COMMAND

                USED TO WRITE A DATA PACK WITH DATA ACCEPTABLE TO THE RP04/5/6 MULTI-
                DRIVE EXERCISER PROGRAM.

                FORMAT: WN<CR>

                N = DRIVE NUMBER.  MAY BE 0 TO 7 OR 'A'.  ENTRY MUST BE
                    TERMINATED BY A CARRIAGE RETURN <CR>.

                EXAMPLE:  WA<CR> - WRITE DATA PACKS ON ALL AVAILABLE DRIVES.
                          W0<CR> - GENERATE A DATA PACK ON DRIVE 0.
                NOTES:  1.  DATA PACKS GENERATED BY THE RP04/5/6 FORMATTER PROGRAM
                            (MD-11-CZRJB) OR BY THE RP04/5/6 MECHANICAL & READ/WRITE
                            PROGRAM (MD-11-CZRJA), TEST 20, ARE ACCEPTABLE.  (PACKS
                            WRITTEN BY TESTS 16, 17 OR 21 OF 'CZRJA' CANNOT BE USED
                            AND MUST BE REWRITTEN.)

                        2.  THE 'W' COMMAND SHOULD NOT BE USED UNLESS 'MAXDL'
                            PARAMETER IS APPROXIMATELY 1 TRACK IN SIZE - 5000 (10).
                            IF THE BUFFER SIZE IS MUCH LESS THAN 1 TRACK, THE
                            TIME REQUIRED TO WRITE A DATA PACK IS TOO GREAT TO
                            BE PRACTICAL.

                        3.  THE 'W' COMMAND PERFORMS A SEQUENTIAL WRITE OF THE
                            PACK USING A 'WRITE DATA' COMMAND.  THE DATA PATTERN
                            USED FOR EACH WRITE IS SELECTED RANDOMLY.  HOWEVER,
                            THE OPERATION OF THE COMMAND IS SEQUENTIAL, BEGINNING
                            AT 'MINCYL', 'MINTRK' AND CONTINUING TO 'MAXCYL',
                            'MAXTRK'.

                        4.  THE 'W' COMMAND DOES NOT WRITE HEADERS AND ASSUMES
                            THAT THE FORMAT OF THE PACK IS GOOD.

                        5.  THE 'W' COMMAND CANNOT BE STARTED IF SWITCH 0 (READ
                            ONLY MODE) IS SET.  IF SWITCH 0 SET SET DURING THE
                            OPERATION OF THE 'W' COMMAND, THE DRIVE PERFORMING THE
                            'W' COMMAND WILL IGNORE THE SWITCH.

                        6.  THE DATA WRITTEN IS VERIFIED BY A 'WRITE CHECK DATA'
                            COMMAND.


        4.4.5   'R' COMMAND

                USED TO PERFORM A SEQUENTIAL READ OF THE PACK.

                FORMAT: RN<CR>

                N = DRIVE NUMBER.  MAY BE 0 TO 7 OR 'A'.  ENTRY MUST BE
                    TERMINATED BY A CARRIAGE RETURN <CR>.

                EXAMPLE:  RA<CR> - READ THE PACKS ON ALL OF THE ONLINE DRIVES.
                          R0<CR> - READ THE PACK ON DRIVE 0.

                NOTES:  1.  THE PROGRAM WILL PERFORMA NORMAL CHECK OF ALL DATA
                            READ.  HOWEVER, ALL OPERATIONS WILL BE SEQUENTIAL.

                        2.  THE PROGRAM WILL READ THE PACK STARTING AT THE ADDRESS
                            SPECIFIED BY 'MINCYL', 'MINTRK' TO THE ADDRESS SPECIFIED
                            BY 'MAXCYL', 'MAXTRK'.  THE READ WILL BE SEQUENTIAL.


        4.4.6   GENERAL COMMAND INFORMATION

                A.  WHEN A COMMAND IS ENTERED, THE PROGRAM WILL TYPE OUT THE TIME

                B.  IF THE COMMAND ENTERED IS NOT VALID, THE PROGRAM WILL TYPE
```

'INVALID COMMAND'.

C. DRIVES ASSIGNED (WITH THE 'T' COMMAND) OR DEASSIGNED (WITH THE 'D' COMMAND) MAY BE ENTERED IN ANY SEQUENCE.

D. THE ERROR RESPONSES FROM THE PROGRAM ARE AS FOLLOWS

RESPONSE	COMMAND(S)
?UNIT N OFFLINE	T, W, R
?UNIT N NOT ASSIGNED	D, S
?UNIT N ALREADY ASSIGNED	T, W, R
?UNIT N NOT PRESENT	T, W, R
?UNIT N UNSAFE	T, W, R
?UNIT N NOT AN RP04/5/6	T, W, R

5. PERFORMANCE SUMMARY TYPEOUT

5.1 THE PROGRAM WILL DISPLAY A PERFORMANCE SUMMARY FOR THE DRIVES BEING EXERCISED. THIS SUMMARY WILL BE DISPLAYED AUTOMATICALLY IF THE PARAMETER 'INTRVL' IS NOT ZERO OR CAN BE DISPLAYED ON REQUEST BY THE OPERATOR THROUGH THE USE OF THE 'S' COMMAND. THE PERFORMANCE SUMMARY TYPEOUT CONTAINS THE FOLLOWING FIELDS:

'DRV'	THE DRIVE NUMBER
'PASS'	THE PRESENT PASS COUNT FOR THE DRIVE
'ORDERS'	THE NUMBER OF ORDERS PERFORMED BY THE DRIVE
'SEEKS'	THE NUMBER OF SEEK OPERATIONS THE DRIVE PERFORMED
'WRDS XFER'	THE TOTAL NUMBER OF WORDS WRITTEN AND READ BY THE DRIVE
'WRDS READ'	THE TOTAL NUMBER OF WORDS READ BY THE DRIVE
'SOFT'	THE NUMBER OF SOFT DATA ERRORS
'HARD'	THE NUMBER OF HARD DATA ERRORS
'SKI'	THE NUMBER OF 'SKI' OR 'OCYL' ERRORS
'MISP'	THE NUMBER OF POSITIONING ERRORS
'OTHER'	THE TOTAL ERRORS OF OTHER TYPES

NOTE: ERRORS EM1, EM2, EM3, EM4, EM5, & EM10 ARE NOT INCLUDED IN THE 'OTHER' ERROR TOTAL.

5.2 SOFT/HARD ERROR DEFINITIONS

5.2.1 HARD ERRORS

A. A 'DTE' (DRIVE TIMING ERROR) OR A 'DCK' (DATA CHECK ERROR) WHICH OCCURS DURING A READ DATA OR A READ HEADER & DATA OPERATION AND IS NOT CORRECTED OR BECOMES CORRECTABLE AFTER THE PROGRAM HAS PERFORMED THE COMPLETE RETRY SEQUENCE ON THE BAD SECTOR. THE RETRY SEQUENCE IS 16 RE-READS AT TRACK CENTER AND 2 ATTEMPTS AT EACH OF THE FOLLOWING OFFSETS:

RP04/5	RP06

+400 MICRO-INCHES	+200 MICRO-INCHES
-400 MICRO-INCHES	-200 MICRO-INCHES
+800 MICRO-INCHES	+400 MICRO-INCHES
-800 MICRO-INCHES	-400 MICRO-INCHES
+1200 MICRO-INCHES	+800 MICRO-INCHES
-1200 MICRO-INCHES	-800 MICRO-INCHES

5.2.2 SOFT ERRORS

A. ECC CORRECTABLE 'DCK' ERRORS.
B. 'DCK' & 'ECH' ERRORS WHICH BECOME ECC CORRECTABLE DURING RETRY OR WHICH ARE READ CORRECTLY DURING RETRY.
C. HEADER READ ERRORS - READ DATA, READ HEADER & DATA, OR WRITE DATA ORDERS.
D. 'DTE' ERRORS WHICH ARE CORRECTED OR WHICH BECOME ECC CORRECTABLE 'DCK' ERROR DURING THE RETRY SEQUENCE.

6. DATA CHECKING & ERROR RECOVERY

6.1 DATA COMPARISON

DATA COMPARISON OCCURS AFTER EACH 'RDDAT' (READ DATA) OR 'RDHD' (READ HEADER AND DATA) OPERATION UNDER THE FOLLOWING CONDITIONS:

A. THE ORDER TERMINATED WITH NO ERROR.

B. THE OPERATION TERMINATED WITH 'DCK' SET AND THE ERROR IS ECC CORRECTABLE OR THE SECTOR IN ERROR IS READ CORRECTLY AFTER RETRY ATTEMPTS.

6.2 VERIFICATION OF DATA WRITTEN

AFTER EACH WRITE OPERATION, THE DATA WRITTEN IS CHECKED WITH THE APPROPRIATE WRITE CHECK COMMAND - IF PARAMETER 'AUTOCK' IS 1. (IF 'AUTOCK' IS 0, WRITE CHECKS WILL BE PERFORMED RANDOMLY.)

6.3 SECTOR REFORMATTING

THE PROGRAM WILL REFORMAT AN UNCORRECTABLE ERROR SECTOR IN THE FOLLOWING CASES (PARAMETER 'FORMAT' MUST BE SET AND SW<00> = 0). THIS PREVENTS THE SAME ERROR FROM BEING CONTINUOUSLY REPORTED.

A. DATA CHECK ERRORS - EM21
B. HEADER READ ERRORS - EM20, EM24, EM25, EM26, EM27
C. DRIVE TIMING ERRORS - EM31
D. OPERATION INCOMPLETE ERRORS - EM32
E. WRITE CHECK ERRORS - EM22, EM23

6.4 BAD TRACK/SECTOR FLAGGING

SINCE THE RP04 SUBSYSTEM DOES NOT HAVE AN AUTOMATIC BAD TRACK HANDLING CAPABILTIY, THE MULTIDRIVE EXERCISER ALLOWS THE OPERATOR TO IDENTIFY UP TO 8 BAD TRACK/SECTOR LOCATIONS WHEN THE DRIVE IS ASSIGNED FOR TEST. (SEE SECTION 4.1 FOR ADDITIONAL INFORMATION.)

IF ONE OF THE FOLLOWING ERRORS OCCURS AT A LOCATION IDENTIFIED BY THE OPERATOR, THE PROGRAM WILL INHIBIT THE ERROR REPORT FOR THAT ERROR.

DATA CHECK ERRORS ('DCK')
WRITE CHECK ERRORS ('WCE')
OPERATION INCOMPLETE ERRORS ('OPI')
DRIVE TIMING ERRORS ('DTE')
HEADER READ ERRORS ('FER' & 'HCRC', 'HCE' & 'HCRC', 'HCRC')

OPTIONALLY, THE OPERATOR MAY REQUEST AN ERROR REPORT FOR FLAGGED AREAS. (PARAMETER 'NOTPRT' MUST BE SET TO 0 AT STARTUP.) THE PROGRAM WILL IDENTIFY ERROR MESSAGES ASSOCIATED WITH THE PACK; THESE ERRORS WILL NOT BE ADDED TO THE ERROR TOTALS MAINTAINED BY THE PROGRAM.

7. ERROR MESSAGES

DRIVE ERRORS ARE REPORTED ON THE TELETYPE OR (IF AVAILABLE) A LINE PRINTER. ALL ERROR CONDITIONS ARE REPORTED IN ERROR MESSAGES; THE PROGRAM CONTAINS NO CODED ERROR HALTS. IF THE PROGRAM HALTS (ASSUMING, OF COURSE, THAT SW<15> IS NOT SET), AN UNRECOVERABLE PROGRAM CONDITION HAS OCCURRED OR A CENTRAL PROCESSOR FAILURE HAS OCCURRED.

ERROR MESSAGES ARE MADE UP OF SEVERAL LINES. EACH TYPE OF ERROR HAS SEVERAL OPTIONAL LINES WHICH MAY APPEAR WITH IT. ALL OF THE POSSIBLE ERROR MESSAGE LINES WHICH MAY APPEAR ARE GIVEN IN THE SECTION DESCRIBING THE PARTICULAR ERROR HEADER.

7.1 ERROR DESCRIPTION LINES

MESSAGES EM1, EM2, EM3, EM4, EM5, EM10, EM11, & EM12 ARE ALWAYS DISPLAYED ON THE TTY. THE OTHER MESSAGES ARE DISPLAYED ON EITHER THE LINE PRINTER (IF AVAILABLE) OR THE TTY.

(THE MESSAGE TAGS ARE GIVEN FOR REFERENCE.)

```
MESSAGE
TAG             TEXT
---             ----

EM1     RH11 INTERRUPT OCCURRED (RPAS=0)

        THE RH11 INTERRUPTED AND THE ATTENTION SUMMARY REGISTER
        (RPAS) WAS CLEARED.

EM2     UNEXPECTED ATTENTION OCCURRED

        THE INDICATED DRIVE INTERRUPTED BUT THE DRIVE WAS NOT
        PERFORMING AN OPERATION.
```

```
EM3     MASSBUS PARITY ERROR (MCPE=1)

        THE RH11 DETECTED A CONTROL BUS PARITY ERROR WHEN READING
        THE INDICATED REGISTER FROM THE INDICATED DRIVE.

EM4     MASSBUS PARITY ERROR (PAR=1)

        THE INDICATED RP04 DETECTED A CONTROL BUS PARITY ERROR
        WHEN THE RH11 LOADED THE SPECIFIED REGISTER.

EM5     ADDRESS PLUG CHANGE BIT SET

        THE 'OPE' BIT WAS SET WHEN THE INDICATED DRIVE INTERRUPTED.

EM6     RH11 DIDN'T RESPOND TO ADDRESSING

        WHEN THE PROGRAM ADDRESSED THE RH11, NO RESPONSE WAS RECEIVED
        FROM THE INDICATED ADDRESS.

EM10    UNCORRECTABLE MASSBUS PARITY ERROR

        THE PROGRAM HAS TRIED 3 TIMES TO READ OR WRITE THE INDICATED
        REGISTER.

EM11    FATAL MASSBUS PARITY ERROR

        A CONTROL BUS PARITY ERROR OCCURRED WHEN THE RH11 ATTEMPTED
        TO PROCESS A PREVIOUS, DIFFERENT PARITY ERROR.

EM12    PERSISTENT DEVICE UNSAFE

        THE DRIVE BECAME UNSAFE; DRIVE CLEAR TO THE DRIVE DID
        NOT CLEAR THE UNSAFE CONDITION.  THE PROGRAM WILL
        AUTOMATICALLY DEASSIGN THE DRIVE.  THE DRIVE CANNOT
        BE EXERCISED UNTIL THE UNSAFE CONDITION HAS BEEN
        CLEARED.

EM13    OPERATION NOT COMPLETED WITHIN TIME LIMIT

        THE DRIVE DID NOT COMPLETE THE OPERATION WITHIN 1 SECOND
        AFTER THE OPERATION WAS INITIATED.

EM14    UNIT WENT OFFLINE

        THE DRIVE WENT OFFLINE DURING THE INDICATED OPERATION.
        (THE 'MOL' BIT BECAME ZERO.)  THE PROGRAM WILL AUTOMATICALLY
        DEASSIGN THE DRIVE.  THE OPERATOR MUST REASSIGN THE DRIVE
        WITH THE 'T' COMMAND TO RE-INITIATE TESTING.

EM15    NO RESPONSE TO PORT REQUEST

        THE PROGRAM IS TESTING A DUAL PORT DRIVE WHICH HAS NOT SWITCHED
        TO THE REQUESTING PORT WITHIN 10 SECONDS AFTER PORT REQUEST
        TO THE DRIVE FROM THE REPORTING PORT.

EM20    HEADER CRC ERROR
```

```
        A HEADER CRC ERROR WAS DETECTED AT THE INDICATED DISK ADDRESS.
        THE CONTENTS OF THE HEADER ARE DISPLAYED.  THE OPERATION WILL
        BE RETRIED 3 TIMES.

EM21    DATA CHECK ('DCK') ERROR

        A DATA CHECK ERROR WAS DETECTED AT THE INDICATED SECTOR.
        THE FULL RETRY SEQUENCE (INCLUDING OFFSET) WILL BE INITIATED
        FOR THE SECTOR IN ERROR IF THE ECC HARD ERROR ('ECH) BIT
        IS SET.

EM22    WRITE CHECK ERROR - DATA CHECK ('DCK') SET

        A WRITE CHECK ERROR OCCURRED AND THE DATA CHECK ('DCK') BIT
        WAS SET.  IF 'ECH' IS NOT SET, THE OPERATION WILL BE RETRIED
        UP TO 3 TIMES; IF THE 'ECH' BIT IS SET, THE OPERATION WILL
        BE RETRIED UP TO 16 TIMES.

EM23    WRITE CHECK ERROR - DATA CHECK ('DCK') NOT SET

        A WRITE CHECK ERROR OCCURRED AND 'DCK' WAS NOT SET.  THE
        WORDS WHICH CAUSED THE ERROR ARE DISPLAYED IN THE ERROR
        MESSAGE.  THE OPERATION WILL BE RETRIED 3 TIMES.

EM24    HEADER READ ERROR - 'FMT' BIT DROPPED

        A WRITE DATA, WRITE CHECK DATA, OR A READ DATA WAS BEING
        PERFORMED AND A 'FMT' ERROR OCCURRED.  THE PROGRAM RE-READ THE
        HEADER OF THE ERROR SECTOR AND THE 'HCRC' BIT WAS SET.  THE
        CONTENTS OF THE HEADER ARE DISPLAYED.  THE OPERATION WILL
        BE RETRIED 3 TIMES.

EM25    HEADER READ ERROR - HEADER COMPARE ('HCE') ERROR

        SIMILAR TO EM24, EXCEPT THAT THE 'HCE' ERROR BIT WAS
        SET INITIALLY.  THE OPERATION WILL BE RETRIED 3 TIMES.

EM26    FORMAT ERROR ('FER')

        FORMAT ERROR OCCURRED.  WHEN THE HEADER WAS RE-READ, THE
        'HCRC' BIT WAS NOT SET.  THE CONTENTS OF THE HEADER ARE
        DISPLAYED.  THE OPERATION WILL BE RETRIED 3 TIMES.

EM27    HEADER COMPARE ('HCE') ERROR

        SIMILAR TO EM26 EXCEPT THAT THE 'HCE' BIT WAS SET INITIALLY.
        THE OPERATION WILL BE RETRIED 3 TIMES.

EM30    MISCELLANEOUS DRIVE ERROR

        THIS MESSAGE IS GIVEN FOR THE FOLLOWING ERROR BITS:
        'IXE', 'AOE', 'RMR', 'ILF', OR 'ILR'

EM31    OPERATION INCOMPLETE ('OPI') ERROR

        AN OPERATION INCOMPLETE ERROR OCCURRED AT THE INDICATED
        SECTOR.
```

```

EM32    DRIVE TIMING ('DTE') ERROR

        DRIVE TIMING ERROR OCCURRED ON THE INDICATED SECTOR.  THE
        OPERATION WILL BE RETRIED 3 TIMES.

EM33    PARITY ('PAR') ERROR AFTER OPERATION STARTED

        THE 'PAR' BIT WAS SET WHEN THE OPERATION WAS COMPLETED.  THE
        OPERATION WILL BE RETRIED 3 TIMES.

EM34    WRITE CLOCK FAILURE ('WCF')

        A WRITE CLOCK FAILURE OCCURRED DURING THE OPERATION.  THE
        OPERATION WILL BE RETRIED 3 TIMES.

EM35    INVALID ADDRESS ('IAE') ERROR

        AN INVALID ADDRESS ERROR OCCURRED DURING THE OPERATION.

EM36    WRITE LOCK ('WLE') ERROR

        A WRITE OPERATION WAS ATTEMPTED BUT THE DRIVE WAS WRITE
        LOCKED.

EM40    RH11 OR UNIBUS TRANSFER ERROR

        'TRE' IS SET IN THE RH11 CONTROL REGISTER AND NO DRIVE
        ERROR HAS OCCURRED.  THE OPERATION WILL BE RETRIED 3
        TIMES IF THE ERROR WAS CAUSED BY 'DLT', 'UPE', 'MXF',
        OR 'MDPE'.

EM41    BUS ADDRESS OR WORD COUNT INCORRECT

        NO DRIVE ERROR OCCURRED BUT EITHER THE BUS ADDRESS INDICATES
        THAT AN INCORRECT NUMBER OF WORDS WERE TRANSFERED OR THE
        WORD COUNT REGISTER IS NOT ZERO.

EM42    DATA COMPARE ERRORS - NO DRIVE ERROR DETECTED

        NO SUBSYSTEM ERROR WAS SIGNALED; HOWEVER, THE DATA DOES NOT
        COMPARE.

EM43    CAN'T MATCH DATA READ WITH A PATTERN

        THE DATA IN THE BUFFER DOES NOT MATCH ANY OF THE STANDARD
        PATTERNS.

EM44    ERROR BIT(S) SET, BUT NO ERROR SIGNALED BY THE RH11

        THE OPERATION COMPLETED NORMALLY; HOWEVER, THE PROGRAM
        FOUND EITHER ERROR BITS IN THE RP04 SET OR ERROR BITS IN
        THE RH11 SET.

EM45    ECC LOGIC FAILURE

        THE CONTENTS OF EITHER THE ECC POSITION REGISTER (RPEC1)
```

```
                        OR THE CONTENTS OF ECC PATTERN REGISTER (RPEC2) ARE NOT
                        VALID.  THE POSITION REIGSTER IS EITHER'0' OR > 10041 (8)
                        OR THE PATTERN REGISTER CONTAINS ZEROS.

                EM46    BUS ADDRESS OR WORD COUNT NOT CONSISTENT

                        THE PROGRAM WAS PROCESSING AN ERROR AND FOUND THAT THE
                        NUMBER OF WORDS TRANSFERED AS INDICATED BY THE BUS ADDRESS
                        REGISTER DOES NOT AGREE WITH THE TRANSFER COUNT FROM THE
                        WORD COUNT REGISTER.

                EM50    SEEK INCOMPLETE OR OFF CYLINDER ERROR

                        THE DRIVE SIGNALED EITHER 'SKI' OR 'OCYL' ERROR BITS.

                EM51    PROGRAM DETECTED POSITIONING ERROR

                        A HEADER COMPARE ERROR OCCURRED ('HCE'); HOWEVER, WHEN THE
                        PROGRAM EXAMINED THE HEADER OF THE SECTOR IN ERROR,  IT
                        FOUND THAT THE CYLINDER FIELD DID NOT AGREE WITH THE CONTENTS
                        OF 'RPCC' OF THE DRIVE.  THE DRIVE WILL BE RECALIBRATED.

                EM60    DEVICE UNSAFE

                        THE INDICATED DRIVE UNSAFE ERROR OCCURRED; THE ERROR WAS
                        CLEARED BY A 'DRIVE CLEAR' INSTRUCTION.


        7.2     DETAIL ERROR LINES

                THE LINE NUMBERS GIVEN BELOW ARE FOR REFERENCE ONLY.

                LINE 1
                ------

                TT:TT:TT  (DESCRIPTION OF ERROR)

                'TT:TT:TT' IS THE TIME SINCE THE PROGRAM
                WAS STARTED.  TT:TT:TT IS GIVEN IN HOURS:
                MINUTES: SECONDS.

                LINE 2
                ------

                'PRESENT ORDER = XXXX  PREVIOUS ORDER = YYYY'

                MNEMONICS USED FOR THE ORDERS ARE DEFINED BELOW:

                        UNLOAD - UNLOAD (OCTAL 3)
                        SEEK -  SEEK (OCTAL 5)
                        RECAL - RECALIBRATE (OCTAL 7)
                        DRVCLR - DRIVE CLEAR (OCTAL 11)
                        RELSE - RELEASE (OCTAL 13)
                        OFFSET - OFFSET (OCTAL 15)
                        RTC - RETURN TO CENTERLINE (OCTAL 17)
                        READIN - READIN PRESET (OCTAL 21)
                        PACK - PACK ACKNOWLEDGE (OCTAL 23)
                        SEARCH - SEARCH (OCTAL 31)
                        WCKD -  WRITE CHECK DATA (OCTAL 51)
                        WCKHD - WRITE CHECK HEADER & DATA (OCTAL 53)
                        WRTDAT - WRITE DATA (OCTAL 61)
                        WRTHD - WRITE CHECK HEADER & DATA (OCTAL 63)
                        RDDAT - READ DATA (OCTAL 71)
                        RDHD -  READ HEADER & DATA (OCTAL 73)

                (DISPLAY OF THE RH11/RP04/5/6 REGISTERS IN TWO GROUPS:
                RPCS1,RPCS2,RPDS1,RPER1,RPER2,RPER3,RPEC1, & RPEC2 FORM THE FIRST
                GROUP;  ALL THE OTHER REGISTERS ARE IN THE SECOND GROUP.
                IF SW<05> IS SET, ONLY THE REGISTERS IN THE FIRST GROUP WILL BE
                DISPLAYED.)


                THE ABOVE LINE WILL BE TYPED IF THE ERROR OCCURRED DURING
                THE NON-DATA TRANSFER PART OF THE OPERATION.

                '* ERROR AT BAD TRACK/SECTOR'

                THE ABOVE LINE WILL BE PRINTED IF A DATA ERROR OCCURES AT AN ADDRESS
                ON THE PACK WHICH THE OPERATOR HAS IDENTIFIED AS BEING BAD.  PARAMETER
                'NOTPRT' MUST BE 0 FOR THE ERROR TO BE REPORTED.

                A WORD CALLED 'STATUS' IS DISPLAYED WITH THE RP04/5/6 REGISTERS.  THE
                CONTENTS OF THIS WORD IDENTIFY HOW THE ERROR WAS PROCESSED BY THE
                RP04/5/6 DRIVE HANDLER ROUTINE.  THE BITS IN THIS WORD ARE ENCODED
                AS FOLLOWS:

                        BIT #           MEANING IF BIT IS '1'
                        -----           ---------------------

                         15             ERROR OCCURRED
                                          DONE (BIT07=0), BITS 14-9, 2, 1 SPECIFY TYPE
                                          DONE (BIT07=1), BITS 6-3 SPECIFY TYPE

                         14             DRIVE IS OFFLINE

                         12             PERSISTENT UNSAFE CONDITION EXISTS

                         11             UNCORRECTABLE PARITY ERROR OCCURRED

                         10             FATAL PARITY ERROR OCCURRED.  MASSBUS CLEAR
                                          WAS PERFORMED

                          9             OPERATION NOT COMPLETED WITHIN 1 SECOND
                                          MASSBUS CLEAR PERFORMED.  ALL OTHER
                                          OUTSTANDING OPERATIONS WERE RESTARTED.

                          7             DONE - OPERATION COMPLETED

                          6             DATA ERROR OCCURRED DURING THE TRANSFER

                          5             ERROR OCCURRED WHILE SEARCHING FOR THE 'TRANSFER'
                                          SECTOR OR DURING RECALIBRATE OR OFFSET COMMANDS

                          4             CORRECTABLE UNSAFE CONDITION OCCURRED

                3              DRIVE ERROR OCCURRED THAT CAUSED AN AUTOMATIC
                                 RECALIBRATE SEQUENCE

                2              PORT REQUEST TIMEOUT

                1              NON-EXISTENT DRIVE REQUESTED

        LINE 3
        ------

        ERROR AT CXXX TYY SZZ  PREV ADDR = CUUU TVV SWW

        THE ACTUAL ADDRESS OF THE ERROR SECTOR AND THE PREVIOUS
        DISK ADDRESS ARE GIVEN IN THIS LINE.  CYLINDER, TRACK, &
        SECTOR ADDRESSES ARE IN DECIMAL.

        LINE 4
        -------

        PRESENT ADDR = CXXX TYY SZZ  PREV ADDR = CUUU TVV SWW

        THIS LINE IDENTIFIES THE ADDRESS WHEN THE ERROR WAS DETECTED;
        THE PREVIOUS ADDRESS IS ALSO GIVEN.  CYLINDER, TRACK, & SECTOR
        ADDRESSES ARE GIVEN IN DECIMAL.

        LINE 5
        -------

        START CYL = XXX END CYL = YYY

        THIS LINE IDENTIFIES THE STARTING CYLINDER OR A SEEK (IMPLIED)
        AND THE DESTINATION CYLINDER.  CYLINDER ADDRESSES ARE IN
        DECIMAL.

        LINE 6
        -------

        START CYL = XXX  END CYL = YYY  ACTUAL CYL = ZZZ

        THIS LINE IDENTIFIES THE STARTING CYLINDER OF AN IMPLIED SEEK,
        THE DESTINATION CYLINDER, AND THE CYLINDER THE DISK ACTUALLY
        STOPPED AT.  CYLINDER ADDRESSES ARE IN DECIMAL.

        LINE 7
        -------

        RPBA = XXXX  RPWC = YYYY

        THIS LINE GIVES THE CONTENTS OF THE RH11 BUFFER ADDRESS
        REGISTER AND THE RH11 WORD COUNT REGISTER.  THIS LINE IS
        NOT PRINTED IF SW<05> IS NOT SET.

        LINE 8
        -------

        START CYL = XXX  START TRK = YY  START SECTOR = ZZ
```

```

THIS LINE IDENTIFIES THE STARTING DISK ADDRESS OF THE PRESENT
OPERATION.  CYLINDER, TRACK, AND SECTOR VALUES ARE DECIMAL.

LINE 9
--------

RPDA = XXXX  RPCA = YYYY

THIS LINE GIVES THE CONTENTS OF THE RP04 TRACK AND SECTOR
ADDRESS REGISTER AND THE CONTENTS OF THE DESIRED CYLINDER
ADDRESS REGISTER.  THIS LINE IS NOT PRINTED IF SW<05> IS NOT
SET.

LINE 10
------

BUFFER ADDR = XXXX  SIZE = YYYY  ACTUAL NUMBR WRDS XFRD = ZZZZ

THIS LINE GIVES THE STARTING ADDRESS OF THE BUFFER USED FOR THE
CURRENT DATA TRANSFER OPERATION, ITS SIZE, AND THE ACTUAL NUMBER
OF WORD TRANSFERED.  THE STARTING ADDRESS OF THE BUFFER IS IN
OCTAL, THE SIZE AND WORD TRANSFERED VALUE ARE IN DECIMAL.

LINE 11
------

GOOD DATA = XXXX  BAD DATA = YYYY  SECT POS = ZZZ

THIS LINE GIVES THE GOOD DATA, THE ACTUAL DATA FROM THE DISK,
AND THE LOCATION IN THE SECTOR OF THE ACTUAL DATA.  THE SECTOR
POSITION IS IN DECIMAL.

LINE 12
--------

HEADER CONTENTS OF ERROR SECTOR = XXXX XXXX XXXX XXXX

THIS LINE GIVES THE CONTENTS OF THE HEADER OF THE SECTOR WHICH
GAVE THE ERROR.

LINE 13
--------

RPEC1 = XXXX  RPEC2 = YYYY

THIS LINE WILL BE PRINTED AFTER A SUCESSFUL RETRY OF A SECTOR
WHICH BECAME ECC CORRECTABLE DURING RETRY.

LINE 14
------

ECC CORRECTABLE WITHOUT OFFSET

THE SECTOR IN ERROR IS ECC CORRECTABLE; NO RETRY ATTEMPTS ARE
NECESSARY.

```

```
LINE 15
-------

READ CORRECTLY AT OFFSET  X  MICRO-INCHES

THE SECTOR IN ERROR WAS READ WITHOUT ERROR AT THE INDICATED
OFFSET VALUE.

LINE 16
-------

ECC CORRECTABLE AT OFFSET  X  MICRO-INCHES

THE SECTOR IN ERROR BECAME ECC CORRECTABLE AT THE INDICATED
OFFSET.

LINE 17
-------

CORRECTED ON X RETRY

THE OPERATION WAS PERFORMED ERROR FREE ON THE INDICATED RETRY
ATTEMPT.

LINE 18
-------

UNCORRECTABLE AFTER X RETRIES

THE OPERATION COUNT NOT BE PERFORMED CORRECTLY AFTER THE
INDICATED NUMBER OF RETRY ATTEMPTS.

LINE 19
------

DIFFERENT ERROR DURING RETRY

WHILE THE PROGRAM WAS RETRYING THE ERROR, A DIFFERENT OCCURRED.
IF THIS LINE IS PRINTED, THE RH11/RP04 REGISTERS WILL ALSO BE
PRINTED (SEE LINE 2).

LINE 20
-------

DATA COMPARISON ERRORS

A PRINTOUT OF THE DATA COMPARISON ERRORS FOLLOW THIS LINE.

LINE 21
-------

TOTAL COMPARE ERRORS = XXXX

THIS LINE GIVES THE TOTAL DATA COMPARISON ERROR COUNT.  THE
VALUE GIVEN IS IN DECIMAL.

LINE 22
```

```
--------

THE DATA COMPARED OK

THIS LINE INDICATES THE RESULTS OF THE DATA COMPARISON FOLLOWING
ECC CORRECTION.

LINE 23
--------

ECC CORRECTION RESULTS

THE PROGRAM PERFORMED ECC CORRECTION AND THE RESULTS ARE REPORTED.
THE ADDRESS IN MEMORY OF THE WORD(S) IN ERROR ARE GIVEN, THE WORD(S)
BEFORE CORRECTION AND THE WORD(S) AFTER CORRECTION ARE PRINTED.

LINE 24
-------

ERROR BURST BEGINS AT WORD XXX IN DATA FIELD OF ERROR SECTOR

THIS IS AN INFORMATIONAL LINE WHICH WILL BE PRINTED FOR 'DCK' ERRORS
WHICH ARE ECC CORRECTABLE OR WHICH BECOME ECC CORRECTABLE DURING
RETRY.  'XXX' IS THE WORD OFFSET VALUE FROM 'RPEC1' AND IS IN
DECIMAL.

LINE 25
--------

ERROR WAS NOT IN THE DATA READ -
ECC CORRECTION CAN'T BE PERFORMED

THE DATA ERROR WAS NOT IN DATA TRANSFERED TO MEMORY.

LINE 26
-------

CONTENTS OF THE ERROR SECTOR (REPORTED ABOVE)

IF SW<03> IS SET, THE SECTOR WHICH GAVE THE 'DCK', 'DTE' OR,
'WCF' ERROR OR 'HARD' DATA CHECK ERROR IS PRINTED.  THE
CONTENTS OF THE SECTOR FOLLOW THIS LINE.

LINE 27
------

ORDERS: WWW  ERRORS: X  WRDS XFR: YYYY  WRDS READ: ZZZZ

THIS IS THE LAST LINE PRINTED FOR ALL NON-POSITIONING
TYPE ERRORS.

'ORDERS' IS THE TOTAL NUMBER OF COMMANDS GIVEN TO THE DRIVE
WHICH REPORTED THE ERROR.

'ERRORS IS THE TOTAL ERROR COUNT FOR THE DRIVE AND INCLUDES
EVERY ERROR DETECTED, REGARDLESS OF TYPE.
```

'WRDS XFR' IS THE TOTAL NUMBER OF WORDS WRITTEN AND READ BY THE DRIVE.

'WRDS READ' IS THE TOTAL NUMBER OF WORD READ BY THE DRIVE.

LINE 28

ORDERS: WWWW TOTAL SEEKS: XXX TOTAL POS ERR = YYY TOTAL SKI, OCYL ERR = Z

THIS IS THE LAST LINE PRINTED FOR ALL POSITIONING TYPE ERRORS.

'ORDERS' IS THE TOTAL NUMBER OF ORDERS GIVEN TO THE DRIVE WHICH REPORTED THE ERROR.

'TOTAL SEEKS' IS THE TOTAL NUMBER OF SEEK OPERATIONS PERFORMED BY THE DRIVE.

'TOTAL POS ERR' IS THE TOTAL NUMBER OF POSITIONING ERRORS WHICH THE PROGRAM DETECTED FOR THE DRIVE.

'TOTAL SKI,OCYL ERR' IS THE TOTAL NUMBER OF 'SKI' OR 'OCYL' ERRORS SIGNALED BY THE DRIVE.

8. PROGRAM DESCRIPTION

8.1 PROGRAM OPERATION

WHEN THE PROGRAM IS STARTED, ALL TABLES AND PARAMETERS ARE CLEARED OR INITIALIZED. THE PARAMETERS WHICH ARE UNDER OPERATOR TTY ENTRY CONTROL ARE CHECKED FOR VALIDITY AND CONSISTENCY. RH11 INTERRUPT ENABLE ('IE') IS SET, TTY KEYBOARD INTERRUPT ENABLE IS SET, AND THE KW11-L OR KW11-P CLOCK IS STARTED. WHEN THESE ACTIONS HAVE BEEN COMPLETED, THE PROGRAM TYPES OUT 'PROGRAM INTIALIZE COMPLETE'. COMMAND ENTRIES WILL NOW BE ACCEPTED BY THE PROGRAM

THE PROGRAM SCANS ITS INTERNAL ASSIGNMENT TABLES, LOOKING FOR:

1) DRIVES TO ASSIGN/DEASSIGN
2) PERFORMANCE SUMMARY TYPEOUT REQUESTS
3) DRIVES REQUIRING COMMAND INITIATION, BUFFER ASSIGNEMENT, OR PARAMETER SELECTION.
4) DRIVES COMPLETING CURRENT OPERATIONS.

THE PROGRAM CONTINUES SCANNING ITS TABLES UNTIL AN ENTRY IS FOUND. IN THE CASE OF THE PROGRAM AT INITIAL START, THE FIRST ENTRY WILL BE MADE BY THE OPERATOR WHEN A DRIVE IS ASSIGNED ('T' COMMAND).

WHEN A DRIVE IS ASSIGNED, THE KEYBOARD ENTRY ROUTINE VERIFIES THAT THE DRIVE IS PRESENT, IS AN RP04/5/6, AND IS ONLINE. THE ASSIGNMENT ROUTINE THEN ISSUES A 'READIN PRESET' INSTRUCTION, SETS 'FMT22', AND ISSUES A 'RECALIBRATE' INSTRUCTION.

PARAMETERS FOR THE OPERATION ARE SELECTED AND A BUFFER IS ASSIGNED. IF

```
THE OPERATION IS A WRITE OR WRITE CHECK ORDER, THE ASSIGNED BUFFER
WILL BE FILLED WITH THE SELECTED PATTERN.  (WRITE CHECK ORDERS ARE
ISSUED AFTER EACH WRITE ORDER.  THE WRITE CHECK ORDER USES THE
PARAMETERS SELECTED FOR THE PRECEEDING WRITE ORDER.)  CONTROL
IS THEN PASSED TO THE COMMAND INITIATION ROUTINE.

THE COMMAND INITIATION ROUTINE FIRST LOOKS AT THE CYLINDER ADDRESS OF
THE REQUESTED OPERATION.  IF THE DRIVE MUST SEEK TO ANOTHER CYLINDER
PERFORM THE OPERATION, THE PROGRAM ISSUES A SEARCH INSTURCTION
TO THE DRIVE WITH A 'TARGET' SECTOR WHICH IS 8 SECTORS EARLIER
THAN THE 'TRANSFER' SECTOR.  (THIS ALLOWS THE PROGRAM TO INITIATE
OPERATIONS ON ANOTHER DRIVE WHILE THE PRESENT DRIVE, OR OTHER
DRIVES, ARE SEARCHING FOR 'TARGET' SECTORS.  ALL SEEKS ISSUED BY
THE PROGRAM ARE IMPLIED SEEK SEARCH OPERATIONS.)  WHEN A SEARCHING
DRIVE FINDS THE 'TARGET' SECTOR AND INTERRUPTS, THE PROGRAM READS
THE LOOK AHEAD REGISTER (RPLA) OF THE INTERRUPTING DRIVE AND
COMPARES THE POSITION OF THE DISK WITH THAT OF THE DESIRED SECTOR.

IF OTHER DRIVES ARE WAITING ON CYLINDER, THEY ARE ALSO CHECKED.  THE
PROGRAM THEN ISSUES THE REQUESTED ORDER TO THE DRIVE NEAREST ITS
TRANSFER SECTOR.  THE DRIVES NOT SELECTED WILL HAVE ANOTHER SEARCH
INITIATED.  IF A DRIVE IS NOT SELECTED FOR TRANSFER AFTER THREE
REVOLUTIONS OF ITS DISK, IT IS GIVEN PRIORITY OVER DRIVES WHICH HAVE
NOTE BEEN ON CYLINDER AS LONG.

WHEN THE DATA TRANSFER OPERATION IS COMPLETE, THE DRIVE REGISTERS
ARE STORED AND A DATA TRANSFER IS INITIATED FOR A WAITING DRIVE.

IF THE OPERATION HAS BEEN COMPLETED NORMALLY, THE SAVED DRIVE REGISTERS
ARE CHECKED TO VERIFY THAT NO ERROR BITS ARE SET; THE RH11 BUS ADDRESS
AND WORD COUNT ADDRESS REGISTERS ARE CHECKED TO VERIFY THAT THE
CORRECT NUMBER OF WORDS HAVE BEEN TRANSFERED AND THAT THE TWO REGISTERS
ARE CONSISTENT WITH EACH OTHER; AND IF THE ORDER WAS A READ ORDER, THE
DATA BUFFER IS COMPARED.  WHEN THIS SEQUENCE IS COMPLETED, THE DRIVE
IS RETURNED TO THE ASSIGNED, INACTIVE LIST.  THE PROGRAM THEN
INITIATES A DATA TRANSFER ON A WAITING DRIVE AND RESELECTS AND
REINITIATES ANOTHER OPERATION ON THE RELEASED DRIVE.

ERRORS WHICH OCCUR ARE PROCESSED IN THE FOLLOWING ORDER.  MULTIPLE
ERRORS WILL BE REPORTED AS THE FIRST ERROR TYPE CHECKED.

A.  ERRORS REPORTED FOR OPERATIONS WHICH HAVE NOT COMPLETED NORMALLY.

    PERSISTENT UNSAFE CONDITION - EM12
    UNCORRECTABLE MASSBUS PARITY ERROR - EM10
    FATAL MASSBUS PARITY ERROR - EM11
    OPERATION NOT COMPLETED WITHIN TIME LIMIT - EM13
    UNIT WENT OFFLINE - EM14

B.  ERRORS REPORTED FOR OPERATIONS WHICH COMPLETE NORMALLY.

    CORRECTABLE UNSAFE - EM60
    DRIVE TIMING ERROR - EM32
    DATA CHECK ERROR - EM21
    WRITE CHECK WITH DCK SET - EM22
    HEADER CRC ERRORS - EM20
    FORMAT ERRORS - EM24, EM26
```

HEADER COMPARE ERRORS - EM25, EM27
PROGRAM DETECTED POSITIONING ERROR - EM51
SEEK INCOMPLETE OR OFF CYLINDER ERROR - EM50
WRITE CHECK WITHOUT 'DCK' SET - EM23
RH11 OR UNIBUS TRANSFER ERROR - EM40
'OPI' ERROR - EM31
'PAR' ERROR - EM33
'WCF' ERROR - EM34
'IAE' ERROR - EM35
'WLE' ERROR - EM36
MISCELLANEOUS DRIVE ERROR - EM30

C. ERRORS NOT FLAGGED BY THE HARDWARE ERROR DETECTION LOGIC.

BUS ADDRESS OR WORD COUNT INCORRECT - EM41
DATA COMPARE ERRORS - NO DRIVE ERROR DETECTED - EM42
CAN'T MATCH DATA READ WITH A PATTERN - EM43
ERROR BIT(S) SET, BUT NO ERROR SIGNALED BY THE RH11 - EM44
ECC LOGIC FAILURE - EM45
BUS ADDRESS OR WORD COUNT NOT CONSISTENT - EM46

8.2 DUAL PORT OPERATION

PROGRAMMABLE DRIVES (DUAL PORT) OPERATION IS INHIBITED WITH STARTING ADDRESS 200-SEE SEC 2.4. THEREFORE, USE STARTING ADDRESS 204 OR 220 FOR DUAL PORT TESTING.

DUAL PORT OPERATION IS NEARLY IDENTICAL TO THE OPERATION DESCRIBED IN SECTION 8.1. THE DIFFERENCES ARE IN COMMAND SEQUENCE INITIATION AND ORDER TERMINATION.

WHEN THE DUAL PORT HANDLER ROUTINE IN THE MULTIDRIVE PROGRAM RECEIVES A REQUEST FOR A DRIVE, THE PROGRAM VERIFIES THAT THE DRIVE IS ONLINE. THE DRIVE IS SELECTED AND 0'S ARE WRITTEN INTO 'RPDS1': IF THE DRIVE IS IN NEUTRAL, THIS WILL SEIZE THE DRIVE. IF THE DRIVE IS SEIZED BY THE OTHER PORT, WRITING INTO 'RPDS1' WILL SET 'PORT REQUEST'. THE PROGRAM CHECKS 'DVA' IN 'RPCS1'. IF THE DRIVE IS AVAILABLE AS INDICATED BY THE 'DVA' BIT, THE COMMAND SEQUENCE WILL BE INITIATED IN THE NORMAL MANNER (SEE SECTION 8.1 ABOVE). IF 'DVA' WAS NOT SET, THE PROGRAM MAKES AN ENTRY FOR THE DRIVE IN AN INTERNAL 'PORT REQUEST PENDING' TABLE AND STARTS A 20 SECOND TIMER FOR THE DRIVE. IF THE DRIVE HAS NOT SWITCHED TO THE REQUESTING SYSTEM WITHIN THE 20 SECOND INTERVAL, THE PROGRAM REPORTS A 'NO RESPONSE TO PORT REQUEST' ERROR. NORMALLY THIS ERROR MESSAGE INDICATES A FAILURE IN THE DUAL PORT CONTROL LOGIC IN THE DRIVE BEING TESTED; HOWEVER, UNDER CERTAIN CONDITIONS (E.G. MASSBUS PARITY ERRORS BEING REPORTED ON THE OTHER SYSTEM ON A MOD33 TTY), THE OTHER PROCESSOR WAS UNABLE TO PROCESS THE DRIVE AFTER IT HAD REQUESTED THE DRIVE. THE OPERATOR MUST BE AWARE OF WHAT THE OTHER SYSTEM IS DOING AT ALL TIMES TO INTERPRET THE PORT RELATED ERROR MESSAGES PROPERLY.

AFTER A DRIVE HAS COMPLETED AN OPERATION, THE PROGRAM WILL STORE THE REGISTERS AND ISSUE A 'RELEASE' TO THE DRIVE; IF THE OPERATION TERMINATED WITH AN ERROR, THE DRIVE WILL NOT BE RELEASED UNTIL ERROR PROCESSING HAS BEEN COMPLETED.

SINGLE PORT DRIVES, DRIVES WHICH ARE IN NEUTRAL BUT NOT BEING EXERCISED BY THE OPPOSITE PORT ARE STILL TREATED AS DUAL PORT DRIVES IN THAT A RELEASE COMMAND IS ISSUED AT THE END OF NORMAL ORDER PROCESSING OR AT THE END OF ERROR PROCESSING. A RELEASE COMMAND ISSUED UNDER THESE CONDITIONS HAS NO FUNCTIONAL EFFECT ON THE OPERATION OF THE DRIVE.

8.3 SELECTION OF OPERATION VARIABLES

A. SECTOR ADDRESS SELECTION IS RANDOM BETWEEN THE VALUES IN 'MINSEC' AND 'MAXSEC'. TRACK ADDRESS SELECTION IS RANDOM BETWEEN THE VALUES IN 'MINTRK' AND 'MAXTRK'. CYLINDER ADDRESS SELECTION IS RANDOM BETWEEN 'MINCYL' AND 'MAXCYL'. IF A MINIMUM ADDRESS IS GREATER THAN THE CORRESPONDING MAXIMUM ADDRESS, THE PROGRAM WILL EXCLUDE ALL ADDRESSES BETWEEN 'MAX' AND 'MIN' FROM THE SELECTION. FOR EXAMPLE: IF 'MINTRK' IS 18 AND 'MAXTRK' IS 5, THEN TRACK ADDRESS SELECTION WILL EXCLUDE TRACKS 6 - 17 FROM THE SELECTION AND SELECT AN ADDRESS FROM AMONG ADDRESSES 18, 0, 1, 2, 3, 4, 5.

B. THE BUFFER SIZE IS RANDOM SELECTED BETWEEN 4 (10) - AND THE VALUE IN 'MAXDL'. THE SIZE SELECTED IS WEIGHTED TO ENSURE THAT AT LEAST 4 WORDS ARE WRITTEN IN THE DATA AREA OF THE LAST SECTOR. THIS IS NECESSARY AS THE PROGRAM REQUIRES 4 LOCATIONS IN THE DATA PORTION OF THE SECTOR TO BE ABLE TO MATCH THE DATA TO A PATTERN FOR DATA COMPARISON PURPOSES.

C. THE DATA WRITTEN IS RANDOMLY SELECTED AMONG THE 15 STANDARD PATTERNS. THE KEYWORDS IN THE HEADER (WHEN PERFORMING A WRITE HEADER & DATA ORDER) ARE ZERO FILLED. THE PROGRAM EXPECTS TO FIND THAT THE KEYWORDS ARE ZERO.

D. THE ORDERS ARE SELECTED RANDOMLY. WRITE CHECK DATA AND WRITE CHECK HEADER & DATA ORDERS ARE PERFORMED ONLY IF THE PREVIOUS ORDER WAS THE APPROPRIATE DATA ORDER. IF THE 'FORMAT' PARAMETER IS ZERO, THE PROGRAM WILL NOT SELECT WRITE HEADER & DATA (AND WRITE CHECK HEADER & DATA) ORDERS. WHEN THE PROGRAM SELECTS A WRITE HEADER & DATA ORDER, THE BUFFER SIZE IS FORCED TO 260 (10); THE PROGRAM WILL NOT PERFORM A MULTI-SECTOR FORMAT WRITE OPERATION.

E. THE FIRST ORDER PERFORMED AFTER A UNIT IS ASSIGNED WITH A 'T', 'W', OR 'R' COMMAND IS NOT RANDOMLY SELECTED. THE PARAMETERS FOR THE FIRST OPERATION ARE THE MINIMUM OR STARTING VALUES OF THE VARIABLES.

8.4 DATA PATTERNS

THE PROGRAM SELECTS ONE OF THE FOLLOWING DATA PATTERNS TO WRITE WHEN A WRITE ORDER IS SELECTED. THE ENTIRE BUFFER IS FILLED WITH THE SELECTED PATTERN. WHEN DATA IS READ FROM THE DISK, THE PROGRAM COMPARES DATA ON A SECTOR BASIS: FROM THE FIRST 4 DATA WORDS OF EACH SECTOR, THE PROGRAM MATCHES THE DATA TO ONE OF THE FOLLOWING PATTERNS. TO MANTAIN COMPATIABILITY WITH PACKS WRITTEN BY THE FORMAT PROGRAM (CZRJB), THE PROGRAM WILL ACCEPT ALL ZERO'S AND AND ALL ONE'S PATTERNS; HOWEVER, ALL ZERO'S AND ALL ONE'S PATTERNS ARE NOT WRITTEN BY THE EXERCISER PROGRAM.

```

                        PATTERN '8' IS DEFINED AS THE 'WORST CASE' PATTERN.

                        PAT 1   PAT 2   PAT 3   PAT 4   PAT 5   PAT 6   PAT 7   PAT 8
                        ------  ------  ------  ------  ------  ------  ------  ------
                        000001  177776  000000  000000  052525  007417  026455  165555
                        000003  177774  000000  010421  052525  007417  026455  133333
                        000007  177770  000000  021042  052525  007417  026455  165555
                        000017  177760  177777  031463  125252  170360  151322  133333
                        000037  177740  177777  042104  125252  170360  151322  165555
                        000077  177700  177777  052525  125252  170360  151322  133333
                        000177  177600  000000  063146  052525  007417  026455  165555
                        000377  177400  000000  073567  052525  007417  026455  133333
                        000777  177000  177777  104210  125252  170360  151322  165555
                        001777  176000  177777  114631  125252  170360  151322  133333
                        003777  174000  000000  125252  052525  007417  026455  165555
                        007777  170000  177777  135673  125252  170360  151322  133333
                        017777  160000  000000  146314  052525  007417  026455  165555
                        037777  140000  177777  156735  125252  170360  151322  133333
                        077777  100000  000000  167356  052525  007417  026455  165555
                        177777  000000  177777  177777  125252  170360  151322  133333

                        PAT 9   PAT 10  PAT 11  PAT 12  PAT 13  PAT 14  PAT 15
                        ------  ------  ------  ------  ------  ------  ------
                        000001  177776  172666  077777  153333  000000  177777
                        000002  177775  155555  137777  066667  177777  000000
                        000004  177773  172666  157777  153333  177777  000000
                        000010  177767  155555  167777  066667  177777  000000
                        000020  177757  172666  173777  153333  177777  000000
                        000040  177737  155555  175777  066667  177777  000000
                        000100  177677  172666  176777  153333  177777  000000
                        000200  177577  155555  177377  066667  177777  000000
                        000400  177377  172666  177577  153333  177777  000000
                        001000  176777  155555  177677  066667  177777  000000
                        002000  175777  172666  177737  153333  177777  000000
                        004000  173777  155555  177757  066667  177777  000000
                        010000  167777  172666  177767  153333  177777  000000
                        020000  157777  155555  177773  066667  177777  000000
                        040000  137777  172666  177775  153333  177777  000000
                        100000  077777  155555  177776  066667  177777  000000


                9.      RP/RH DRIVER DOCUMENT
                        ---------------------

                9.1     RH11/RP04/5/6 DRIVER

                        THIS DOCUMENT IS THE USER'S GUIDE FOR THE RH11/RP04/5/6 DRIVER.
                        THE DRIVE INITIALIZATION ROUTINE HAS BEEN MODIFIED TO
                        FLAG A PROGRAMMABLE DRIVE (OCT 1978).

                9.2     TO INITIALIZE THE DRIVER:

                                JSR     PC,RPINIT
                                RETURN

                        UPON RETURN YOU MUST EXAMINE THE "DRVSTA" TABLE TO DETERMINE
                THE DRIVES THAT ARE ONLINE FOR TESTING.  THE 'DRVSTA' TABLE IS
                EIGHT BYTES; ONE BYTE PER DRIVE.  THE STATE OF EACH DRIVE WILL
                BE INDICATED AS FOLLOWS:

                        DRVSTA          DRIVE STATE
                        ------          -----------

                          >0            ONLINE RP04/5/6
                          =0            OFFLINE RP04/5/6, DRIVE
                                        IS NOT AN RP04/5/6, OR
                                        NONEXISTENT DRIVE
                          <0            UNSAFE RP04/5/6

                THE DRIVE TYPE IS DEFINED IN AN 8 BYTE LONG TABLE TAGGED 'DRVTYP'.
                THE TABLE CONTAINS ONE BYTE FOR EACH DRIVE AND IS INDEXED BY THE
                DRIVE NUMBER.  ENTRIES ARE ENCODED AS FOLLOWS:

                        DRVTYP          CONDITION
                        ------          ---------

                           0            NONEXISTENT DRIVE
                           1            RP04
                           2            RP05
                           4            RP06
                          10            PROGRAMMABLE DRIVE
                          -1            NOT AN RP04/5/6

                THE 'RPINIT' ROUTINE WILL DO A READIN PRESET AND WILL SET FMT22.

        9.3     AFTER THE DRIVER HAS BEEN INITIALIZED, IT IS CALLED USING THE
                FOLLOWING SEQUENCE.

                CALL:
                        JSR     R0,RP04                 ;MAKE THE CALL
                        PNTDPB                          ;ADDRESS OF DPB*
                        RETURN1                         ;RETURN IF QUEUE IS FULL
                        RETURN2                         ;RETURN IF REQUEST IS IN
                                                        ;QUEUE OR THERE IS AN
                                                        ;ERROR CONDITION

                *DPB (DATA PARAMETER BLOCK)

                PNTDPB: .BYTE   0                       ;(0) DRIVE NUMBER
                        .BYTE   0                       ;(1) OFFSET VALUE OR FMT22, ECT, AND HCI
                        .BYTE   0                       ;(2) COMMAND
                        .BYTE   0                       ;(3) PSEL AND A17 AND A16
                        .WORD   0                       ;(4) WORD COUNT (MUST BE NEG.)
                        .WORD   0                       ;(6) BUFFER ADDRESS OR
                                                        ;REGISTER TABLE POINTER
                        .BYTE   0                       ;(10) SECTOR ADDRESS OR
                                                        ;FIRST REG. INDEX
                        .BYTE   0                       ;(11) TRACK ADDRESS OR
                                                        ;LAST REG. INDEX
                        .WORD   0                       ;(12) CYLINDER ADDRESS
                        .WORD   0                       ;(14) ERROR TABLE POINTER
                                                        ;POINTS TO THE FIRST OF TWENTY
                                                        ;LOCATIONS OF WHERE THE DRIVER
```

```
                                                 ;IS TO STORE THE RH11/RP04
                                                 ;REGISTERS ON AN ERROR. IF LEFT
                                                 ;ZERO REGISTERS ARE NOT SAVED.
                       .WORD    0                ;(16) STATUS/ERROR INDICATOR
                                                 ;BIT15=1=>ERROR OCCURRED
                                                 ;BIT07=1=>DONE
                                                 ;BIT14-BIT09 AND BIT06-BIT03
                                                 ;INDICATE TYPE OF ERROR

        9.4     THE DRIVER PROVIDES A SOFTWARE TIMEOUT CAPABILITY.
                TO UTILIZE THIS CAPABILITY YOU MUST SUPPLY THE 'RP TIMER'' ROUTINE
                WITH THE ELAPSED TIME IN THE FOLLOWING MANNER:

                       MOV      #16.,-(SP)       ;16 MILLISECONDS BETWEEN
                                                 ;CLOCK TICKS
                       JSR      PC,RPTMR         ;CALL THE TIMER ROUTINE

                IT SHOULD BE NOTED THAT YOU MUST PROVIDE THE CODE TO DRIVE THE
                CLOCK.  AND THE ELAPSED TIME MUST BE IN MILLISECONDS.
                THE DRIVER WILL SET THE TIMEOUT TO 1 SECOND FOR ALL POSITIONING
                AND DATA TRANSFER OPERATIONS AND WILL SET THE TIMOUT TO 30
                SECONDS FOR ERROR RECOVERY OPERATIONS.


        4.1     EXAMPLE - WRITE 1000. WORDS

                1$:    JSR      R0,RP03          ;CALL THE DRIVER
                       WRTDPB                    ;DPB ADDRESS
                       BR       1$               ;WAIT FOR QUEUE IF FULL
                2$:    TST      WRTDPB+16        ;WAIT FOR COMMAND TO COMPLETE
                       BEQ      2$
                       BMI      ERROR1           ;ERROR OCCURRED
                       .
                       .
                       .
                       .
                WRTDPB: .BYTE   5                ;DRIVE #5
                       .BYTE    0                ;
                       .BYTE    161              ;WRITE COMMAND
                       .BYTE    0                ;
                       .WORD    -1000.           ;WORD COUNT
                       .WORD    WRTBUF           ;BUFFER ADDRESS
                       .BYTE    3                ;SECTOR
                       .BYTE    5                ;TRACK
                       .WORD    400              ;CYLINDER
                       .WORD    ERRTB5           ;ERROR TABLE
                       .WORD    0                ;STATUS/ERROR INDICATOR

                       ALTERNATE DPB SETUP

                WRTDPB: .WORD   5                ;THIS SETUP ACHIEVED
                       .WORD    WRITE            ;EVERYTHING THE
                       .WORD    -1000.           ;ABOVE TABLE DID, BUT
                       .WORD    WRTBUF           ;IN A CLEANER FORMAT
                       .BYTE    3,5
                       .WORD    400,ERRTB5,0
```

```
9.5     RH11/RP04/5/6 REGISTERS

                MNEMONIC                INDEX
                --------                -----
                RPCS1                     0
                RPWC                      2
                RPBA                      4
                RPDA                      6
                RPCS2                    10
                RPDS1                    12
                RPER1                    14
                RPAS                     16
                RPLA                     20
                RPDB                     22
                RPMR                     24
                RPDT                     26
                RPSN                     30
                RPOF                     32
                RPCA                     34
                RPCC                     36
                RPER2                    40
                RPER3                    42
                RPEC1                    44
                RPEC2                    46

9.6     COMMANDS PERFORMED BY THE DRIVER

                COMMAND                 CODE            COMMAND TYPE
                -------                 ----            ------------
                NO OPERATION            101                  N
                UNLOAD                  103                  N
                SEEK                    105                  P
                RECALIRATE              107                  P
                DRIVE CLEAR             111                  N
                RELEASE                 113                  N
                OFFSET                  115                  P
                RETURN TO CENTER        117                  P
                READIN PRESET           121                  N
                PACK ACKNOWLEDGE        123                  N
                SEARCH                  131                  P
                GET REGISTER(S)         141                  S
                SET FORMAT              143                  S
                SELECT DRIVE            145                  S
                WRITE CHECK DATA        151                  D
                WRITE CHECK HDR & DATA  153                  D
                WRITE DATA              161                  D
                WRITE HEADER & DATA     163                  D
                READ DATA               171                  D
                READ HEADER & DATA      173                  D

                N = HOUSEKEEPING
                P = POSITIONING
                D = DATA TRANSFER
                S = SPECIAL PROVIDED BY THE DRIVER

9.7     DPB STATUS/ERROR INDICATOR WORD
```

```
THIS INDICATOR WILL INFORM THE USER OF THE RESULTS OF THE REQUEST.
THIS IS ACCOMPLISHED BY SETTING VARIES BITS OF THE INDICATOR TO
A ONE.

          BIT NO.          MEANING IF ON A "1"
          -------          -------------------

          15               ERROR OCCURRED
                             DONE (BIT07=0); BITS 14-10 SPECIFIES TYPE
                             DONE (BIT07=1); BITS 06-03 SPECIFIES TYPE

          14(1)            USER MADE A REQUEST FOR A FUNCTION TO BE
                           PERFORMED ON AN OFFLINE OR UNSAFE DRIVE

          13(1)            USER MADE A REQUEST FOR A FUNCTION
                           TO BE PERFORMED ON A DRIVE THAT HAS AN
                           UNLOAD REQUEST IN QUEUE.

          12(2)            PERSISTENT UNSAFE CONDITION EXIST.

          11(2)            UNCORRECTABLE PARITY ERROR OCCURRED

          10(2)(4)         FATAL PARITY ERROR.  A MASSBUS CLEAR WAS
                           PERFORMED, ALL QUEUES WERE EMPTIED, AND
                           ALL DRVACT'S SET TO THE IDLE STATE

          9(3)(4)          SOFTWARE TIMEOUT OCCURRED ON THIS DRIVE

          8(4)             SOFTWARE TIMEOUT OCCURRED ON ANOTHER DRIVE

          7                DONE

          6(2)             ERROR OCCURRED DURING AN I/O OPERATION

          5(2)             ERROR OCCURRED DURING AN OPERATION
                           OTHER THAN I/O.

          4(2)             CORRECTABLE UNSAFE CONDITION OCCURRED

          3(2)             DRIVE ERROR OCCURRED THAT CAUSED AN
                           AUTOMATIC "RECALIBRATE" SEQUENCE

          2                PORT REQUEST TIMEOUT.  THE DRIVER REQUESTED
                           THE DRIVE BUT THE OPPOSITE PORT DID NOT
                           RELEASE THE DRIVE WITHIN 20 SECONDS.

          1                NON-EXISTENT DRIVE REQUESTED.  USER MADE
                           A REQUEST FOR A NON-EXISTENT DRIVE.

          ------------------------------------------------------------
          NOTES FOR ABOVE
          ------------------------------------------------------------
          (1) =>           REQUEST WASN'T PUT IN QUEUE.  (RH11/RP04
                           REGISTERS WERE NOT SAVED)

          (2) =>           REQUEST QUEUE HAS BEEN EMPTIED.  THE DRIVER
                           ISSUED A "DRIVE CLEAR" TO THE DRIVE.
```

```
                                  NOTE: ALL RH11/RP04 REGISTERS ARE SAVED
                                  AS PER DPB+14 BEFORE THE 'DRIVE CLEAR''.

                   (3) =>         REQUEST QUEUE HAS BEEN EMPTIED.  THE
                                  DRIVER ISSUED A MASSBUS INIT.  ALL
                                  RH11/RP04 REGISTERS FOR THE DRIVE WERE
                                  SAVED AS PER DPB+14 BEFORE THE INIT.

                   (4) =>         A ''RECALIBRATE'' SHOULD BE ISSUED
                                  BEFORE ANY OTHER COMMAND.

9.8     ERROR CALLS MADE BY THE DRIVER.

        THERE ARE A FEW ERRORS THAT CAN OCCUR THAT CAN NOT BE INDICATED IN A DPB.

        WHEN THIS TYPE OF ERROR IS DETECTED BY THE DRIVER IT WILL MAKE
        AN ERROR CALL OF THE FORM ''ERROR N'', WHERE ''N'' IS THE ERROR
        NUMBER AND THE ERROR WILL BE AN EMT INSTRUCTION.


        N       TYPE                    DATA AVAILABLE
        -       ----                    --------------

        1       RH11 INTERRUPT          *R4= RPCS1'S ADDRESS
                OCCURRED (RHAS=0)

        2       UNEXPECTED ATTENTION     R1= DRIVE NUMBER
                OCCURRED                 R3= ATA BIT
                                        *R4= RPCS1'S ADDRESS
                                         R5= (RPAS)
                                         RPERRS  =RPDS1
                                         RPERRS+2=RPER1
                                         RPERRS+4=RPER2
                                         RPERRS+6=RPER3

        3       MASSBUS PARITY           RD.ADR= ADDRESS OF REG. READ
                ERROR (MCPE=1)           RD.WRD= WORD READ

        4       MASSBUS PARITY           WRT.AD= ADDRESS OF REG. WRITTEN
                ERROR (PAR=1)            WRT.WD= WORD WRITTEN
                                         RD.WRD= WORD READ BACK

        5       ADDRESS PLUG CHANGE      R1= DRIVE NUMBER
                BIT SET ('OPE' ERROR)    R3= ATA BIT
                                        *R4= RPCS1'S ADDRESS
                                         R5= (RPAS)
                                         RPERRS  =RPDS1
                                         RPERRS+2=RPER1
                                         RPERRS+4=RPER2
                                         RPERRS+6=RPER3

        * THIS IS THE ACTUAL UNIBUS ADDRESS (176700)


10.     PROGRAM LISTING
        ---------------
@
```

```
                        ;*LAST REVISION 05-NOV-81

                        .TITLE  CZRJDEO RP04/5/6 MLT-DR LGC
                        ;*COPYRIGHT (C) 1975,1979,1981
                        ;*DIGITAL EQUIPMENT CORPORATION
                        ;*COLORADO SPGS., CO. 80919
                        ;*
                        ;*
                        ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                        ;*PACKAGE (MAINDEC-11-DZQAC-C5), 18-MAR-81
                        ;*
                        .SBTTL  OPERATIONAL SWITCH SETTINGS
                        ;*
                        ;*      SWITCH               USE
                        ;*      ------          --------------------
                        ;*        15            HALT ON ERROR
                        ;*        13            INHIBIT ERROR TYPEOUTS
                        ;*        10            BELL ON ERROR
                        ;*         7            DISPLAY ALL DATA COMPARE ERRORS
                        ;*         6            DON'T CHANGE PARAMETERS (LOOP ON PRESENT VALUES)
                        ;*         5            A. PARTIAL REGISTER DISPLAY IF ERROR
                        ;*                      B. NO ECC CORRECTION RESULTS DISPLAYED IF ERROR
                        ;*         4            A. DO NOT CHECK FOR MAXIMUM ERROR COUNTS
                        ;*                      B. DO NOT DROP DRIVE WHEN NORMAL END OF TEST REACHED
                        ;*         3            A. DISPLAY ERROR SECTOR IF 'DCK', 'DTE', OR 'WCF' ERROR
                        ;*                      B. DISPLAY SECTOR IF 'DCK' ERR UNCORRECTABLE AFTER
                        ;*                         28TH RETRY
                        ;*                      C. IF DATA COMPARE ERROR & SW7 SET, DISPLAY
                        ;*                         REMAINDER OF BUFFER
                        ;*         2            A. DON'T TYPE SUBSYSTEM STATUS WHEN PROGRAM STARTED
                        ;*                      B. DON'T TYPE PERFORMANCE SUMMARY
                        ;*         1            INHIBIT DATA COMPARSION AFTER READ ORDERS
                        ;*         0            READ ONLY MODE

                        .SBTTL  BASIC DEFINITIONS

                        ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
        001100          STACK   = 1100
        104000          ERROR   = EMT           ;;BASIC DEFINITION OF ERROR CALL
        000004          SCOPE   = IOT           ;;BASIC DEFINITION OF SCOPE CALL

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

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

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

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

                                ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
        100000                  BIT15   = 100000
        040000                  BIT14   = 40000
        020000                  BIT13   = 20000
        010000                  BIT12   = 10000
        004000                  BIT11   = 4000
        002000                  BIT10   = 2000
        001000                  BIT09   = 1000
        000400                  BIT08   = 400
        000200                  BIT07   = 200
        000100                  BIT06   = 100
        000040                  BIT05   = 40
        000020                  BIT04   = 20
        000010                  BIT03   = 10
        0C0004                  BIT02   = 4
        000002                  BIT01   = 2
        000001                  BIT00   = 1
        001000                  BIT9=BIT09
        000400                  BIT8=BIT08
        000200                  BIT7=BIT07
        000100                  BIT6=BIT06
        000040                  BIT5=BIT05
        000020                  BIT4=BIT04
        000010                  BIT3=BIT03
        000004                  BIT2=BIT02
        000002                  BIT1=BIT01
        000001                  BIT0=BIT00

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

                                .SBTTL  RH11 REGISTERS

                                ;CONTROL AND STATUS REGISTER 1 (RPCS1)

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



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


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


                                ;CONTROL AND STATUS REGISTER 2 (RPCS2)

        000001                  US1=    1                       ;UNIT SELECT (BIT #0)
        000002                  US2=    2                       ;UNIT SELECT (BIT #1)
        000004                  US4=    4                       ;UNIT SELECT (BIT #2)
        000010                  BAI=    10                      ;BUS ADDRESS INCREMENT INHIBIT (BIT #3)
        000020                  PAT=    20                      ;MASSBUS PARITY TEST (BIT #4)
        000040                  CLR=    40                      ;CLEAR (BIT #5)
        000100                  IR=     100                     ;INPUT READY (BIT #6)
        000200                  OR=     200                     ;OUTPUT READY (BIT #7)
        000400                  MPE=    400                     ;MASS BUS PARITY ERROR (BIT #8)
        001000                  MXF=    1000                    ;MISSED TRANSFER ERROR (BIT #9)
        002000                  PGE=    2000                    ;PROGRAM ERROR (BIT #10)
        004000                  NEM=    4000                    ;NON EXISTENT MEMORY (BIT #11)
        010000                  NED=    10000                   ;NON EXISTENT DRIVE (BIT #12)
        020000                  UPE=    20000                   ;UNIBUS PARITY ERROR (BIT #13)
        040000                  WCE=    40000                   ;WRITE CHECK ERROR (BIT #14)
        100000                  DLT=    100000                  ;DATA LATE (BIT #15)


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


                                .SBTTL  RP04/5/6 REGISTERS

                                ;CONTROL AND STATUS 1 REGISTER. (#00)

        000001                  GO=     1                       ;GO BIT (BIT #0)
        000002                  F1=     2                       ;FUNCTION CODE BIT #1
        000004                  F2=     4                       ;FUNCTION CODE BIT #2
        000010                  F3=     10                      ;FUNCTION CODE BIT #3
        000020                  F4=     20                      ;FUNCTION CODE BIT #4
        000040                  F5=     40                      ;FUNCTION CODE BIT #5
        004000                  DVA=    4000                    ;DEVICE AVAILABLE (BIT #11)

                                ;DRIVE STATUS REGISTER (RPDS1) (#01)

                                ;DF5=   1                       DRIVE FORWARD 5''/SEC.  (BIT #0)
        000002                  DFF20=  2                       ;DRIVE FORWARD 20''/SEC.  (BIT #1)
        000004                  DIGB=   4                       ;DRIVE TO INNER GUARD BAND (BIT #2)
        000010                  GRV=    10                      ;GO REVERSE (BIT #3)
        000020                  DL64=   20                      ;DIFFERENCE LESS THAN 64 (BIT #4)
        000040                  DE1=    40                      ;DIFFERENCE EQUALS 1 (BIT #5)
        000100                  VV=     100                     ;VOLUME VALID (BIT #6)
        000200                  DRY=    200                     ;DRIVE READY (BIT #7)
        000400                  DPR=    400                     ;DRIVE PRESENT (BIT #8)
        001000                  PGM=    1000                    ;PROGRAMABLE (BIT #9)
        002000                  LST=    2000                    ;LAST SECTOR TRANSFERRED (BIT #10)
        004000                  WRL=    4000                    ;WRITE LOCK (BIT #11)
        010000                  MOL=    10000                   ;MEDIUM ON-LINE (BIT #12)
        020000                  PIP=    20000                   ;POSITIONING OPERATION IN PROGRESS (BIT #13)
        040000                  ERR=    40000                   ;COMPOSITE ERROR (BIT #14)
        100000                  ATA=    100000                  ;ATTENTION ACTIVE (BIT #15)

                                ;ERROR REGISTER #01 (RPER1) (#02)

        000001                  ILF=    1                       ;ILLEGAL FUNCTION (BIT #0)
        000002                  ILR=    2                       ;ILLEGAL REGISTER (BIT #1)
        000004                  RMR=    4                       ;REGISTER MODIFICATION REFUSED (BIT #2)
        000010                  PAR=    10                      ;PARITY ERROR (BIT #3)
```

```
000020                  FER=    20                      ;FORMAT ERROR (BIT #4)
000040                  WCF=    40                      ;WRITE CLOCK FAIL (BIT #5)
000100                  ECH=    100                     ;ECC HARD ERROR (BIT #6)
000200                  HCE=    200                     ;HEADER COMPARE ERROR (BIT #7)
000400                  HCRC=   400                     ;HEADER CRC ERROR (BIT #8)
001000                  AOE=    1000                    ;ADDRESS OVERFLOW ERROR (BIT #9)
002000                  IAE=    2000                    ;INVALID ADDRESS ERROR (BIT #10)
004000                  WLE=    4000                    ;WRITE LOCK ERROR (BIT #11)
010000                  DTE=    10000                   ;DRIVE TIMING ERROR (BIT #12)
020000                  OPI=    20000                   ;OPERATION INCOMPLETE (BIT #13)
040000                  UNS=    40000                   ;DRIVE UNSAFE (BIT #14)
100000                  DCK=    100000                  ;DATA CHECK ERROR (BIT 15)

                        ;MAINTAINABILITY REGISTER (RPMR)(#03)

000001                  DMD=    1                       ;DIAGINOSTIC MODE (BIT #0)
000002                  MCLK=   2                       ;MAINTAINABILITY CLOCK (BIT #1)
000004                  MINX=   4                       ;MAINTAINABILITY INDEX (BIT #2)
000010                  MSTCK=  10                      ;MAINTAINABILITY SECTOR CLOCK (BIT #3)
000020                  MRD=    20                      ;MAINTAINABILITY READ (BIT #4)
000040                  MWR=    40                      ;MAINTAINABILITY WRITE (BIT #5)
000200                  DTSY=   200                     ;MAINTAINABILITY SYNC DETECTED (BIT #7)

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

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


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


                        ;DRIVE TYPE REGISTER (RPDT) (#06)

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

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

```

```
000001          EXT1=   1               ;EXTENSION 1 (BIT #0)
000002          EXT2=   2               ;EXTENSION 2 (BIT #1)
000004          EXT4=   4               ;EXTENSION 3 (BIT #2)
000010          EXT10=  10              ;EXTENSION 4 (BIT #3)
000020          EXT20=  20              ;EXTENSION 5 (BIT #4)
000040          EXT40=  40              ;EXTENSION 6 (BIT #5)
000100          SC1=    100             ;SECTOR COUNT FIELD 0 (BIT #6)
000200          SC2=    200             ;SECTOR COUNT FIELD 1 (BIT #7)
                ;SC4=   400             ;SECTOR COUNT FIELD 2 (BIT #8)
001000          SC10=   1000            ;SECTOR COUNT FIELD 3 (BIT #9)
002000          SC20=   2000            ;SECTOR COUNT FIELD 4 (BIT #10)
004000          TRK1=   4000            ;TRACK FIELD 1 (BIT #11)
010000          TRK2=   10000           ;TRACK FIELD 2 (BIT #12)
020000          TRK4=   20000           ;TRACK FIELD 3 (BIT #13)
040000          TRK10=  40000           ;TRACK FIELD 4 (BIT #14)
100000          TRK20=  100000          ;TRACK FIELD 5 (BIT #15)

                ;RP04 ERROR REGISTER #2 (RPER2) (#10)

000001          WCU=    1               ;WRITE CURRENT UNSAFE (BIT #0)
000002          CSF=    2               ;CURRENT SINK FAILURE (BIT #1)
000004          WSU=    4               ;WRITE SELECT UNSAFE (BIT #2)
000010          CSU=    10              ;CURRENT SWITCH UNSAFE (BIT #3)
000020          MSE=    20              ;MOTOR SEQUENCE ERROR (BIT #4)
000040          TDF=    40              ;TRANSITIONS DETECTOR FAILURE (BIT #5)
000100          TUF=    100             ;TRANSITIONS UNSAFE (BIT #6)
000200          FEN=    200             ;FAILSAFE ENABLED (BIT #7)
000400          WRU=    400             ;WRITE READY UNSAFE (BIT #8)
001000          MHS=    1000            ;MULTIPLE HEAD SELECT (BIT #9)
002000          NHS=    2000            ;NO HEAD SELECTION (BIT #10)
004000          IXE=    4000            ;INDEX ERROR (BIT #11)
010000          VU30=   10000           ;30VOLT UNSAFE (BIT #12)
020000          PLU=    20000           ;PLO UNSAFE (BIT #13)
100000          ACU=    100000          ;AC UNSAFE (BIT #15)

                ;RP05/6 ERROR REGISTER #02 (RPER2) (#10)

000001          WCU=    1               ;WRITE CURRENT UNSAFE (BIT #0)
000002          CSF=    2               ;CURRENT SINK FAILURE (BIT #1)
000004          WSU=    4               ;WRITE SELECT UNSAFE (BIT #2)
000010          CSU=    10              ;CURRENT SWITCH UNSAFE (BIT #3)
000020          RAW=    20              ;READ AND WRITE (BIT #4)
000040          TDF=    40              ;TRANSITIONS DETECTOR FAILURE (BIT #5)
000100          TUF=    100             ;TRANSITIONS UNSAFE (BIT #6)
000200          ABS=    200             ;ABNORMAL STOP (BIT #7)
000400          WRU=    400             ;WRITE READY UNSAFE (BIT #8)
001000          MHS=    1000            ;MUTLTIPLE HEAD SELECT (BIT #9)
002000          NHS=    2000            ;NO HEAD SELECTION (BIT #10)
004000          IXE=    4000            ;INDEX ERROR (BIT #11)
020000          PLU=    20000           ;PLO UNSAFE (BIT #12)

                ;OFFSET REGISTER (RPOF) (#11)

000001          OF25=   1               ;OFFSET 25 MICRO INCHES (BIT #0)
000002          OF50=   2               ;OFFSET 50 MICRO INCHES (BIT #1)
000004          OF100=  4               ;OFFSET 100 MICRO INCHES (BIT #2)
000010          OF200=  10              ;OFFSET 200 MICRO INCHES (BIT #3)
```

```
000020              OF400=  20                     ;OFFSET 400 MICRO INCHES (BIT #4)
000040              OF800=  40                     ;OFFSET 800 MICRO INCHES (BIT #5)
000200              OFREV=  200                    ;OFFSET NEGATIVE (REVERSE) (BIT #5)
002000              HCI=    2000                   ;HEADER COMPARE INHIBIT (BIT #10)
004000              ECI=    4000                   ;ERROR CORRECTION CODE INHIBIT (BIT #11)
010000              FMT22=  10000                  ;FORMAT BIT (BIT #12)


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


                    ;CURRENT CYLINDER ADDRESS (RPCC) (#13)
                    ;(EACH BIT IS CALLED BY BIT NUMBER)


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


                    ;RP04 ERROR REGISTER #03 (RPER3) (#15)

000001              PSU=    1                      ;PACK SPEED UNSAFE (BIT #0)
000002              VUF=    2                      ;VELOCITY UNSAFE (BIT #1)
000010              UWR=    10                     ;ANY UNSAFE EXCEPT READ/WRITE (BIT #3)
000040              ACL=    40                     ;AC LOW (BIT #5)
000100              DCL=    100                    ;DC LOW (BIT #6)
040000              SKI=    40000                  ;SEEK INCOMPLETE (BIT #14)
100000              OCYL=   100000                 ;OFF CYLINDER (BIT #15)

                    ;RP05/6 ERROR REGISTER #03 (RPER3) (#15)

000001              DCU=    1                      ;DC UNSAFE (BIT #0)
000002              WAO=    2                      ;WRITE AND OFFSET (BIT #1)
000040              ACL=    40                     ;AC LOW (BIT #5)
000100              DCL=    100                    ;DC LOW (BIT #6)
020000              OPE=    20000                  ;OPERATOR PLUG ERROR (BIT #13)
040000              SKI=    40000                  ;SEEK INCOMPLETE (BIT #14)
100000              OCYL=   100000                 ;OFF CYLINDER ERROR (BIT #15)


                    ;ECC POSITION REGISTER (RPEC1) (#16)
                    ;(EACH BIT IS CALLED BY BIT NUMBER)


                    ;ECC PATTERN REGISTER (RPEC2) (#17)
                    ;(EACH BIT IS CALLED BY BIT NUMBER)


                    .SBTTL  RP04/5/6 DRIVER COMMANDS

000101              RNOP    =       101            ;NO OPERATION
000103              UNLOAD  =       103            ;UNLOAD
000105              SEEK    =       105            ;SEEK
000107              RECAL   =       107            ;RECALIBRATE
000111              DRVCLR  =       111            ;DRIVE CLEAR
000113              RELSE   =       113            ;RELEASE
```

```
        000115          OFFSET  =       115             ;OFFSET
        000117          RTC     =       117             ;RETURN TO CENTER LINE
        000121          READIN  =       121             ;READ IN PRESET
        000123          ACK     =       123             ;PACK ACKNOWLEDGE
        000131          SEARCH  =       131             ;SEARCH
        000141          GETREG  =       141             ;GET REGISTERS
        000143          SETFMT  =       143             ;SET FORMAT (& ECI OR HCI)
        000145          SELDRV  =       145             ;SELECT DRIVE
        000151          WCKD    =       151             ;WRITE CHECK DATA
        000153          WCKHD   =       153             ;WRITE CHECK HEADER & DATA
        000161          WRTDAT  =       161             ;WRITE DATA
        000163          WRTHD   =       163             ;WRITE HEADER & DATA
        000171          RDDAT   =       171             ;READ DATA
        000173          RDHD    =       173             ;READ HEADER & DATA
```

```
                                .SBTTL  TRAP CATCHER

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

                                .SBTTL  STARTING ADDRESS(ES)

000200  000137  004166                  JMP     @#START         ;;JUMP TO STARTING ADDRESS OF PROGRAM
000204  000137  004200                  JMP     @#START1        ;CHANGE THE RH11 UNIBUS ADDRESS
                                                                ;AFTER INITIAL START &
                                                                ;DO NOT INHIBIT PROGRAMMABLE DRIVES
        000220                          .=220
000220  000137  004216                  JMP     @#START3        ;SAME AS 200, EXCEPT, DO NOT INHIBIT
                                                                ;PROGRAMMABLE DRIVES

                                .SBTTL  ACT11 HOOKS

                                ;;*****************************************************************
                                ;HOOKS REQUIRED BY ACT11
        000224                          $SVPC=.                 ;SAVE PC
        000046                          .=46
000046  005662                          $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
        000052                          .=52
000052  040000                          .WORD   40000           ;;2)SET LOC.52 TO 40000
        000224                          .=$SVPC                 ;; RESTORE PC
```

```
    0                                  .SBTTL  COMMON TAGS

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

              001100                          .=1100
       001100                          $CMTAG:                                 ;;START OF COMMON TAGS
       001100 000000                   $PASS:   .WORD   0                      ;;CONTAINS PASS COUNT
       001102    000                   $TSTNM:  .BYTE   0                      ;;CONTAINS THE TEST NUMBER
       001103    000                   $ERFLG:  .BYTE   0                      ;;CONTAINS ERROR FLAG
       001104 000000                   $ICNT:   .WORD   0                      ;;CONTAINS SUBTEST ITERATION COUNT
       001106 000000                   $LPADR:  .WORD   0                      ;;CONTAINS SCOPE LOOP ADDRESS
       001110 000000                   $LPERR:  .WORD   0                      ;;CONTAINS SCOPE RETURN FOR ERRORS
       001112 000000                   $ERTTL:  .WORD   0                      ;;CONTAINS TOTAL ERRORS DETECTED
       001114    000                   $ITEMB:  .BYTE   0                      ;;CONTAINS ITEM CONTROL BYTE
       001115    001                   $ERMAX:  .BYTE   1                      ;;CONTAINS MAX. ERRORS PER TEST
       001116 000000                   $ERRPC:  .WORD   0                      ;;CONTAINS PC OF LAST ERROR INSTRUCTION
       001120 000000                   $GDADR:  .WORD   0                      ;;CONTAINS ADDRESS OF 'GOOD' DATA
       001122 000000                   $BDADR:  .WORD   0                      ;;CONTAINS ADDRESS OF 'BAD' DATA
       001124 000000                   $GDDAT:  .WORD   0                      ;;CONTAINS 'GOOD' DATA
       001126 000000                   $BDDAT:  .WORD   0                      ;;CONTAINS 'BAD' DATA
       001130 000000                            .WORD   0                      ;;RESERVED--NOT TO BE USED
       001132 000000                            .WORD   0
       001134    000                   $AUTOB:  .BYTE   0                      ;;AUTOMATIC MODE INDICATOR
       001135    000                   $INTAG:  .BYTE   0                      ;;INTERRUPT MODE INDICATOR
       001136 000000                            .WORD   0
       001140 177570                   SWR:     .WORD   DSWR                   ;;ADDRESS OF SWITCH REGISTER
       001142 177570                   DISPLAY: .WORD   DDISP                  ;;ADDRESS OF DISPLAY REGISTER
       001144 177560                   $TKS:    177560                         ;;TTY KBD STATUS
       001146 177562                   $TKB:    177562                         ;;TTY KBD BUFFER
       001150 177564                   $TPS:    177564                         ;;TTY PRINTER STATUS REG. ADDRESS
       001152 177566                   $TPB:    177566                         ;;TTY PRINTER BUFFER REG. ADDRESS
       001154    000                   $NULL:   .BYTE   0                      ;;CONTAINS NULL CHARACTER FOR FILLS
       001155    002                   $FILLS:  .BYTE   2                      ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
       001156    012                   $FILLC:  .BYTE   12                     ;;INSERT FILL CHARS. AFTER A "LINE FEED"
       001157    000                   $TPFLG:  .BYTE   0                      ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
       001160    207      377     377  $BELL:   .ASCIZ  <207><377><377>        ;;CODE FOR BELL
       001164    077                   $QUES:   .ASCII  /?/                    ;;QUESTION MARK
       001165    015                   $CRLF:   .ASCII  <15>                   ;;CARRIAGE RETURN
       001166    012      000          $LF:     .ASCIZ  <12>                   ;;LINE FEED
                                       ;;*******************************************************************
```

```
                                        .SBTTL  USER DEFINED TAGS

     001170  176700                     $RPADR: .WORD   176700          ;FIRST ADDRESS OF RH11/RP04/5/6 REGISTERS
     001172  000254                     $RPVEC: .WORD   254             ;RP04 VECTOR ADDRESS
     001174  172540                     $LKCSR: .WORD   172540          ;ADDR OF KW11-P STATUS REGISTER
     001176  172542                     $LKCSB: .WORD   172542          ;ADDR OF KW11-P COUNTER BUFFER
     001200  000104                     $LPVEC: .WORD   104             ;ADDR OF KW11-P VECTOR
     001202  177546                     $LKS:   .WORD   177546          ;ADDR OF KW11-L STATUS REGISTER
     001204  000100                     $LLVEC: .WORD   100             ;ADDR OF KW11-L VECTOR
     001206  177777                     PCLOCK: .WORD   -1              ;'0' IF KW11-P IS ON SYSTEM
     001210  177777                     CLKFLG: .WORD   -1              ;'0' IF A CLOCK IS AVAILABLE
     001212  000074                     HZ:     .WORD   74              ;74 (8) IF 60 HZ SYSTEM; 62 (8) IF 50 HZ SYSTEM
     001214  000000                     STATIN: .WORD   0               ;'TYPE STATISTICS' INDICATOR
     001216  000000                     PACK:   .WORD   0               ;'W ' COMMAND INDICATOR
     001220  000000  000000  000000     DATE:   .WORD   0,0,0,0,0       ;OPERATOR ENTERED DATE
     001232  000000  000000  000000     OPERID: .WORD   0,0,0,0         ;OPERATOR ID
     001242  000000                     DRIVE:  .WORD   0               ;DRIVE # STORAGE: ERRORS 1-5 & 10
     001244  000000                     ATTN:   .WORD   0               ;ATTN REG STORAGE: ERRORS 1-5 & 10
     001246  000000                     UNIT:   .WORD   0               ;DRIVE # STORAGE FOR PRINTOUT
     001250  000000                     MASK:   .WORD   0               ;ERROR RETRY REGISTER MASK
     001252     000     000             RETRY:  .BYTE   0,0             ;ERROR RETRY LIMIT IN THE LOWER BYTE
                                                                        ;RETRY COUNT IN THE UPPER BYTE
     001254  000003                     FAIRNS: .WORD   3               ;MAXIMUM TIME IN QUEUE VALUE
     001256  000000                     LSTAD:  .WORD   0               ;STORE LAST MEMORY ADDRESS HERE
     001260  000000                     CHGADR: .WORD   0               ;CHANGE RH11 UNIBUS ADDRESS FLAG
     001262  000000                     CFLAG:  .WORD   0               ;'CONTROL C' FLAG
     001264  000000                     BADSEC: .WORD   0               ;BAD SECTOR/TRACK FLAG
     001266  000000                     HOUR:   .WORD   0               ;HOUR COUNT STORED HERE (MAXIMUM - 999.)
     001270  000000                     MINUTE: .WORD   0               ;MINUTE'S COUNT STORED HERE
     001272  000000                     SECOND: .WORD   0               ;SECOND'S COUNT STORED HERE
     001274  000000                     SIXTEE: .WORD   0               ;TIMER ROUTINE COUNTER (FOR ONE SECOND)
     001276  177777                     ZROIND: .WORD   -1              ;ZERO INDICATOR FOR THE DATA COMPARE ROUTINE
     001300     000                     FRSTER: .BYTE   0               ;DATA COMPARE ERROR FLAG
                                                                        ;IF > 0, PROCESSING 'DCKER' OR CAN'T MATCH PATTERN
                                                                        ;IF < 0, MISCOMPARSION FOUND
     001301     000                             .BYTE   0               ;MISCOMPARSION OR CAN'T MATCH PATTERN FLAG
                                                                        ;IF < 0, ERROR IN BUFFER
     001302  000000                     SAVER1: .WORD   0               ;SAVE R1 HERE
     001304  000000                     SAVER5: .WORD   0               ;SAVE R5 HERE
     001306  000000                     ERCTR:  .WORD   0               ;NUMBER OF ERRORS
     001310  000000                     LIMIT:  .WORD   0               ;DISPLAY LIMIT
     001312  000000                     CMCNT:  .WORD   0               ;WORD COUNT
     001314  000000                     CMCYL:  .WORD   0               ;CYLINDER ADDRESS
     001316     000                     CMSEC:  .BYTE   0               ;SECTOR ADDRESS
     001317     000                     CMTRK:  .BYTE   0               ;TRACK ADDRESS
     001320  000000                     ECBIT:  .WORD   0               ;ERROR BURST BIT OFFSET
     001322  000000                     ECSEC:  .WORD   0               ;ERROR BURST WORD OFFSET (RELATIVE TO SECTOR)
     001324  000000                     ECMSK0: .WORD   0               ;CORRECTION MASK FOR FIRST ERROR WORD
     001326  000000                     ECMSK1: .WORD   0               ;CORRECTION MASK FOR SECOND ERROR WORD
     001330  000000                     ECWRD:  .WORD   0               ;LOCATION OF FIRST ERROR WORD
     001332  000000                     ECGD:   .WORD   0               ;GOOD DATA, FIRST WORD
     001334  000000                     ECBAD0: .WORD   0               ;BAD DATA, FIRST WORD
     001336  000000                     ECWRD1: .WORD   0               ;LOCATION OF SECOND ERROR WORD
     001340  000000                     ECGD1:  .WORD   0               ;GOOD DATA, SECOND WORD
     001342  000000                     ECBAD1: .WORD   0               ;BAD DATA, SECOND WORD
     001344  000025                     SECLMT: .WORD   21.             ;SECTOR ADDRESS LIMIT
     001346  000022                     TRKLMT: .WORD   18.             ;TRACK ADDRESS LIMIT
     001350  000632                  CYLIMT: .WORD   410.            ;CYLINDER ADDRESS LIMIT FOR RP04/5'S
                                                                     ;(CHANGED TO 814. FOR RP06)

                                     .SBTTL  COMMON PARAMETERS

                                     ;PROGRAM USES THESE PARAMETERS TO DETERMINE REGULAR END OF PASS

     001352  002740                  ENDCN:  .WORD   002740          ;1.875X10^8 WORDS (10) [3X10^9 BITS]
     001354  005455                          .WORD   005455          ;MSW
     001356  143300                  ENDSK:  .WORD   143300          ;3 X 10^6 SEEKS  (LSW)
     001360  000055                          .WORD   55              ;MSW

                                     ;PROGRAM USES THESE PARAMETERS TO DETERMINE Q.V. END OF PASS

     001362  120274                  QVCON:  .WORD   120274          ;2.3437X10^7 WORDS (10)
     001364  000005                          .WORD   5               ;MSW
     001366  134330                  QVSEK:  .WORD   134330          ;3.75 X 10^5 SEEKS (10)
     001370  000005                          .WORD   5               ;MSW

                                     ;THE NUMBERS TO DETERMINE END OF PASS ARE LOADED IN HERE BY THE PROGRAM.
                                     ;THE FIRST TIME THROUGH, THE QV PARAMETERS ARE USED, AFTER THAT THE
                                     ;REGULAR PARAMETERS ARE USED.

     001372  000000                  ENDCON: .WORD   0
     001374  000000                          .WORD   0
     001376  000000                  ENDSEK: .WORD   0
     001400  000000                          .WORD   0

     001402  000001                  PASCNT: .WORD   1               ;NUMBER OF PASSES TO END OF TEST
     001404  000000                  MAXDL:  .WORD   0               ;MAXIMUM DATA TRANFER SIZE IN WORDS
                                                                     ;(FILLED BY PROGRAM AT STARTUP OR BY OPERATOR
                                                                     ;DURING PARAMETER ENTRY DIALOG.)
     001406  000144                  MAXER:  .WORD   100.            ;MAXIMUM ERRORS - 100(10)
     001410  000005  000000          INTRVL: .WORD   5,0             ;FIRST WORD IS THE PERFORMANCE TYPEOUT INTERVAL
                                                                     ;(IN MINUTES).  SECOND WORD IS THE INTERVAL
                                                                     ;COUNTER
                                                                     ;COUNTER.  UPPER BYTE IS VALUE.
     001414  000004                  CMPLMT: .WORD   4               ;NUMBER OF COMPARE ERRORS TYPED OUT
     001416  000001                  FORMAT: .WORD   1               ;IF NOT EQ 0, ALLOW WRITE HEADER & DATA ORDERS
                                                                     ;IF EQ 0, DO NOT ALLOW WRITE HEADER & DATA ORDERS
     001420  000000                  WCSEL:  .WORD   0               ;IF EQ TO 0, GENERATE A RANDOM WORD COUNT
                                                                     ;  FOR THE OPERATION.
                                                                     ;IF NOT EQ TO 0, USE THE VALUE IN 'MAXDL' FOR
                                                                     ;  THE WORD COUNT
     001422  000003                  RATIO:  .WORD   3               ;READ/WRITE RATIO [RANGE 0 - 7]
                                                                     ;0 - 0/8        (READ/WRITE)
                                                                     ;1 - 7/1
                                                                     ;2 - 6/2
                                                                     ;3 - 5/3
                                                                     ;4 - 4/4
                                                                     ;5 - 3/5
                                                                     ;6 - 2/6
                                                                     ;7 - 1/7
     001424  000001                  AUTOCK: .WORD   1               ;IF NOT EQ 0, DO AN APPROPRITE WRITE
                                                                     ;  CHECK AFTER EACH WRITE ORDER.
                                                                     ;IF EQ 0, SELECT WRITE CHECK ORDERS
                                                                     ;  RANDOMLY.
```

```
        001426  000001                  NOTPRT: .WORD   1               ;IF EQ 1, DO NOT PRINT DATA ERROR MESSAGES
                                                                        ;  ASSOCIATED WITH OPERATOR SPECIFIED
                                                                        ;  BAD PACK AREAS.
                                                                        ;IF NOT EQ 0, PRINT ERROR MESSAGES RELATING TO
                                                                        ;  THESE AREAS.
        001430  000001                  ENDET:  .WORD   1               ;IF NOT EQ 0, END OF PASS DETERMINED
                                                                        ;  BY THE 'WORDS READ' COUNT.
                                                                        ;IF EQ 0, END OF PASS DETERMINED
                                                                        ;  BY THE SEEK COUNT.

                                        .SBTTL  VALUES FOR FIRST OPERATION

        001432  000010                  BEGPAT: .WORD   10              ;STARTING PATTERN CODE [RANGE 1 - 17 (OCTAL)]
        001434  000005                  BEGCOD: .WORD   5               ;STARTING COMMAND CODE [RANGE 0 - 5]
                                                                        ;0 = WRITE CHECK DATA ('WCKD')
                                                                        ;1 = WRITE CHECK HEADER & DATA ('WCHKHD')
                                                                        ;2 = WRITE DATA ('WRTDAT')
                                                                        ;3 = WRITE HEADER & DATA ('WRTHD')
                                                                        ;4 = READ DATA ('RDDAT')
                                                                        ;5 = READ HEADER & DATA ('RDHD')
        001436  000404                  BEGSIZ: .WORD   404             ;STARTING RECORD SIZE [RANGE 4 - MAXMEM]
                                                                        ;NOTE:  THE SIZE MUST BE AT LEAST 4 IF
                                                                        ;WRITE DATA OR READ DATA; THE SIZE MUST
                                                                        ;BE AT LEAST 8 IF WRITE HEADER AND
                                                                        ;DATA OR READ HEADER AND DATA.
                                                                        ;IF THE SIZE IS GREATER THAN 1 SECTOR, THE
                                                                        ;SIZE MUST ALLOW FOR OVERLAPPING 4 OR 8
                                                                        ;WORDS INTO THE LAST SECTOR USED.
```

```
                                         .SBTTL  TABLES, CONSTANTS, AND VARIABLE LOCATIONS

001440  000000  000000  000000  ORDERQ: .WORD   0,0,0,0,0,0,0,0,0  ;LIST OF DRIVES PERFORMING COMMANDS
001462  000000                  ASNLST: .WORD   0                  ;A BIT SET IS AN ASSIGNED DRIVE
001464  000000  000000  000000  DUNIT:  .WORD   0,0,0,0,0,0,0,0,0          ;ADDRESSES OF DRIVES TO BE DEASSIGNED
001506  000000  000000  000000  NEWUNT: .WORD   0,0,0,0,0,0,0,0,0          ;ADDRESSES OF NEWLY ASSIGNED DRIVES
001530  000000  000000  000000  AVAIL:  .WORD   0,0,0,0,0,0,0,0,0          ;LIST OF DRIVES WAITING FOR BUFFERS/PARAMETERS
001552  000000  000000  000000  WAIT:   .WORD   0,0,0,0,0,0,0,0,0          ;LIST OF DRIVES WAITING FOR BUFFERS
001574  000000  000000  000000  PARQ:   .WORD   0,0,0,0,0,0,0,0,0          ;LIST OF DRIVES WAITING FOR NEXT PARAMETERS
001616  000000                  BUFTBL: .WORD   0                  ;BUFFER ALLOCATION TABLE ENTRY COUNT
        000024                          .REPT   24
001620  000000  000000                  .WORD   0,0
001624  000000  000000                  .WORD   0,0
001630  000000  000000                  .WORD   0,0
001634  000000  000000                  .WORD   0,0
001640  000000  000000                  .WORD   0,0
001644  000000  000000                  .WORD   0,0
001650  000000  000000                  .WORD   0,0
001654  000000  000000                  .WORD   0,0
001660  000000  000000                  .WORD   0,0
001664  000000  000000                  .WORD   0,0
001670  000000  000000                  .WORD   0,0
001674  000000  000000                  .WORD   0,0
001700  000000  000000                  .WORD   0,0
001704  000000  000000                  .WORD   0,0
001710  000000  000000                  .WORD   0,0
001714  000000  000000                  .WORD   0,0
001720  000000  000000                  .WORD   0,0
001724  000000  000000                  .WORD   0,0
001730  000000  000000                  .WORD   0,0
001734  000000  000000                  .WORD   0,0

001740  043110                  BLKADR: .WORD   DRIVE0             ;ADDRESS OF THE BLOCK FOR DRIVE 0
001742  043414                          .WORD   DRIVE1             ;ADDRESS OF THE BLOCK FOR DRIVE 1
001744  043720                          .WORD   DRIVE2             ;ADDRESS OF THE BLOCK FOR DRIVE 2
001746  044224                          .WORD   DRIVE3             ;ADDRESS OF THE BLOCK FOR DRIVE 3
001750  044530                          .WORD   DRIVE4             ;ADDRESS OF THE BLOCK FOR DRIVE 4
001752  045034                          .WORD   DRIVE5             ;ADDRESS OF THE BLOCK FOR DRIVE 5
001754  045340                          .WORD   DRIVE6             ;ADDRESS OF THE BLOCK FOR DRIVE 6
001756  045644                          .WORD   DRIVE7             ;ADDRESS OF THE BLOCK FOR DRIVE 7

001760     151                  COMTBL: .BYTE   WCKD               ;WRITE CHECK DATA
001761     153                          .BYTE   WCKHD              ;WRITE CHECK HEADER AND DATA
001762     161                          .BYTE   WRTDAT             ;WRITE DATA
001763     163                          .BYTE   WRTHD              ;WRITE HEADER AND DATA
001764     171                          .BYTE   RDDAT              ;READ DATA
001765     173                          .BYTE   RDHD               ;READ HEADER AND DATA

001766     002                  OPTBL:  .BYTE   2                  ;UNLOAD
001767     004                          .BYTE   4                  ;SEEK
```

```
      001770   006                            .BYTE   6              ;RECAL
      001771   010                            .BYTE   10             ;DRIVE CLEAR
      001772   012                            .BYTE   12             ;RELEASE
      001773   014                            .BYTE   14             ;OFFSET
      001774   016                            .BYTE   16             ;RETURN TO CENTERLINE
      001775   020                            .BYTE   20             ;READIN PRESET
      001776   022                            .BYTE   22             ;PACK ACKNOWLEDGE
      001777   030                            .BYTE   30             ;SEARCH
      002000   050                            .BYTE   50             ;WRITE CHECK DATA
      002001   052                            .BYTE   52             ;WRITE CHECK HEADER AND DATA
      002002   060                            .BYTE   60             ;WRITE DATA
      002003   062                            .BYTE   62             ;WRITE HEADER AND DATA
      002004   070                            .BYTE   70             ;READ DATA
      002005   072                            .BYTE   72             ;READ HEADER AND DATA
      002006   377                            .BYTE   -1             ;TERMINATOR

                                     .EVEN

      002010   125     116     114   MNTBL:   .ASCIZ  /UNLOAD /
      002020   123     105     105            .ASCIZ  /SEEK   /
      002030   122     105     103            .ASCIZ  /RECAL  /
      002040   104     122     126            .ASCIZ  /DRVCLR /
      002050   122     105     114            .ASCIZ  /RELSE  /
      002060   117     106     106            .ASCIZ  /OFFSET /
      002070   122     124     103            .ASCIZ  /RTC    /
      002100   122     105     101            .ASCIZ  /READIN /
      002110   120     101     103            .ASCIZ  /PACK   /
      002120   123     105     101            .ASCIZ  /SEARCH /
      002130   127     103     113            .ASCIZ  /WCKD   /
      002140   127     103     113            .ASCIZ  /WCKHD  /
      002150   127     122     124            .ASCIZ  /WRTDAT /
      002160   127     122     124            .ASCIZ  /WRTHD  /
      002170   122     104     104            .ASCIZ  /RDDAT  /
      002200   122     104     110            .ASCIZ  /RDHD   /
      002210   116     117     116            .ASCIZ  /NONE   /

      002220   000                   OFFCOD:  .BYTE   0              ;OFFSET CODE TABLE
      002221   010                            .BYTE   10             ;+200 U INCHES
      002222   210                            .BYTE   210            ;-200 U INCHES
      002223   020                            .BYTE   20             ;+400 U INCHES
      002224   220                            .BYTE   220            ;-400 U INCHES
      002225   030                            .BYTE   30             ;+600 U INCHES
      002226   230     000                    .BYTE   230,0          ;-600 U INCHES, TERMINATOR
      002230   020                            .BYTE   20             ;+400 U INCHES
      002231   220                            .BYTE   220            ;-400 U INCHES
      002232   040                            .BYTE   40             ;+800 U INCHES
      002233   240                            .BYTE   240            ;-800 U INCHES
      002234   060                            .BYTE   60             ;+1200 U INCHES
      002235   260     000                    .BYTE   260,0          ;-1200 U INCHES, TERMINATOR
                                     .EVEN

      002240 002274                  OFMTBL:  .WORD   OFMSG0         ;1ST OFFSET MESSAGE
      002242 002327                           .WORD   OFMSG1         ;2ND OFFSET MESSAGE
      002244 002363                           .WORD   OFMSG2         ;3RD OFFSET MESSAGE
      002246 002417                           .WORD   OFMSG3         ;4TH OFFSET MESSAGE
      002250 002453                           .WORD   OFMSG4         ;5TH OFFSET MESSAGE
      002252 002507                           .WORD   OFMSG5         ;6TH OFFSET MESSAGE
```

```
        002254  002543                          .WORD   OFMSG6          ;7TH OFFSET MESSAGE
        002256  002274                          .WORD   OFMSG0          ;1ST OFFSET MESSAGE
        002260  002417                          .WORD   OFMSG3          ;4TH OFFSET MESSAGE
        002262  002453                          .WORD   OFMSG4          ;5TH OFFSET MESSAGE
        002264  002577                          .WORD   OFMSG7          ;8TH OFFSET MESSAGE
        002266  002633                          .WORD   OFMSG8          ;9TH OFFSET MESSAGE
        002270  002667                          .WORD   OFMSG9          ;10TH OFFSET MESSAGE
        002272  002724                          .WORD   OFMSGA          ;11TH OFFSET MESSAGE

        002274     101     106     124  OFMSG0: .ASCIZ  /AFTER RETRY WITHOUT OFFSET/
        002327     101     124     040  OFMSG1: .ASCIZ  /AT OFFSET +200 MICRO-INCHES/
        002363     101     124     040  OFMSG2: .ASCIZ  /AT OFFSET -200 MICRO-INCHES/
        002417     101     124     040  OFMSG3: .ASCIZ  /AT OFFSET +400 MICRO-INCHES/
        002453     101     124     040  OFMSG4: .ASCIZ  /AT OFFSET -400 MICRO-INCHES/
        002507     101     124     040  OFMSG5: .ASCIZ  /AT OFFSET +600 MICRO-INCHES/
        002543     101     124     040  OFMSG6: .ASCIZ  /AT OFFSET -600 MICRO-INCHES/
        002577     101     124     040  OFMSG7: .ASCIZ  /AT OFFSET +800 MICRO-INCHES/
        002633     101     124     040  OFMSG8: .ASCIZ  /AT OFFSET -800 MICRO-INCHES/
        002667     101     124     040  OFMSG9: .ASCIZ  /AT OFFSET +1200 MICRO-INCHES/
        002724     101     124     040  OFMSGA: .ASCIZ  /AT OFFSET -1200 MICRO-INCHES/
                                        .EVEN
```

```
                                .SBTTL  DATA PATTERNS

002762  000000                  STNDAT: .WORD   0                       ;STANDARD DATA PATTERN POINTER TABLE
002764  003066                          .WORD   DATA1
002766  003126                          .WORD   DATA1+40
002770  003166                          .WORD   DATA1+100
002772  003226                          .WORD   DATA1+140
002774  003266                          .WORD   DATA1+200
002776  003326                          .WORD   DATA1+240
003000  003366                          .WORD   DATA1+300
003002  003426                          .WORD   DATA1+340
003004  003466                          .WORD   DATA1+400
003006  003526                          .WORD   DATA1+440
003010  003566                          .WORD   DATA1+500
003012  003626                          .WORD   DATA1+540
003014  003666                          .WORD   DATA1+600
003016  003726                          .WORD   DATA1+640
003020  003766                          .WORD   DATA1+700
003022  003026                          .WORD   DATA0                   ;ZEROES
003024  003730                          .WORD   DATA1+642               ;ONES

003026  000000                  DATA0:  .WORD   0                       ;DUMMY DATA PATTERN
        000017                          .REPT   17
003030  000000                          .WORD   0
003032  000000                          .WORD   0
003034  000000                          .WORD   0
003036  000000                          .WORD   0
003040  000000                          .WORD   0
003042  000000                          .WORD   0
003044  000000                          .WORD   0
003046  000000                          .WORD   0
003050  000000                          .WORD   0
003052  000000                          .WORD   0
003054  000000                          .WORD   0
003056  000000                          .WORD   0
003060  000000                          .WORD   0
003062  000000                          .WORD   0
003064  000000                          .WORD   0

003066  000001                  DATA1:  .WORD   000001  ;STANDARD PATTERN 1
003070  000003                          .WORD   000003
003072  000007                          .WORD   000007
003074  000017                          .WORD   000017
003076  000037                          .WORD   000037
003100  000077                          .WORD   000077
003102  000177                          .WORD   000177
003104  000377                          .WORD   000377
003106  000777                          .WORD   000777
003110  001777                          .WORD   001777
003112  003777                          .WORD   003777
003114  007777                          .WORD   007777
003116  017777                          .WORD   017777
003120  037777                          .WORD   037777
003122  077777                          .WORD   077777
003124  177777                          .WORD   177777

003126  177776                          .WORD   177776  ;STANDARD PATTERN 2
```

```
        003130  177774                          .WORD   177774
        003132  177770                          .WORD   177770
        003134  177760                          .WORD   177760
        003136  177740                          .WORD   177740
        003140  177700                          .WORD   177700
        003142  177600                          .WORD   177600
        003144  177400                          .WORD   177400
        003146  177000                          .WORD   177000
        003150  176000                          .WORD   176000
        003152  174000                          .WORD   174000
        003154  170000                          .WORD   170000
        003156  160000                          .WORD   160000
        003160  140000                          .WORD   140000
        003162  100000                          .WORD   100000
        003164  000000                          .WORD   000000

        003166  000000                          .WORD   000000  ;STANDARD PATTERN 3
        003170  000000                          .WORD   000000
        003172  000000                          .WORD   000000
        003174  177777                          .WORD   177777
        003176  177777                          .WORD   177777
        003200  177777                          .WORD   177777
        003202  000000                          .WORD   000000
        003204  000000                          .WORD   000000
        003206  177777                          .WORD   177777
        003210  177777                          .WORD   177777
        003212  000000                          .WORD   000000
        003214  177777                          .WORD   177777
        003216  000000                          .WORD   000000
        003220  177777                          .WORD   177777
        003222  000000                          .WORD   000000
        003224  177777                          .WORD   177777

        003226  000000                          .WORD   000000  ;STANDARD PATTERN 4
        003230  010421                          .WORD   010421
        003232  021042                          .WORD   021042
        003234  031463                          .WORD   031463
        003236  042104                          .WORD   042104
        003240  052525                          .WORD   052525
        003242  063146                          .WORD   063146
        003244  073567                          .WORD   073567
        003246  104210                          .WORD   104210
        003250  114631                          .WORD   114631
        003252  125252                          .WORD   125252
        003254  135673                          .WORD   135673
        003256  146314                          .WORD   146314
        003260  156735                          .WORD   156735
        003262  167356                          .WORD   167356
        003264  177777                          .WORD   177777

        003266  052525                          .WORD   052525  ;STANDARD PATTERN 5
        003270  052525                          .WORD   052525
        003272  052525                          .WORD   052525
        003274  125252                          .WORD   125252
        003276  125252                          .WORD   125252
        003300  125252                          .WORD   125252
        003302  052525                          .WORD   052525
```

```
    003304  052525                          .WORD   052525
    003306  125252                          .WORD   125252
    003310  125252                          .WORD   125252
    003312  052525                          .WORD   052525
    003314  125252                          .WORD   125252
    003316  052525                          .WORD   052525
    003320  125252                          .WORD   125252
    003322  052525                          .WORD   052525
    003324  125252                          .WORD   125252

    003326  007417                          .WORD   007417  ;STANDARD PATTERN 6
    003330  007417                          .WORD   007417
    003332  007417                          .WORD   007417
    003334  170360                          .WORD   170360
    003336  170360                          .WORD   170360
    003340  170360                          .WORD   170360
    003342  007417                          .WORD   007417
    003344  007417                          .WORD   007417
    003346  170360                          .WORD   170360
    003350  170360                          .WORD   170360
    003352  007417                          .WORD   007417
    003354  170360                          .WORD   170360
    003356  007417                          .WORD   007417
    003360  170360                          .WORD   170360
    003362  007417                          .WORD   007417
    003364  170360                          .WORD   170360

    003366  026455                          .WORD   026455  ;STANDARD PATTERN 7
    003370  026455                          .WORD   026455
    003372  026455                          .WORD   026455
    003374  151322                          .WORD   151322
    003376  151322                          .WORD   151322
    003400  151322                          .WORD   151322
    003402  026455                          .WORD   026455
    003404  026455                          .WORD   026455
    003406  151322                          .WORD   151322
    003410  151322                          .WORD   151322
    003412  026455                          .WORD   026455
    003414  151322                          .WORD   151322
    003416  026455                          .WORD   026455
    003420  151322                          .WORD   151322
    003422  026455                          .WORD   026455
    003424  151322                          .WORD   151322

    003426  165555                          .WORD   165555  ;STANDARD PATTERN 8
    003430  133333                          .WORD   133333
    003432  165555                          .WORD   165555
    003434  133333                          .WORD   133333
    003436  165555                          .WORD   165555
    003440  133333                          .WORD   133333
    003442  165555                          .WORD   165555
    003444  133333                          .WORD   133333
    003446  165555                          .WORD   165555
    003450  133333                          .WORD   133333
    003452  165555                          .WORD   165555
    003454  133333                          .WORD   133333
    003456  165555                          .WORD   165555
```

```
003460  133333                   .WORD   133333
003462  165555                   .WORD   165555
003464  133333                   .WORD   133333

003466  000001                   .WORD   000001  ;STANDARD PATTERN 9
003470  000002                   .WORD   000002
003472  000004                   .WORD   000004
003474  000010                   .WORD   000010
003476  000020                   .WORD   000020
003500  000040                   .WORD   000040
003502  000100                   .WORD   000100
003504  000200                   .WORD   000200
003506  000400                   .WORD   000400
003510  001000                   .WORD   001000
003512  002000                   .WORD   002000
003514  004000                   .WORD   004000
003516  010000                   .WORD   010000
003520  020000                   .WORD   020000
003522  040000                   .WORD   040000
003524  100000                   .WORD   100000

003526  177776                   .WORD   177776  ;STANDARD PATTERN 10
003530  177775                   .WORD   177775
003532  177773                   .WORD   177773
003534  177767                   .WORD   177767
003536  177757                   .WORD   177757
003540  177737                   .WORD   177737
003542  177677                   .WORD   177677
003544  177577                   .WORD   177577
003546  177377                   .WORD   177377
003550  176777                   .WORD   176777
003552  175777                   .WORD   175777
003554  173777                   .WORD   173777
003556  167777                   .WORD   167777
003560  157777                   .WORD   157777
003562  137777                   .WORD   137777
003564  077777                   .WORD   077777

003566  172666                   .WORD   172666  ;STANDARD PATTERN 11
003570  155555                   .WORD   155555
003572  172666                   .WORD   172666
003574  155555                   .WORD   155555
003576  172666                   .WORD   172666
003600  155555                   .WORD   155555
003602  172666                   .WORD   172666
003604  155555                   .WORD   155555
003606  172666                   .WORD   172666
003610  155555                   .WORD   155555
003612  172666                   .WORD   172666
003614  155555                   .WORD   155555
003616  172666                   .WORD   172666
003620  155555                   .WORD   155555
003622  172666                   .WORD   172666
003624  155555                   .WORD   155555

003626  077777                   .WORD   077777  ;STANDARD PATTERN 12
003630  137777                   .WORD   137777
```

```
        003632  157777                          .WORD   157777
        003634  167777                          .WORD   167777
        003636  173777                          .WORD   173777
        003640  175777                          .WORD   175777
        003642  176777                          .WORD   176777
        003644  177377                          .WORD   177377
        003646  177577                          .WORD   177577
        003650  177677                          .WORD   177677
        003652  177737                          .WORD   177737
        003654  177757                          .WORD   177757
        003656  177767                          .WORD   177767
        003660  177773                          .WORD   177773
        003662  177775                          .WORD   177775
        003664  177776                          .WORD   177776

        003666  153333                          .WORD   153333  ;STANDARD PATTERN 13
        003670  066667                          .WORD   066667
        003672  153333                          .WORD   153333
        003674  066667                          .WORD   066667
        003676  153333                          .WORD   153333
        003700  066667                          .WORD   066667
        003702  153333                          .WORD   153333
        003704  066667                          .WORD   066667
        003706  153333                          .WORD   153333
        003710  066667                          .WORD   066667
        003712  153333                          .WORD   153333
        003714  066667                          .WORD   066667
        003716  153333                          .WORD   153333
        003720  066667                          .WORD   066667
        003722  153333                          .WORD   153333
        003724  066667                          .WORD   066667

        003726  000000                          .WORD   000000  ;STANDARD PATTERN 14
        003730  177777                          .WORD   177777
        003732  177777                          .WORD   177777
        003734  177777                          .WORD   177777
        003736  177777                          .WORD   177777
        003740  177777                          .WORD   177777
        003742  177777                          .WORD   177777
        003744  177777                          .WORD   177777
        003746  177777                          .WORD   177777
        003750  177777                          .WORD   177777
        003752  177777                          .WORD   177777
        003754  177777                          .WORD   177777
        003756  177777                          .WORD   177777
        003760  177777                          .WORD   177777
        003762  177777                          .WORD   177777
        003764  177777                          .WORD   177777

        003766  177777                          .WORD   177777  ;STANDARD PATTERN 15
        003770  000000                          .WORD   000000
        003772  000000                          .WORD   000000
        003774  000000                          .WORD   000000
        003776  000000                          .WORD   000000
        004000  000000                          .WORD   000000
        004002  000000                          .WORD   000000
        004004  000000                          .WORD   000000
        004006  000000                          .WORD   000000
        004010  000000                          .WORD   000000
        004012  000000                          .WORD   000000
        004014  000000                          .WORD   000000
        004016  000000                          .WORD   000000
        004020  000000                          .WORD   000000
        004022  000000                          .WORD   000000
        004024  000000                          .WORD   000000
```

```
                                .SBTTL  ERROR POINTER TABLE

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

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

        004026                  $ERRTB:

                                ;ERROR 1

 004026  046240                         EM1                     ;RH11 INTERRUPT OCCURRED (RPAS = 0)
 004030  050700                         DH1
 004032  051332                         DT1
 004034  000000                         0

                                ;ERROR 2

 004036  046303                         EM2                     ;UNEXPECTED ATTENTION OCCURRED
 004040  050705                         DH2
 004042  051336                         DT2
 004044  000000                         0

                                ;ERROR 3

 004046  046341                         EM3                     ;MASSBUS PARITY ERROR (MCPE=1)
 004050  050762                         DH3
 004052  051354                         DT3
 004054  000000                         0

                                ;ERROR 4

 004056  046377                         EM4                     ;MASSBUS PARITY ERROR (PAR=1)
 004060  051010                         DH4
 004062  051364                         DT4
 004064  000000                         0

                                ;ERROR 5

 004066  046434                         EM5                     ;ADDRESS PLUG BIT CHANGED
 004070  050705                         DH2
 004072  051336                         DT2
 004074  000000                         0

                                ;ERROR 6

 004076  046470                         EM6                     ;RH11 DIDN'T RESPOND TO ADDRESSING
 004100  051047                         DH6
 004102  051376                         DT6
 004104  000000                         0
```

```
                                        ;THIS ROUTINE HANDLES UNEXPECTED TIMEOUTS

004106  011600                          BADTMO: MOV     (SP),R0         ;SAVE PC WHERE THE TIME OUT OCCURED
004110  005740                                  TST     -(R0)           ;ADJUST PC -2
004112  022626                                  CMP     (SP)+,(SP)+     ;RESTORE STACK POINTER
004114  104401  004122                          TYPE    ,65$            ;;TYPE ASCIZ STRING
004120  000417                                  BR      64$             ;;GET OVER THE ASCIZ
                                        ;;65$:  .ASCIZ  <CRLF>/UNEXPECTED BUS TIMEOUT, PC=/
004160                                  64$:
004160  010046                                  MOV     R0,-(SP)        ;SETUP FOR TYPING OUT PC
004162  104402                                  TYPOC
004164  000240                                  NOP                     ;PUT 'HALT(0)' INSTRUCTION HERE IF YOU WISH
                                                                        ;TO STOP ON UNEXPECTED TIMEOUT.


                                        .SBTTL  START OF PROGRAM

004166  005037  001260                  START:  CLR     CHGADR          ;CLEAR THE RH11 ADDRESS CHANGE FLAG
004172  005037  035336                          CLR     TSTPGM          ;DISABLE PROGRAMMABLE DRIVES
004176  000414                                  BR      START2          ;START THE PROGRAM

004200  012737  177777  001260          START1: MOV     #-1,CHGADR      ;SET RH11 ADDRESS CHANGE FLAG
004206  012737  000001  035336                  MOV     #1,TSTPGM       ;ENABLE PROGRAMMABLE DRIVES
004214  000405                                  BR      START2          ;START THE PROGRAM

004216  012737  000001  035336          START3: MOV     #1,TSTPGM       ;ENABLE PROGRAMMABLE DRIVES
004224  005037  001260                          CLR     CHGADR          ;CLEAR THE RH11 ADDRESS CHANGE FLAG

004230  005227  000000                  START2: INC     #0              ;TTY LOOP, WAIT FOR INCREMENT
004234  001375                                  BNE     .-4             ;OF WORD
004236  000005                                  RESET                   ;CLEAR THE WORLD


                                        .SBTTL  INITIALIZE THE COMMON TAGS
                                        ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
004240  012706  001100                          MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
004244  005026                                  CLR     (R6)+           ;;CLEAR MEMORY LOCATION
004246  022706  001140                          CMP     #SWR,R6         ;;DONE?
004252  001374                                  BNE     .-6             ;;LOOP BACK IF NO
004254  012706  001100                          MOV     #STACK,SP       ;;SETUP THE STACK POINTER
                                        ;;INITIALIZE A FEW VECTORS
004260  012737  031712  000030                  MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
004266  012737  000340  000032                  MOV     #340,@#EMTVEC+2 ;;LEVEL 7
004274  012737  034246  000034                  MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
004302  012737  000340  000036                  MOV     #340,@#TRAPVEC+2;LEVEL 7
004310  012737  176543  033632                  MOV     #176543,$HINUM  ;;PRIME THE RANDOM NUMBER GENERATOR
004316  012737  123456  033634                  MOV     #123456,$LONUM  ;;BOTH HIGH AND LOW WORDS
                                        ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
                                        ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
004324  013746  000004                          MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
004330  012737  004364  000004                  MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
004336  012737  177570  001140                  MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
004344  012737  177570  001142                  MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
004352  022777  177777  174560                  CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
004360  001012                                  BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                                        ;;AND  THE HARDWARE SWR IS NOT = -1
004362  000403                                  BR      65$             ;;BRANCH IF NO TIMEOUT
004364  012716  004372                  64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
004370  000002                                  RTI
```

INITIALIZE THE COMMON TAGS

```
 004372 012737 000176 001140 65$:   MOV    #SWREG,SWR         ;;POINT TO SOFTWARE SWR
 004400 012737 000174 001142        MOV    #DISPREG,DISPLAY
 004406 012637 000004        66$:   MOV    (SP)+,@#ERRVEC     ;;RESTORE ERROR VECTOR

                             ;SETUP ''TIMEOUT'' TRAP VECTOR FOR UNEXPECTED BUS TIMEOUTS
 004412 012737 004106 000004        MOV    #BADTMO,ERRVEC     ;SETUP FOR UNEXPECTED TIMEOUT
 004420 012737 000300 000006        MOV    #PR6,ERRVEC+2      ;LEVEL 6


                             .SBTTL TYPE PROGRAM NAME
                             ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
 004426 005227 177777               INC    #-1                ;;FIRST TIME?
 004432 001031                      BNE    67$                ;;BRANCH IF NO
 004434 104401 004442               TYPE   ,68$               ;;TYPE ASCIZ STRING
 004440 000426                      BR     67$                ;;GET OVER THE ASCIZ
                             ;;68$: .ASCIZ  <CRLF>@CZRJDEO - RP04/5/6 MULTI DRIVE LOGIC TEST@<CRLF>
 004516                      67$:
                             .SBTTL GET VALUE FOR SOFTWARE SWITCH REGISTER
 004516 005737 000042               TST    @#42               ;;ARE WE RUNNING UNDER XXDP/ACT?
 004522 001006                      BNE    69$                ;;BRANCH IF YES
 004524 023727 001140 000176        CMP    SWR,#SWREG         ;;SOFTWARE SWITCH REG SELECTED?
 004532 001005                      BNE    70$                ;;BRANCH IF NO
 004534 104406                      GTSWR                     ;;GET SOFT-SWR SETTINGS
 004536 000403                      BR     70$
 004540 112737 000001 001134 69$:   MOVB   #1,$AUTOB          ;;SET AUTO-MODE INDICATOR
 004546                      70$:

 004546 013737 001352 001372        MOV    ENDCN,ENDCON       ;SET UP FOR NORMAL PASS
 004554 013737 001354 001374        MOV    ENDCN+2,ENDCON+2
 004562 013737 001356 001376        MOV    ENDSK,ENDSEK
 004570 013737 001360 001400        MOV    ENDSK+2,ENDSEK+2
 004576 012737 000240 000032        MOV    #240,EMTVEC+2      ;CHANGE EMT PRIORITY TO 5
 004604 012737 000240 000036        MOV    #240,TRAPVEC+2     ;CHANGE TRAP PRIORITY TO 5
 004612 005227 177777               INC    #-1                ;FIRST START ?
 004616 001025                      BNE    3$                 ;BR IF NOT
 004620 023737 000042 000046        CMP    @#42,@#46          ;ACT11 AUTOMATIC MODE?
 004626 001003                      BNE    1$                 ;BRANCH IF NO
 004630 012737 000001 035336        MOV    #1,TSTPGM          ;ENABLE PROGRAMMABLE DRIVES
 004636 005737 035336        1$:    TST    TSTPGM             ;CAN WE USE PROGRAMMABLE DRIVES?
 004642 001402                      BEQ    2$                 ;BRANCH IF NO
 004644 104401 053366               TYPE   ,USE               ;TYPE MSG

 004650 005737 000042        2$:    TST    @#42               ;AUTO ACCEPT OR CHAIN MODE ?
 004654 001006                      BNE    3$                 ;BR IF EITHER
 004656 122737 000011 000041        CMPB   #11,@#41           ;LOADED FROM AN RP04/5/6 ?
 004664 001002                      BNE    3$                 ;BR IF NOT
 004666 104401 056712               TYPE   ,LOADRV            ;INSTRUCT THE OPERATOR ON HOW TO TEST DRIVE 0

 004672 004737 030400        3$:    JSR    PC,$TKINT          ;TURN ON THE KEYBOARD INTERRUPT
                             .SBTTL GET VALUE FOR SOFTWARE SWITCH REGISTER
 004676 005737 000042               TST    @#42               ;;ARE WE RUNNING UNDER XXDP/ACT?
 004702 001006                      BNE    71$                ;;BRANCH IF YES
 004704 023727 001140 000176        CMP    SWR,#SWREG         ;;SOFTWARE SWITCH REG SELECTED?
 004712 001005                      BNE    72$                ;;BRANCH IF NO
 004714 104406                      GTSWR                     ;;GET SOFT-SWR SETTINGS
 004716 000403                      BR     72$
 004720 112737 000001 001134 71$:   MOVB   #1,$AUTOB          ;;SET AUTO-MODE INDICATOR
 004726                      72$:
```

```
004726  005227  177777                    INC     #-1               ;FIRST START ?
004732  001015                            BNE     4$                ;BR IF NOT
004734  004737  056302                    JSR     PC,BUSADR         ;CHECK RH11 BUS ADDRESS
004740  013737  001170  034502            MOV     $RPADR,RPADR      ;RH11 ADDRESS
004746  013737  001172  034504            MOV     $RPVEC,RPVEC      ;RH11 VECTOR ADDRESS
004754  005737  000042                    TST     @#42              ;ACT-11 AUTO OR CHAIN MODE?
004760  001002                            BNE     4$                ;BRANCH IF EITHER, SKIP
                                                                    ;DATE & OPERATOR ID INPUT
004762  004737  055764                    JSR     PC,OPRDAT         ;GET THE DATE AND OPERATOR ID
004766  005037  001214           4$:      CLR     STATIN            ;CLEAR PERFORMANCE SUMMARY TYPEOUT FLAG
004772  012705  001440                    MOV     #ORDERQ,R5        ;START OF AREA TO CLEAR
004776  005025                   5$:      CLR     (R5)+
005000  022705  001740                    CMP     #BLKADR,R5        ;LOOK FOR END OF CLEAR AREA
005004  001374                            BNE     5$                ;BR IF NOT FINISHED
005006  012706  001100                    MOV     #STACK,SP         ;SETUP THE STACK POINTER
005012  005037  177776                    CLR     PS                ;CLEAR THE PROCESSOR STATUS WORD
005016  013737  001212  001274            MOV     HZ,SIXTEE         ;1/60 TH OR 1/50 TH SECOND COUNTER VALUE
005024  005037  001266                    CLR     HOUR              ;CLEAR THE HOUR'S COUNTER
005030  005037  001270                    CLR     MINUTE            ;CLEAR THE MINUTE'S COUNTER
005034  005037  001272                    CLR     SECOND            ;CLEAR THE SECOND'S COUNTER
005040  005037  001412                    CLR     INTRVL+2          ;CLEAR INTERVAL COUNTER
005044  005037  001216                    CLR     PACK              ;CLEAR THE 'R' OR 'W' COMMAND FLAG
005050  005037  001262                    CLR     CFLAG             ;CLEAR THE 'CONTROL C' FLAG
005054  042737  170000  001406            BIC     #170000,MAXER     ;MAKE SURE ERROR LIMITS ARE NOT TOO HIGH

                                 ;ROUTINE TO DETERMINE BUFFER AREA SIZE

005062  005227  177777           SIZMEM:  INC     #-1               ;SEE IF TIME TO SIZE MEMORY
005066  001005                            BNE     1$                ;BR IF NOT
005070  004737  056142                    JSR     PC,$SIZE          ;SEE HOW MUCH MEMORY ON SYSTEM
005074  013737  056272  001256            MOV     $LSTAD,LSTAD      ;SAVE THE LAST ADDRESS
005102  012737  000001  001616   1$:      MOV     #1,BUFTBL         ;LOAD NUMBER OF BUFFERS
005110  012737  055764  001620            MOV     #ENDPGM,BUFTBL+2  ;STARTING ADDRESS OF BUFFER
005116  013737  001256  001622            MOV     LSTAD,BUFTBL+4    ;LAST ADDR TO BUFFER ALLOCATION TABLE
005124  162737  055764  001622            SUB     #ENDPGM,BUFTBL+4  ;SUBTRACT PROGRAM SPACE
005132  000241                            CLC                       ;CLEAR THE 'C' BIT
005134  006037  001622                    ROR     BUFTBL+4          ;CONVERT TO WORD COUNT
005140  162737  000144  001622            SUB     #100.,BUFTBL+4    ;SAVE ROOM FOR THE 'ABS' LOADER
005146  023727  001256  100000            CMP     LSTAD,#100000     ;16K ON THE SYSTEM ?
005154  103406                            BLO     3$                ;BR IF YES
005156  105737  000041                    TSTB    41                ;SEE WHO LOADED THE PROGRAM
005162  001403                            BEQ     3$                ;BR IF LOADED BY PAPER TAPE
005164  162737  003000  001622            SUB     #1536.,BUFTBL+4   ;SUBTRACT 'XXDP' LOADER SIZE
005172  005737  001404           3$:      TST     MAXDL             ;VALUE IN 'MAXDL' ?
005176  001012                            BNE     4$                ;BR IF VALUE IS
005200  012737  013534  001404            MOV     #5980.,MAXDL      ;ASSUME FULL TRACK + 1 SEC MAXIMUM
005206  023737  001404  001622            CMP     MAXDL,BUFTBL+4    ;IS THAT TOO LARGE ?
005214  103403                            BLO     4$                ;BR IF NOT
005216  013737  001622  001404            MOV     BUFTBL+4,MAXDL    ;USE MAX AVAIL MEMORY AS MAX BUFFER SIZE
005224  013737  001622  054754   4$:      MOV     BUFTBL+4,PARLST+2 ;VALUE FOR THE PARAMETER TABLE

                                 ;SEE IF THE OPERATOR WANTS TO CHANGE ANY PARAMETERS

005232  005737  000042           LKPAR:   TST     @#42              ;'XXDP' CHAIN MODE OR 'ACT11' OPERATION ?
005236  001022                            BNE     SETVEC            ;BR IF YES
005240  104401  055050                    TYPE    ,ASKPAR           ;ASK FOR PARMETERS
005244  104411                            RDLIN                     ;READ THE ENTRY
```

```
005246  012605                          MOV     (SP)+,R5        ;ADDRESS OF ENTRY TO R5
005250  122715  000131                  CMPB    #'Y,(R5)        ;WAS ENTRY A 'Y' (YES)
005254  001013                          BNE     SETVEC          ;BR IF NOT 'Y'

005256  012703  054752          ENTPR:  MOV     #PARLST,R3      ;PARAMETER TABLE ADDRESS
005262  004737  026604                  JSR     PC,PARENT       ;GET THE PARAMETER ENTRY
005266  023727  001404  000004          CMP     MAXDL,#4        ;IS THE 'MAXDL' VALUE OK ?
005274  103003                          BHIS    SETVEC          ;BR IF IT IS
005276  012737  000004  001404          MOV     #4,MAXDL        ;SET 'MAXDL' TO THE MINIMUM VALUE

                                ;DISPLAY DRIVE STATUS AND SET UP THE OTHER SYSTEM DEVICES THAT
                                ;       THE PROGRAM WILL USE

005304  004737  022714          SETVEC: JSR     PC,CKCLK        ;START THE CLOCK
005310  004737  034520                  JSR     PC,RPINIT       ;INITIALIZE THE RP04/5/6 DRIVER
005314  012737  177777  034442          MOV     #-1,SAVEFG      ;SET THE SAVE REGISTERS FLAG
005322  062727  177777  000000          ADD     #-1,#0          ;CHECK FOR FIRST START
005330  103004                          BCC     11$             ;BR IF FIRST START
005332  032777  000004  173600          BIT     #SW02,@SWR      ;TYPEOUT THE DRIVE STATUS TABLE ?
005340  001105                          BNE     10$             ;BR IF NOT
005342  012737  000340  177776  11$:    MOV     #PR7,PS         ;SET PRIORITY TO 7
005350  005004                          CLR     R4              ;DRIVE TABLE POINTER
005352  104401  001165                  TYPE    ,$CRLF          ;CR-LF
005356  104401  053672                  TYPE    ,SYSTAT         ;TYPE STATUS HEADING
005362                          1$:
005362  010446                          MOV     R4,-(SP)        ;;SAVE R4 FOR TYPEOUT
                                                                ;;TYPE DRIVE NUMBER
005364  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
005366     002                          .BYTE   2               ;;TYPE 2 DIGIT(S)
005367     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
005370  104401  053361                  TYPE    ,LIN4SP         ;SPACES
005374  105764  034354                  TSTB    DRVSTA(R4)      ;CHECK DRIVE'S STATUS
005400  100425                          BMI     4$              ;BR IF UNSAFE
005402  001027                          BNE     5$              ;BR IF ONLINE
005404  105764  034364                  TSTB    DRVTYP(R4)      ;SEE IF OFFLINE OR NONEXISTENT
005410  001404                          BEQ     2$              ;BR IF NONEXISTENT
005412  100006                          BPL     3$              ;BR IF OFFLINE
005414  104401  053605                  TYPE    ,NOTRP          ;DRIVE NOT AN RP04/5/6
005420  000447                          BR      9$              ;CHECK NEXT DRIVE
005422  104401  053626          2$:     TYPE    ,NOTPRS         ;DRIVE NOT PRESENT
005426  000444                          BR      9$              ;CHECK NEXT DRIVE
005430  132764  000010  034364  3$:     BITB    #BIT03, DRVTYP(R4)      ;DRIVE PROGRAMMABLE?
005436  001403                          BEQ     12$             ;BRANCH IF NO
005440  104401  053441                  TYPE    ,NOUSE          ;PRINT MSG
005444  000410                          BR      6$              ;PRINT DRIVE TYPE
005446  104401  053514          12$:    TYPE    ,UNTOFF         ;DRIVE OFFLINE
005452  000405                          BR      6$              ;PRINT DRIVE TYPE
005454  104401  053662          4$:     TYPE    ,NOTSAF         ;DRIVE UNSAFE
005460  000402                          BR      6$              ;PRINT DRIVE TYPE
005462  104401  053525          5$:     TYPE    ,UNTON          ;DRIVE ONLINE
005466  104401  053363          6$:     TYPE    ,LINSP          ;SPACES
005472  012737  053711  005536          MOV     #RP04B,8$       ;ADDRESS OF RP04 MESSAGE
005500  132764  000001  034364          BITB    #BIT00,DRVTYP(R4)       ;RP04 ?
005506  001012                          BNE     7$              ;BR IF YES
005510  012737  053716  005536          MOV     #RP05,8$        ;ADDRESS OF RP05 MESSAGE
005516  132764  000002  034364          BITB    #BIT01,DRVTYP(R4)       ;RP05 ?
005524  001003                          BNE     7$              ;BR IF YES
```

```
005526  012737  053723  005536            MOV     #RP06,8$          ;ADDRESS OF RP06 MESSAGE
005534  104401                    7$:     TYPE                      ;TYPE THE DRIVE TYPE MESSAGE
005536  000000                    8$:     .WORD   0                 ;MESSAGE ADDRESS HERE
005540  104401  001165            9$:     TYPE    ,$CRLF            ;CR-LF
005544  005204                            INC     R4                ;INCREMENT DRIVE NUMBER/TABLE POINTER
005546  020427  000010                    CMP     R4,#8.            ;FINISHED ?
005552  001303                            BNE     1$                ;BR IF NOT
005554  104401  001165            10$:    TYPE    ,$CRLF            ;CR-LF
005560  005037  177776                    CLR     PS                ;SET PRIORITY BACK TO '0'
005564  000137  005570                    JMP     MONTR             ;CHECK FOR 'XXDP' OR 'ACT11' MONITOR

                                  ;SETUP IF 'XXDP' OR 'ACT11' OPERATION

005570  005737  000042            MONTR:  TST     42                ;'XXDP' CHAIN MODE OR 'ACT11' AUTO ACCEPT
005574  001402                            BEQ     1$                ;BR IF NEITHER
005576  004737  024576                    JSR     PC,ASGN2          ;ASSIGN DRIVES
005602  005227  177777            1$:     INC     #-1               ;FIRST START ?
005606  001011                            BNE     2$                ;BR IF NOT
005610  105737  000041                    TSTB    @#41              ;LOADED FROM PAPER TAPE ?
005614  001406                            BEQ     2$                ;BR IF YES
005616  023727  001256  100000            CMP     LSTAD,#100000     ;MORE THAN 16K ON THE SYSTEM ?
005624  103002                            BHIS    2$                ;BR IF YES
005626  104401  057105                    TYPE    ,NOLOAD           ;TELL THE OPERATOR THAT THE 'XXDP' LOADER
                                                                    ;WILL BE OVERWRITTEN
005632  004737  030400            2$:     JSR     PC,$TKINT         ;INITIALIZE THE KEYBOARD INTERRUPT HANDLER
005636  104401  054641                    TYPE    ,INTDON           ;TYPE 'INITIALIZE COMPLETE'
005642  000137  006024                    JMP     MAIN1             ;START THE PROGRAM

                                  ;'XXDP' OR 'ACT11' END OF TEST ROUTINE

005646  013700  000042            $GET42: MOV     42,R0             ;MONITOR ADDRESS
005652  001002                            BNE     1$                ;BR IF MONITOR
005654  000137  006024                    JMP     MAIN1             ;NONE, CONTINUE
005660  000005                    1$:     RESET                     ;CLEAR EVERYTHING

005662  004710                    $ENDAD: JSR     PC,(R0)           ;GO TO THE MONITOR
005664  000240                            NOP                       ;SAVE ROOM
005666  000240                            NOP                       ;FOR
005670  000240                            NOP                       ;ACT11
005672  000137  004166            $DOAGN: JMP     START             ;START AGAIN
```

```
                                .SBTTL  MAIN PROGRAM

 005676  012703  000010         MAIN:   MOV     #8.,R3                  ;DRIVE COUNTER
 005702  012705  001464                 MOV     #DUNIT,R5               ;ADDRESS OF 'DROP DRIVE' TABLE
 005706  005715                 1$:     TST     (R5)                    ;SEE IF ENTRY AT PRESENT POSITION
 005710  001011                         BNE     3$                      ;BR IF THERE IS ONE
 005712  062705  000002         2$:     ADD     #2,R5                   ;INCREMENT TO NEXT TABLE POSITION
 005716  005303                         DEC     R3                      ;DECREMENT DRIVE COUNTER
 005720  001372                         BNE     1$                      ;BR IF MORE TO CHECK
 005722  005737  001462                 TST     ASNLST                  ;ANY DRIVES ACTIVE ?
 005726  001036                         BNE     MAIN1                   ;BR IF YES
 005730  000137  005646                 JMP     $GET42                  ;CHECK FOR MONITOR RETURN
 005734  012701  001530         3$:     MOV     #AVAIL,R1               ;ADDRESS OF 'AVAILABLE DRIVES' TABLE
 005740  005711                 4$:     TST     (R1)                    ;SEE IF AT END OF TABLE
 005742  001405                         BEQ     5$                      ;BR IF AT END: GO CHECK 'WAIT' TABLE
 005744  021115                         CMP     (R1),(R5)               ;IS DRIVE IN 'AVAIL' THE ONE TO BE DROPPED
 005746  001414                         BEQ     7$                      ;BR IF YES
 005750  062701  000002                 ADD     #2,R1                   ;INCREMENT 'AVAIL' TABLE ADDRESS
 005754  000771                         BR      4$                      ;CONTINUE LOOKING
 005756  012701  001552         5$:     MOV     #WAIT,R1                ;MOVE THE ADDRESS OF THE BUFFER WAIT TABLE
 005762  005711                 6$:     TST     (R1)                    ;AT THE END OF THE 'WAIT' TABLE ?
 005764  001752                         BEQ     2$                      ;BR IF YES: SEE IF ANY MORE 'DROP' REQUESTS
 005766  021115                         CMP     (R1),(R5)               ;DRIVE IN THE 'WAIT' TABLE ?
 005770  001403                         BEQ     7$                      ;BR IF IT IS
 005772  062701  000002                 ADD     #2,R1                   ;INCREMENT 'WAIT' TABLE ADDRESS
 005776  000771                         BR      6$                      ;CONTINUE LOOK THROUGH THE 'WAIT' TABLE
 006000  011100                 7$:     MOV     (R1),R0                 ;PUT THE DRIVE'S BLOCK ADDRESS IN R0
 006002  104401  054171                 TYPE    ,DEASSG                 ;TYPE 'DRIVE DEASSIGNED'
 006006  004737  023200                 JSR     PC,TYPEST               ;TYPE THE DRIVE'S PERFORMANCE SUMMARY
 006012  005015                         CLR     (R5)                    ;CLEAR THE 'DROP DRIVE' TABLE ENTRY
 006014  005011                         CLR     (R1)                    ;REMOVE THE DRIVE FROM THE 'AVAIL' OR 'WAIT' TABLE
 006016  004737  020032                 JSR     PC,CMPRES               ;COMPRESS THE RESPECTIVE TABLE
 006022  000733                         BR      2$                      ;SEE IF ANY MORE DRIVES

                                ;LOOK FOR DRIVES TO BE ASSIGNED

 006024  012703  000010         MAIN1:  MOV     #8.,R3                  ;DRIVE COUNT
 006030  005002                         CLR     R2                      ;'AVAIL' INDEX
 006032  005004                         CLR     R4                      ;ASSIGN LIST INDEX
 006034  005005                         CLR     R5                      ;NEW DRIVE INDEX
 006036  005765  001506         1$:     TST     NEWUNT(R5)              ;NEW DRIVE IN THIS POSITION
 006042  001006                         BNE     3$                      ;BR IF THERE IS
 006044  062705  000002         2$:     ADD     #2,R5                   ;INCREMENT R5
 006050  005204                         INC     R4                      ;INCREMENT ASS'GN INDEX
 006052  005303                         DEC     R3                      ;DECREMENT DRIVE COUNT
 006054  001370                         BNE     1$                      ;BR IF MORE DRIVES
 006056  000432                         BR      MAIN2                   ;START OPERATIONS FOR THE AVAILABLE DRIVES
 006060  104401  053506         3$:     TYPE    ,UNTMSG                 ;'DRIVE'
 006064  010446                         MOV     R4,-(SP)                ;;SAVE R4 FOR TYPEOUT
                                                                        ;;TYPE DRIVE NUMBER
 006066  104403                         TYPOS                           ;;GO TYPE--OCTAL ASCII
 006070     002                         .BYTE   2                       ;;TYPE 2 DIGIT(S)
 006071     000                         .BYTE   0                       ;;SUPPRESS LEADING ZEROS
 006072  104401  054237                 TYPE    ,ASGND                  ;'ASSIGNED'
 006076  005762  001530         4$:     TST     AVAIL(R2)               ;AT END OF AVAILABLE TABLE
 006102  001403                         BEQ     5$                      ;BR IF YES
 006104  062702  000002                 ADD     #2,R2                   ;INCREMENT AVAILABLE TABLE INDEX
```

```
006110  000772                          BR      4$              ;CONTINUE LOOKING FOR END OF TABLE
006112  016562  001506  001530  5$:     MOV     NEWUNT(R5),AVAIL(R2)  ;MOVE ADDR OF DRIVE INTO AVAIL LST
006120  005065  001506                  CLR     NEWUNT(R5)      ;TAKE DRIVE OUT OF NEW DRIVE TABLE
006124  156437  034470  001462          BISB    ATABIT(R4),ASNLST ;SET DRIVE ASSIGNED INDICATOR
006132  016200  001530                  MOV     AVAIL(R2),R0    ;PUT STARTING ADDRESS OF BLOCK IN R0
006136  062702  000002                  ADD     #2,R2           ;INCREMENT AVAILABLE TABLE POINTER
006142  000740                          BR      2$              ;LOOK FOR MORE DRIVES

                                ;GET PARAMETERS, BUFFER SPACE, AND START ORDERS FOR DRIVES IN
                                ;       THE 'AVAILABLE' QUEUE

006144  005737  001552          MAIN2:  TST     WAIT            ;OUTSTANDING BUFFER REQUESTS
006150  001113                          BNE     MAIN3           ;BR IF THERE ARE
006152  005002                          CLR     R2              ;CLEAR DRIVE TABLE POINTER
006154  005762  001530          1$:     TST     AVAIL(R2)       ;ANY DRIVES WAITING FOR PARAMETERS
006160  001551                          BEQ     IDLE            ;BRANCH IF NONE
006162  016200  001530                  MOV     AVAIL(R2),R0    ;CONTROL BLOCK ADDR IN R0
006166  005760  000104                  TST     $NEXT(R0)       ;PARAMETERS BEEN SELECTED ?
006172  001021                          BNE     6$              ;BR IF THEY HAVE
006174  105760  000026                  TSTB    $PACK(R0)       ;'R' OR 'W' COMMAND FOR THE DRIVE ?
006200  001403                          BEQ     2$              ;BR IF NOT
006202  004737  020050                  JSR     PC,WRTPK        ;GET DATA PACK PARAMETERS
006206  000415                          BR      7$              ;GET THE BUFFER
006210  012701  001574          2$:     MOV     #PARQ,R1        ;ADDRESS OF THE PARAMETER QUEUE
006214  020011                  3$:     CMP     R0,(R1)         ;IS CURRENT DRIVE IN THE QUEUE ?
006216  001403                          BEQ     4$              ;BR IF IT IS
006220  005721                          TST     (R1)+           ;AT END OF THE QUEUE
006222  001403                          BEQ     5$              ;BR IF AT END
006224  000773                          BR      3$              ;CONTINUE LOOKING
006226  004737  020032          4$:     JSR     PC,CMPRES       ;COMPRESS THE TABLE
006232  004737  016750          5$:     JSR     PC,SELPAR       ;SELECT THE PARAMETERS
006236  004737  017624          6$:     JSR     PC,GETPAR       ;LOAD NEW PARAMETERS
006242  005046                  7$:     CLR     -(SP)           ;MAKE ROOM ON THE STACK FOR THE BUFFER ADDR
006244  004737  016174                  JSR     PC,GETBUF       ;GET BUFFER
006250  012660  000006                  MOV     (SP)+,$BUF(R0)  ;MOVE BUFFER ADDR TO DPB
006254  001424                          BEQ     8$              ;BR IF '0' ADDR (NO BUFFER)
006256  004737  016544                  JSR     PC,FILBUF       ;FILL THE BUFFER
006262  005060  000072                  CLR     $FAIR(R0)       ;CLEAR THE 'FAIRNESS' COUNT
006266  004737  016672                  JSR     PC,GODRIV       ;PUT CURRENT DPB IN DRIVER
006272  012705  001440                  MOV     #ORDERQ,R5      ;ADDRESS OF ORDER QUEUE IN R5
006276  005725                          TST     (R5)+           ;END OF QUEUE ?
006300  001376                          BNE     .-2             ;BR IF NOT
006302  010045                          MOV     R0,-(R5)        ;PUT BLOCK ADDRESS INTO QUEUE
006304  105760  000026                  TSTB    $PACK(R0)       ;'R' OR 'W' COMMAND FOR DRIVE ?
006310  001025                          BNE     10$             ;BR IF EITHER
006312  012705  001574                  MOV     #PARQ,R5        ;PUT BLOCK INTO THE PARAMETER QUEUE
006316  005725                          TST     (R5)+           ;FIND THE END OF THE QUEUE
006320  001376                          BNE     .-2             ;BR IF NOT AT END OF QUEUE
006322  010045                          MOV     R0,-(R5)        ;PUT BLOCK ADDRESS INTO THE QUEUE
006324  000417                          BR      10$             ;CONTINUE LOOKING
006326  026037  000072  001254  8$:     CMP     $FAIR(R0),FAIRNS  ;ENTRY BEEN IN THE QUEUE LONG ENOUGH ?
006334  001405                          BEQ     9$              ;BR IF YES
006336  005260  000072                  INC     $FAIR(R0)       ;INCREMENT THE ENTRY COUNT
006342  062702  000002                  ADD     #2,R2           ;INCREMENT THE POINTER
006346  000702                          BR      1$              ;LOOK FOR SOME MORE DRIVES
006350  012705  001552          9$:     MOV     #WAIT,R5        ;'WAIT' QUEUE ADDRESS
006354  005725                          TST     (R5)+           ;LOOK FOR AN OPENING
```

```
006356  001376                   BNE     .-2                ;BR IF NONE YET
006360  016245  001530           MOV     AVAIL(R2),-(R5)    ;MOVE DRIVE'S BLOCK ADDRESS TO QUEUE
006364  012701  001530   10$:    MOV     #AVAIL,R1          ;'AVAILABLE' TABLE ADDRESS
006370  060201                   ADD     R2,R1              ;FORM ADDRESS OF LAST ENTRY
006372  004737  020032           JSR     PC,CMPRES          ;COMPRESS THE TABLE
006376  000666                   BR      1$                 ;CONTINUE LOOKING

                         ;GET BUFFER ASSIGNMENTS FOR DRIVES IN THE 'BUFFER WAIT' QUEUE

006400  013700  001552   MAIN3:  MOV     WAIT,R0            ;MOVE THE 'WAIT' ENTRY TO R0
006404  005046                   CLR     -(SP)              ;MAKE ROOM ON THE STACK FOR THE BUFFER ADDR
006406  004737  016174           JSR     PC,GETBUF          ;TRY TO GET A BUFFER
006412  012660  000006           MOV     (SP)+,$BUF(R0)     ;MOVE THE BUFFER ADDR TO THE DPB
006416  001002                   BNE     1$                 ;BR IF A BUFFER WAS ASSIGNED
006420  000137  006504           JMP     IDLE               ;NO BUFFER AVAILABLE YET
006424  004737  016544   1$:     JSR     PC,FILBUF          ;FILL THE BUFFER
006430  004737  016672           JSR     PC,GODRIV          ;PUT THE ENTRY IN THE DRIVER
006434  005060  000072           CLR     $FAIR(R0)          ;CLEAR THE 'FAIRNESS' COUNT
006440  012705  001440           MOV     #ORDERQ,R5         ;ADDRESS OF ORDER QUEUE IN R5
006444  005725                   TST     (R5)+              ;AT END OF THE QUEUE
006446  001376                   BNE     .-2                ;BR IF NOT
006450  010045                   MOV     R0,-(R5)           ;PUT BLOCK ADDRESS IN QUEUE
006452  105760  000026           TSTB    $PACK(R0)          ;'R' OR 'W' COMMAND FOR DRIVE ?
006456  001005                   BNE     2$                 ;BR IF YES, DON'T PUT BLOCK INTO 'PARQ'
006460  012705  001574           MOV     #PARQ,R5           ;FIND THE END OF THE PARAMETER QUEUE
006464  005725                   TST     (R5)+              ;OPEN SLOT IN THE QUEUE ?
006466  001376                   BNE     .-2                ;BR IF NOT
006470  010045                   MOV     R0,-(R5)           ;PUT BLOCK ADDRESS INTO THE QUEUE
006472  012701  001552   2$:     MOV     #WAIT,R1           ;ADDRESS OF TABLE TO TO COMPRESS
006476  004737  020032           JSR     PC,CMPRES          ;COMPRESS THE WAIT TABLE
006502  000620                   BR      MAIN2              ;LOOK FOR MORE ENTRIES

                         ;WAIT FOR AN ORDER TO FINISH

006504  012701  001440   IDLE:   MOV     #ORDERQ,R1         ;ADDRESS OF THE ORDER QUEUE IN R1
006510  012100           1$:     MOV     (R1)+,R0           ;PUT BLOCK ADDRESS INTO R0
006512  001433                   BEQ     IDLE1              ;BR IF END OF QUEUE
006514  005760  000016           TST     $STATUS(R0)        ;SEE IF DRIVE FINISHED
006520  001773                   BEQ     1$                 ;BR IF DRIVE NOT FINISHED
006522  162701  000002           SUB     #2,R1              ;CORRECT THE QUEUE POINTER
006526  010146                   MOV     R1,-(SP)           ;SAVE THE QUEUE ADDRESS
006530  004737  016030           JSR     PC,STATIS          ;ACCUMULATE STATISTICS FOR DRIVE IN R0
006534  000240                   NOP                        ;DEBUGGING AID
006536  004737  007042           JSR     PC,PROCES          ;PROCESS END OF ORDER
006542  005037  001264           CLR     BADSEC             ;CLEAR THE BAD TRK/SEC ERROR INDICATOR
006546  004737  027036           JSR     PC,ABNRML          ;SEE IF ANY DRIVES HAVE TOO MANY ERRORS
006552  004737  027064           JSR     PC,EOP             ;SEE IF ANY DRIVE HAS XFERED 3X10^9 BITS
006556  012601                   MOV     (SP)+,R1           ;RESTORE THE ORDER TABLE INDEX
006560  012705  001530           MOV     #AVAIL,R5          ;FIND THE END OF THE 'AVAILABLE' TABLE
006564  005725           2$:     TST     (R5)+              ;END OF THE TABLE ?
006566  001376                   BNE     2$                 ;BR IF NOT AT END OF LIST
006570  011145                   MOV     (R1),-(R5)         ;MOVE THE BLOCK ADDRESS INTO THE TABLE
006572  004737  020032           JSR     PC,CMPRES          ;COMPRESS THE ORDER QUEUE
006576  004737  016330           JSR     PC,RELBUF          ;RESTORE BUFFER

006602  005737  001262   IDLE1:  TST     CFLAG              ;'CONTROL C' FLAG ENTERED ?
006606  001403                   BEQ     1$                 ;BR IF IT WAS
```

```
006610  004737  024212                   JSR   PC,KSR              ;SERVICE THE KEYBOARD
006614  000733                           BR    IDLE                ;SYSTEM WAS BUSY
006616  032777  000004  172314  1$:      BIT   #SW02,@SWR          ;TYPE PERFORMANCE SUMMARY
006624  001007                           BNE   2$                  ;BR IF NOT
006626  005737  001214                   TST   STATIN              ;TIME TO TYPE THE PERFORMANCE SUMMARY ?
006632  001404                           BEQ   2$                  ;BR IF NOT
006634  005037  001214                   CLR   STATIN              ;CLEAR THE INDICATOR
006640  004737  023116                   JSR   PC,STATPR           ;TYPE THE SUMMARY
006644  005737  001574          2$:      TST   PARQ                ;ENTRY IN THE PARAMETER QUEUE ?
006650  001410                           BEQ   3$                  ;BR IF NOT
006652  013700  001574                   MOV   PARQ,R0             ;PUT THE BLOCK ADDRESS INTO R0
006656  004737  016750                   JSR   PC,SELPAR           ;GET THE PARAMETERS FOR NEXT OPERATION
006662  012701  001574                   MOV   #PARQ,R1            ;SETUP TO COMPRESS THE TABLE
006666  004737  020032                   JSR   PC,CMPRES           ;COMPRESS THE PARAMETER QUEUE
006672  000137  005676          3$:      JMP   MAIN                ;CONTINUE THE LOOP


                                ;SETUP TO REFORMAT AN ERROR SECTOR

006676  032777  000001  172234  REFMT:   BIT   #SW0,@SWR           ;READ ONLY SWITCH SET ?
006704  001055                           BNE   REFMTX              ;BR IF IT IS
006706  032777  000200  172224           BIT   #SW7,@SWR           ;SWITCH 7 SET ?
006714  001051                           BNE   REFMTX              ;BR IF IT IS
006716  005737  001416                   TST   FORMAT              ;WRITE HEADER & DATA ORDERS ALLOWED ?
006722  001446                           BEQ   REFMTX              ;BR IF NOT
006724  016060  000272  000100           MOV   $RPCC(R0),$NCYL(R0)   ;USE PRESENT CYLINDER
006732  004737  022616                   JSR   PC,READDR           ;GET CORRECTED SECTOR-TRACK ADDRESSES
006736  112660  000077                   MOVB  (SP)+,$NTRK(R0)     ;TRACK ADDR TO DPB
006742  112660  000076                   MOVB  (SP)+,$NSEC(R0)     ;SECTOR ADDR TO DPB
006746  012760  000404  000102           MOV   #260.,$NWRDL(R0)    ;WORD COUNT FOR FORMAT
006754  023727  001404  000404           CMP   MAXDL,#260.         ;CAN A FULL SECTOR BE WRITTEN ?
006762  103003                           BHIS  1$                  ;BR IF IT CAN
006764  013760  001404  000102           MOV   MAXDL,$NWRDL(R0)    ;PUT TRANSFER SIZE INTO THE DPB
006772  112760  000003  000074  1$:      MOVB  #3,$NCODE(R0)       ;COMMAND CODE
007000  004737  017576                   JSR   PC,GETPAT           ;GET A PATTERN
007004  110560  000075                   MOVB  R5,$NPATC(R0)       ;PATTERN CODE TO CONTROL BLOCK
007010  012760  177777  000104           MOV   #-1,$NEXT(R0)       ;SET PARAMETERS SELECTED INDICATOR
007016  012701  001574                   MOV   #PARQ,R1            ;SET UP TO SEE IF BLOCK IN THE PARAMETER QUEUE
007022  005711                  2$:      TST   (R1)                ;SEE IF AT END OF TABLE
007024  001405                           BEQ   REFMTX              ;BR IF AT END
007026  020021                           CMP   R0,(R1)+            ;SEE IF BLOCK AT PRESENT POSITION
007030  001374                           BNE   2$                  ;BR IF NOT
007032  005041                           CLR   -(R1)               ;CLEAR THE ENTRY
007034  004737  020032                   JSR   PC,CMPRES           ;COMPRESS THE TABLE
007040  000207                  REFMTX:  RTS   PC                  ;RETURN
```

```
                                        ;PROCESS THE ORDER TERMINATION

007042  111037  001246                  PROCES: MOVB    (R0),UNIT               ;DRIVE NUMBER FOR ANY ERROR MESSAGES
007046  005760  000016                          TST     $STATUS(R0)             ;SEE IF DRIVER SIGNALED AN ERROR
007052  100427                                  BMI     ERPROC                  ;BR IF ERROR
007054  032760  100000  000234                  BIT     #BIT15,$RPCS1(R0)       ;SEE IF 'SC' SET
007062  001410                                  BEQ     1$                      ;BR IF NOT SET
007064  032760  040000  000234                  BIT     #BIT14,$RPCS1(R0)       ;SEE IF 'TRE' SET
007072  001017                                  BNE     ERPROC                  ;BR IF SET
007074  032760  040000  000246                  BIT     #BIT14,$RPDS1(R0)       ;SEE IF 'ERR' SET
007102  001013                                  BNE     ERPROC                  ;BR IF SET
007104  004737  013272                  1$:     JSR     PC,CKERR                ;NO ERROR, CHECK ERROR BITS ANYWAY
007110  004737  013372                          JSR     PC,CKBUS                ;NO ERROR, CHECK BUS ADDR & WC
007114  032777  000002  172016                  BIT     #SW01,@SWR              ;DATA COMPARE ?
007122  001002                                  BNE     2$                      ;BR IF NOT
007124  004737  013456                          JSR     PC,CMPAR                ;NO ERROR, COMPARE DATA
007130  000207                          2$:     RTS     PC                      ;RETURN

                                        ;ORDER TERMINATED WITH AN ERROR - PROCESS THE ERROR

007132  032760  000200  000016          ERPROC: BIT     #BIT07,$STATUS(R0)      ;DONE BIT SET ?
007140  001402                                  BEQ     ERPRC1                  ;BR IF ORDER DIDN'T COMPLETE NORMALLY
007142  000137  007566                          JMP     DONE                    ;PROCESS ERROR WITH 'DONE' BIT SET

                                        ;PROCESS ORDER COMPLETION WITH 'ERROR' & 'DONE NOT' BITS

007146  032760  010000  000016          ERPRC1: BIT     #BIT12,$STATUS(R0)      ;SEE IF DRIVE WAS UNSAFE
007154  001025                                  BNE     PUNSAF                  ;BR IF YES
007156  032760  004000  000016                  BIT     #BIT11,$STATUS(R0)      ;PARITY ERROR OCCURRED
007164  001055                                  BNE     UCPAR                   ;BR IF IT DID
007166  032760  002000  000016                  BIT     #BIT10,$STATUS(R0)      ;FATAL PARITY ERROR?
007174  001056                                  BNE     FALPAR                  ;BR IF THERE IS ONE
007176  032760  001000  000016                  BIT     #BIT09,$STATUS(R0)      ;TIMEOUT?
007204  001076                                  BNE     SWTIM                   ;BR IF YES
007206  032760  040002  000016                  BIT     #BIT14!BIT01,$STATUS(R0)        ;DRIVE WENT OFFLINE ?
007214  001111                                  BNE     OFLIN                   ;BR IF IT DID
007216  032760  000004  000016                  BIT     #BIT2,$STATUS(R0)       ;PORT REQUEST TIME OUT ?
007224  001141                                  BNE     PRTIM                   ;BR IF IT DID
007226  000207                                  RTS     PC                      ;ERROR. RETURN

                                        ;DRIVE IS PERSISTENTLY UNSAFE

007230  104401  001165                  PUNSAF: TYPE    ,$CRLF                  ;CR-LF
007234  104401  046630                          TYPE    ,EM12                   ;'DRIVE UNSAFE' MESSAGE
007240  104401  054213                          TYPE    ,DRNUM                  ;DRIVE NUMBER
007244  013746  001246                          MOV     UNIT,-(SP)              ;;SAVE UNIT FOR TYPEOUT
                                                                                ;;TYPE DRIVE NUMBER
007250  104403                                  TYPOS                           ;;GO TYPE--OCTAL ASCII
007252     002                                  .BYTE   2                       ;;TYPE 2 DIGIT(S)
007253     000                                  .BYTE   0                       ;;SUPPRESS LEADING ZEROS
007254  104401  001165                          TYPE    ,$CRLF                  ;CR-LF
007260  004737  020502                          JSR     PC,LINE1                ;PRINT LINE 1 OF ERROR MESSAGE
007264  104414  046630                          DISPLY  ,EM12                   ;PERSISTENT DEVICE UNSAFE MESSAGE
007270  004737  020546                          JSR     PC,LINE2                ;PRINT LINE 2 OF ERROR MESSAGE
007274  004737  021154                          JSR     PC,LINE3                ;PRINT LINE 3 OF ERROR MESSAGE
007300  004737  021630                          JSR     PC,LINE4                ;PRINT LINE 4 OF THE ERROR MESSAGE
007304  004737  023726                          JSR     PC,INCTOT               ;INCREMENT TOTAL ERROR COUNT
```

```
007310  004737  022264                  JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
007314  000137  026760                  JMP     DROP            ;DROP THE DRIVE

                                ;UNCORRECTABLE MASSBUS PARITY ERROR OCCURRED

007320  104401  001165          UCPAR:  TYPE    ,$CRLF          ;CR-LF
007324  104401  046532                  TYPE    ,EM10           ;'UNCORRECTABLE PARITY ERROR' MESSAGE
007330  000404                          BR      FALPR1          ;FINISH PROCESSING THE ERROR

                                ;'FATAL' MASSBUS PARITY ERROR OCCURRED

007332  104401  001165          FALPAR: TYPE    ,$CRLF          ;CR-LF
007336  104401  046575                  TYPE    ,EM11           ;'FATAL PARITY ERROR' MESSAGE
007342  104401  054213          FALPR1: TYPE    ,DRNUM          ;DRIVE NUMBER
007346  013746  001246                  MOV     UNIT,-(SP)      ;;SAVE UNIT FOR TYPEOUT
                                                                ;;TYPE DRIVE NUMBER
007352  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
007354     002                          .BYTE   2               ;;TYPE 2 DIGIT(S)
007355     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
007356  104401  001165                  TYPE    ,$CRLF          ;CR-LF
007362  004737  023726                  JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
007366  032777  100000  171544          BIT     #SW15,@SWR      ;HALT ON ERROR ?
007374  001401                          BEQ     1$              ;BR IF NOT
007376  000000                          HALT                    ;ERROR HALT
007400  000207                  1$:     RTS     PC

                                ;SOFTWARE TIMEOUT OCCURRED

007402  004737  020502          SWTIM:  JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
007406  104414  046661                  DISPLY  ,EM13           ;PRINT THE TIME OUT MESSAGE
007412  004737  020546                  JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
007416  004737  021154                  JSR     PC,LINE3        ;PRINT LINE 3 OF ERROR MESSAGE
007422  004737  021630                  JSR     PC,LINE4        ;PRINT LINE 4 OF ERROR MESSAGE
007426  004737  023726                  JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
007432  004737  022264                  JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
007436  000207                          RTS     PC              ;RETURN

                                ;DRIVE WENT OFFLINE

007440  104401  001165          OFLIN:  TYPE    ,$CRLF          ;CR-LF
007444  104401  046733                  TYPE    ,EM14           ;'DRIVE WENT OFFLINE' MESSAGE
007450  104401  054213                  TYPE    ,DRNUM          ;DRIVE NUMBER
007454  013746  001246                  MOV     UNIT,-(SP)      ;;SAVE UNIT FOR TYPEOUT
                                                                ;;TYPE DRIVE NUMBER
007460  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
007462     002                          .BYTE   2               ;;TYPE 2 DIGIT(S)
007463     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
007464  104401  001165                  TYPE    ,$CRLF          ;CR-LF
007470  004737  020502                  JSR     PC,LINE1        ;PRINT LINE 1 OF THE ERROR MESSAGE
007474  104414  046733                  DISPLY  ,EM14           ;PRINT OFFLINE MESSAGE
007500  004737  020546                  JSR     PC,LINE2        ;PRINT LINE 2 OF THE ERROR MESSAGE
007504  004737  021154                  JSR     PC,LINE3        ;PRINT LINE 3 OF THE ERROR MESSAGE
007510  004737  021630                  JSR     PC,LINE4        ;PRINT LINE 4 OF THE ERROR MESSAGE
007514  004737  023726                  JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
007520  004737  022264                  JSR     PC,LINE7        ;PRINT LINE 7 OF THE ERROR MESSAGE
007524  000137  026760                  JMP     DROP            ;DROP THE DRIVE
```

```
                                    ;PORT REQUEST TIMEOUT ERROR

    007530  004737  020502          PRTIM:  JSR     PC,LINE1          ;TYPE LINE 1 OF THE ERROR MESSAGE
    007534  104414  046756                  DISPLY  ,EM15             ;PRINT PORT TIME OUT MESSAGE
    007540  004737  020546                  JSR     PC,LINE2          ;TYPE LINE 2 OF THE ERROR MESSAGE
    007544  004737  021154                  JSR     PC,LINE3          ;TYPE LINE 3 OF THE ERROR MESSAGE
    007550  004737  021630                  JSR     PC,LINE4          ;TYPE LINE 4 OF THE ERROR MESSAGE
    007554  004737  023726                  JSR     PC,INCTOT         ;INCREMENT TOTAL ERROR COUNT
    007560  004737  022264                  JSR     PC,LINE7          ;TYPE LINE 7 OF THE ERROR MESSAGE
    007564  000207                          RTS     PC                ;RETURN
```

```
                                          ;PROCESS ORDER COMPLETION WITH 'ERROR' & 'DONE' BITS SET

 007566  032760  000030  000016  DONE:    BIT     #BIT04!BIT03,$STATUS(R0) ;UNSAFE OCCURRED
 007574  001402                           BEQ     .+6                  ;BR IF NOT
 007576  000137  012770                   JMP     UNSAF                ;REPORT UNSAFE
 007602  032760  040000  000244           BIT     #BIT14,$RPCS2(R0)    ;IS 'WCE' SET ?
 007610  001006                           BNE     1$                   ;BR IF SET
 007612  032760  040000  000246           BIT     #BIT14,$RPDS1(R0)    ;CHECK 'ERR'
 007620  001002                           BNE     1$                   ;BR IF SET
 007622  000137  012534                   JMP     TRFER                ;PROCESS 'TRE'
 007626  032760  000400  000250  1$:      BIT     #BIT08,$RPER1(R0)    ;'HCRC' SET?
 007634  001402                           BEQ     .+6                  ;BR IF NOT
 007636  000137  011212                   JMP     HCRCER               ;PROCESS 'HCRC'
 007642  032760  000020  000250           BIT     #BIT04,$RPER1(R0)    ;'FMT' SET?
 007650  001402                           BEQ     .+6                  ;BR IF NOT SET
 007652  000137  011374                   JMP     CKFMT                ;CHECK FORMAT ERROR
 007656  032760  000200  000250           BIT     #BIT07,$RPER1(R0)    ;'HCE' SET?
 007664  001402                           BEQ     .+6                  ;BR IF NOT SET
 007666  000137  011570                   JMP     CKHCE                ;CHECK 'HCE' ERROR
 007672  032760  020000  000250           BIT     #BIT13,$RPER1(R0)    ;'OPI' SET?
 007700  001402                           BEQ     .+6                  ;BR IF NOT SET
 007702  000137  012070                   JMP     OPIER                ;REPORT 'OPI'
 007706  032760  000010  000250           BIT     #BIT3,$RPER1(R0)     ;'PAR' SET?
 007714  001402                           BEQ     .+6                  ;BR IF NOT SET
 007716  000137  012222                   JMP     PARER                ;REPORT 'PAR'
 007722  032760  000040  000250           BIT     #BIT5,$RPER1(R0)     ;'WCF' SET?
 007730  001402                           BEQ     .+6                  ;BR IF NOT SET
 007732  000137  012672                   JMP     WCFER                ;REPORT 'WCF'
 007736  032760  002000  000250           BIT     #BIT10,$RPER1(R0)    ;'IAE' SET?
 007744  001402                           BEQ     .+6                  ;BR IF NOT SET
 007746  000137  012314                   JMP     IAEER                ;REPORT 'IAE'
 007752  032760  004000  000250           BIT     #BIT11,$RPER1(R0)    ;'WLE' SET?
 007760  001402                           BEQ     .+6                  ;BR IF NOT SET
 007762  000137  012346                   JMP     WLEER                ;REPORT 'WLE'
 007766  032760  001000  000250           BIT     #BIT9,$RPER1(R0)     ;'AOE' SET?
 007774  001405                           BEQ     2$                   ;BR IF NOT SET
 007776  032760  002000  000246           BIT     #BIT10,$RPDS1(R0)    ;'LST' SET?
 010004  001401                           BEQ     2$                   ;BR IF NOT SET
 010006  000207                           RTS     PC                   ;'AOE' & 'LST' SET, EXIT
 010010  032760  010000  000250  2$:      BIT     #BIT12,$RPER1(R0)    ;SEE IF 'DTE' SET
 010016  001402                           BEQ     .+6                  ;BR IF NOT
 010020  000137  012200                   JMP     DTEER                ;REPORT 'DTE' ERROR
 010024  032760  040000  000244           BIT     #BIT14,$RPCS2(R0)    ;SEE IF 'WCK' SET
 010032  001402                           BEQ     .+6                  ;BR IF NOT SET
 010034  000137  010664                   JMP     WCKER                ;REPORT 'WCK'
 010040  005760  000250                   TST     $RPER1(R0)           ;SEE IF 'DCK' SET
 010044  100002                           BPL     .+6                  ;BR IF NOT
 010046  000137  010072                   JMP     DCKER                ;PROCESS 'DCK'
 010052  032760  140000  000276           BIT     #BIT15!BIT14,$RPER3(R0) ;'SKI' OR 'OCYL' SET
 010060  001402                           BEQ     .+6                  ;BR IF NOT SET
 010062  000137  012634                   JMP     SKIER                ;REPORT ERROR
 010066  000137  011342                   JMP     DRVER                ;REPORT DRIVE ERROR

                                          ;PROCESS DATA ('DCK') CHECK ERROR

 010072  022760  010042  000300  DCKER:   CMP     #10042,$RPEC1(R0)    ;VALID POSITION COUNT ?
 010100  101406                           BLOS    1$                   ;BR IF NOT VALID
```

```
010102  005760  000300                  TST     $RPEC1(R0)              ;POSITION COUNT 0 ?
010106  001403                          BEQ     1$                      ;BR IF 0'S
010110  005760  000302                  TST     $RPEC2(R0)              ;VALUE IN PATTERN REGISTER ?
010114  001026                          BNE     4$                      ;BR IF YES
010116  004737  020502          1$:     JSR     PC,LINE1                ;TYPE FIRST LINE OF ERROR MESSAGE
010122  104414  050363                  DISPLY  ,EM45                   ;TYPE 'ECC LOGIC ERROR'
010126  004737  020546                  JSR     PC,LINE2                ;TYPE LINE 2 OF ERROR MESSAGE
010132  004737  023726                  JSR     PC,INCTOT               ;INCREMENT TOTAL ERROR COUNT
010136  012737  000003  001252          MOV     #3,RETRY                ;RETRY COUNT
010144  004737  015702                  JSR     PC,$RETRY               ;RETRY THE ORDER
010150  000403                          BR      2$                      ;RETRY WAS NOT SUCCESSFUL
010152  004737  022202                  JSR     PC,LINE6C               ;TYPE LINE 6C OF ERROR MESSAGE
010156  000402                          BR      3$                      ;FINISH THE ERROR REPORT
010160  004737  022210          2$:     JSR     PC,LINE6D               ;TYPE LINE 6D OF ERROR MESSAGE
010164  004737  022264          3$:     JSR     PC,LINE7                ;TYPE LINE 7 OF ERROR MESSAGE
010170  000402                          BR      5$                      ;EXIT
010172  004737  020344          4$:     JSR     PC,SPOTCK               ;SEE IF ERROR AT A BAD SPOT ON THE PACK
010176  000207                  5$:     RTS     PC                      ;IT IS, DON'T REPORT IT
010200  126027  000024  000001          CMPB    $CODE(R0),#1            ;IS ORDER A WRITE CHECK ?
010206  101002                          BHI     6$                      ;BR IF NOT
010210  000137  010664                  JMP     WCKER                   ;REPORT ERROR UNDER WRITE CHECK PROCESSING
010214  004737  020502          6$:     JSR     PC,LINE1                ;PRINT LINE 1 OF ERROR MESSAGE
010220  104414  047033                  DISPLY  ,EM21                   ;DATA CHECK ERROR
010224  004737  020546          DCKER1: JSR     PC,LINE2                ;PRINT LINE 2 OF ERROR MESSAGE
010230  004737  021154                  JSR     PC,LINE3                ;PRINT LINE 3 OF ERROR MESSAGE
010234  004737  021630                  JSR     PC,LINE4                ;PRINT LINE 4 OF ERROR MESSAGE
010240  004737  015344                  JSR     PC,PRTBAD               ;SEE IF BAD SECTOR TO BE PRINTED
010244  012737  110100  001250          MOV     #BIT15!BIT12!BIT06,MASK   ;LOAD ERROR MASK
010252  032760  010100  000250          BIT     #BIT12!BIT06,$RPER1(R0)   ;CHECK 'DTE' & 'ECH'
010260  001003                          BNE     1$                      ;BR IF SET
010262  004737  022154                  JSR     PC,LINE6                ;PRINT LINE 6 OF ERROR MESSAGE
010266  000541                          BR      8$                      ;FINISH THE ERROR REPORT
010270  012737  000020  001252  1$:     MOV     #16.,RETRY              ;RETRY COUNT
010276  005001                          CLR     R1                      ;R1 IS OFFSET CODE POINTER
010300  032760  000002  000262          BIT     #BIT01,$RPDT(R0)   ;IS DRIVE AN RP06 ?
010306  001002                          BNE     16$                     ;BR IF IT IS
010310  012701  000007                  MOV     #7,R1                   ;INCREMENT PAST RP06 OFFSET CODES
010314  004737  016330          16$:    JSR     PC,RELBUF               ;RELEASE THE BUFFER
010320  004737  022616                  JSR     PC,READDR               ;GET THE ADDRESS OF THE ERROR SECTOR
010324  112660  000011                  MOVB    (SP)+,$TRK(R0)          ;TRACK ADDRESS OF ERROR SECTOR
010330  112660  000010                  MOVB    (SP)+,$SEC(R0)          ;SECTOR ADDRESS OF ERROR SECTOR
010334  016060  000272  000012          MOV     $RPCC(R0),$CYL(R0)   ;PRESENT CYLINDER
010342  026060  000022  000020          CMP     $SSEC(R0),$WRDL(R0)   ;SEE IF TRANSFER LENGTH LESS THAN 1 SECTOR
010350  103010                          BHIS    15$                     ;BR IF IT IS; USE PRESENT TRANSFER LENGTH
010352  016060  000022  000020          MOV     $SSEC(R0),$WRDL(R0)   ;CHANGE TRANSFER SIZE TO 1 SECTOR
010360  016060  000020  000004          MOV     $WRDL(R0),$WRDM(R0)   ;SETUP WORD COUNT FOR OPERATION
010366  005460  000004                  NEG     $WRDM(R0)               ;CHANGE COUNT TO 2'S COMP
010372  005046                  15$:    CLR     -(SP)                   ;SPACE FOR NEW BUFFER ADDRESS
010374  004737  016174                  JSR     PC,GETBUF               ;GET A BUFFER
010400  012660  000006                  MOV     (SP)+,$BUF(R0)          ;NEW BUFFER ADDRESS TO DPB
010404  004737  016672          2$:     JSR     PC,GODRIV               ;RETRY
010410  005760  000016          3$:     TST     $TATUS(R0)              ;TEST FOR DONE
010414  001775                          BEQ     3$                      ;BR IF NOT DONE
010416  100075                          BPL     10$                     ;BR IF NOT ERROR
010420  032760  000200  000016          BIT     #BIT7,$TATUS(R0)   ;SEE IF ORDER TERMINIATED NORMALLY
010426  001006                          BNE     14$                     ;BR IF NOT
010430  004737  023726                  JSR     PC,INCTOT               ;INCREMENT TOTAL ERROR COUNT
```

```
010434  104414  052550                    DISPLY  ,LIN8M              ;'DIFFERENT ERROR DURING RETRY'
010440  000137  007146                    JMP     ERPRC1              ;SEE WHICH ERROR
010444  033760  001250  000250  14$:      BIT     MASK,$RPER1(R0)     ;LOOK AT CURRENT ERROR
010452  001430                            BEQ     5$                  ;BR IF DIFFERENT ERROR
010454  032760  010100  000250            BIT     #BIT12!BIT6,$RPER1(R0)    ;'ECH' OR 'DTE' STILL SET ?
010462  001437                            BEQ     7$                  ;BR IF NEITHER SET
010464  105237  001253                    INCB    RETRY+1             ;INCREMENT RETRY COUNT
010470  123737  001252  001253            CMPB    RETRY,RETRY+1       ;DONE ?
010476  001342                            BNE     2$                  ;BR IF NOT
010500  005201                            INC     R1                  ;INCREMENT TABLE INDEX
010502  116137  002220  046151            MOVB    OFFCOD(R1),GENDPB+$FMT   ;OFFSET CODE
010510  001435                            BEQ     9$                  ;BR IF END OF OFFSET TABLE
010512  062737  000002  001252            ADD     #2,RETRY            ;NEW RETRY LIMIT
010520  004737  015574                    JSR     PC,OFFST            ;OFFSET
010524  005737  046166          4$:       TST     GENDPB+$STATUS      ;SEE IF FINISHED WITH OFFSET
010530  001775                            BEQ     4$                  ;BR IF NOT
010532  100324                            BPL     2$                  ;BR IF NO ERROR PERFORMING OFFSET
010534  004737  022532          5$:       JSR     PC,LINE8            ;PRINT LINE 8 OF ERROR MESSAGE
010540  004737  023632          6$:       JSR     PC,INCHRD           ;INCREMENT 'HARD' ERROR COUNT
010544  004737  023726                    JSR     PC,INCTOT           ;INCREMENT TOTAL ERROR COUNT
010550  004737  022264                    JSR     PC,LINE7            ;PRINT LINE 7 OF ERROR MESSAGE
010554  004737  015344                    JSR     PC,PRTBAD           ;PRINT THE BAD SECTOR
010560  000436                            BR      13$                 ;CLEAN UP AND RETURN
010562  004737  022174          7$:       JSR     PC,LINE6B           ;PRINT LINE 6B OF ERROR MESSAGE
010566  004737  022112                    JSR     PC,LINE5B           ;PRINT LINE 5B OF THE ERROR MESSAGE
010572  004737  023606          8$:       JSR     PC,INCSOF           ;INCREMENT 'SOFT' ERROR COUNT
010576  004737  014604                    JSR     PC,ECC              ;CORRECT THE ERROR USING ECC AND CHECK IT
010602  000407                            BR      11$                 ;COMPARE THE BUFFER
010604  004737  022210          9$:       JSR     PC,LINE6D           ;PRINT LINE 6D OF ERROR MESSAGE
010610  000753                            BR      6$                  ;INCREMENT ERROR COUNT
010612  004737  022166          10$:      JSR     PC,LINE6A           ;PRINT LINE 6A OF ERROR MESSAGE
010616  004737  023606                    JSR     PC,INCSOF           ;INCREMENT 'SOFT' ERROR COUNT
010622  012737  000001  001300  11$:      MOV     #1,FRSTER           ;SET PROCESSING 'DCKER' INDICATOR
010630  004737  013474                    JSR     PC,CMPARD           ;COMPARE THE BUFFER
010634  105737  001301                    TSTB    FRSTER+1            ;ERROR IN COMPARE ?
010640  100406                            BMI     13$                 ;BRANCH IF ERROR
010642  004737  023726                    JSR     PC,INCTOT           ;INCREMENT TOTAL ERROR COUNT
010646  104414  052757                    DISPLY  ,LIN9G              ;'DATA COMPARE OK' MESSAGE
010652  004737  022264          12$:      JSR     PC,LINE7            ;PRINT LINE 7 OF ERROR MESSAGE
010656  004737  006676          13$:      JSR     PC,REFMT            ;REFORMAT THE ERROR SECTOR
010662  000207                            RTS     PC                  ;RETURN

                                ;WRITE CHECK ERROR PROCESSING

010664  032760  100000  000250  WCKER:    BIT     #BIT15,$RPER1(R0)   ;SEE IF 'DCK' SET ALSO
010672  001034                            BNE     2$                  ;BR IF IT IS
010674  004737  020502                    JSR     PC,LINE1            ;PRINT LINE 1 OF ERROR MESSAGE
010700  104414  047137                    DISPLY  ,EM23               ;PRINT WCE & DCK NOT
010704  005037  001250                    CLR     MASK                ;CLEAR ERROR MASK
010710  004737  020546                    JSR     PC,LINE2            ;PRINT LINE 2 OF ERROR MESSAGE
010714  004737  021154                    JSR     PC,LINE3            ;PRINT LINE 3 OF ERROR MESSAGE
010720  004737  021630                    JSR     PC,LINE4            ;PRINT LINE 4 OF ERROR MESSAGE
010724  004737  021720                    JSR     PC,LINE5            ;PRINT LINE 5 OF ERROR MESSAGE
010730  004737  023726                    JSR     PC,INCTOT           ;INCREMENT TOTAL ERROR COUNT
010734  012737  000003  001252            MOV     #3,RETRY            ;RETRY LIMIT
010742  004737  015702                    JSR     PC,$RETRY           ;RETRY THE OPERATION
010746  000403                            BR      1$                  ;RETRY UNSUCESSFUL
```

```
010750  004737  022202                  JSR     PC,LINE6C       ;PRINT LINE 6C OF ERROR MESSAGE
010754  000502                          BR      8$              ;FINISH PROCESSING THE ERROR
010756  004737  022210          1$:     JSR     PC,LINE6D       ;PRINT LINE 6D OF ERROR MESSAGE
010762  000506                          BR      10$             ;FINISH PROCESSING THE ERROR
010764  004737  020344          2$:     JSR     PC,SPOTCK       ;SEE IF ERROR AT BAD SPOT ON THE PACK
010770  000507                          BR      11$             ;EXIT IF AT BAD SPOT ON PACK
010772  004737  020502                  JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
010776  012737  047064  011024          MOV     #EM22,13$       ;ASSUME THAT EM22 WILL BE PRINTED
011004  032760  040000  000244          BIT     #BIT14,$RPCS2(R0)   ;DID 'WCK' ALSO SET ?
011012  001003                          BNE     12$             ;BR IF IT DID
011014  012737  047765  011024          MOV     #EM37,13$       ;MESSAGE FOR 'DCK' AND 'WCK' NOT DURING
                                                                ;WRITE CHECK
011022  104414                  12$:    DISPLY                  ;TYPE THE ERROR MESSAGE
011024  000000                  13$:    .WORD   0               ;MESSAGE ADDRESS GOES HERE
011026  004737  020546                  JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
011032  004737  021154                  JSR     PC,LINE3        ;PRINT LINE 3 OF ERROR MESSAGE
011036  004737  021630                  JSR     PC,LINE4        ;PRINT LINE 4 OF ERROR MESSAGE
011042  004737  021720                  JSR     PC,LINE5        ;PRINT LINE 5 OF ERROR MESSAGE
011046  032760  000100  000250          BIT     #BIT06,$RPER1(R0)   ;ECH SET ALSO ?
011054  001442                          BEQ     8$              ;FINISH PROCESSING THE ERROR
011056  012737  000020  001252  3$:     MOV     #16.,RETRY      ;RETRY LIMIT - 16 (10)
011064  004737  016672          4$:     JSR     PC,GODRIV       ;RETRY THE ORDER
011070  005760  000016          5$:     TST     $STATUS(R0)     ;ORDER FINISHED ?
011074  001775                          BEQ     5$              ;BR IF NOT
011076  100405                          BMI     6$              ;BR IF ERROR ON ORDER
011100  105237  001253                  INCB    RETRY+1         ;INCREMENT RETRY COUNT
011104  004737  022202                  JSR     PC,LINE6C       ;PRINT LINE 6C OF ERROR MESSAGE
011110  000431                          BR      9$              ;FINISH ERROR PROCESSING
011112  105237  001253          6$:     INCB    RETRY+1         ;INCREMENT RETRY COUNT
011116  123737  001252  001253          CMPB    RETRY,RETRY+1   ;DONE ?
011124  001714                          BEQ     1$              ;BR IF AT RETRY LIMIT
011126  032760  100000  000250          BIT     #BIT15,$RPER1(R0)   ;'DCK' SET
011134  001407                          BEQ     7$              ;BR IF NOT - DIFFERENT ERROR
011136  032760  000100  000250          BIT     #BIT06,$RPER1(R0)   ;'ECH' ALSO SET ?
011144  001347                          BNE     4$              ;BR IF IT IS, RETRY ORDER
011146  004737  022202                  JSR     PC,LINE6C       ;PRINT LINE 6C OF ERROR MESSAGE
011152  000403                          BR      8$              ;FINISH PROCESSING ERROR
011154  004737  022532          7$:     JSR     PC,LINE8        ;PRINT LINE 8 - 'DIFFERENT ERROR '
011160  000405                          BR      9$              ;FINISH PROCESSING ERROR
011162  004737  023726          8$:     JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
011166  004737  022264                  JSR     PC,LINE7        ;FINISH THE ERROR MESSAGE
011172  000406                          BR      11$             ;EXIT
011174  004737  023726          9$:     JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
011200  004737  022264          10$:    JSR     PC,LINE7        ;FINISH THE ERROR MESSAGE
011204  004737  006676                  JSR     PC,REFMT        ;REFORMAT THE SECTOR IN ERROR
011210  000207                  11$:    RTS     PC              ;RETURN

                                ;REPORT 'HCRC' ERROR

011212  004737  020344          HCRCER: JSR     PC,SPOTCK       ;SEE IF ERROR AT PACK BAD SPOT
011216  000450                          BR      3$              ;EXIT IF IT IS
011220  004737  020502                  JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
011224  104414  047012                  DISPLY  ,EM20           ;REPORT 'HCRC'
011230  004737  020546                  JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
011234  004737  021154                  JSR     PC,LINE3        ;PRINT LINE 3 OF ERROR MESSAGE
011240  004737  021630                  JSR     PC,LINE4        ;PRINT LINE 4 OF ERROR MESSAGE
011244  032760  040000  000244          BIT     #BIT14,$RPCS2(R0)   ;'WCE' ERROR ALSO ?
```

```
011252  001402                           BEQ     1$                      ;BR IF NOT
011254  004737  021720                   JSR     PC,LINE5                ;DISPLAY WORDS WHICH CAUSED 'WCE'
011260  004737  023606           1$:     JSR     PC,INCSOF               ;INCREMENT 'SOFT' ERROR COUNT
011264  004737  023726                   JSR     PC,INCTOT               ;INCREMENT TOTAL ERROR COUNT
011270  012737  000400  001250           MOV     #BIT8,MASK              ;SET ERROR MASK
011276  012737  000003  001252           MOV     #3,RETRY                ;RETRY LIMIT
011304  004737  015702                   JSR     PC,$RETRY               ;RETRY ORDER
011310  000405                           BR      2$                      ;RETRY NOT SUCESSFUL
011312  004737  022202                   JSR     PC,LINE6C               ;PRINT LINE 6C OF ERROR MESSAGE
011316  004737  022264                   JSR     PC,LINE7                ;PRINT LINE 7 OF ERROR MESSAGE
011322  000406                           BR      3$                      ;EXIT
011324  004737  022210           2$:     JSR     PC,LINE6D               ;PRINT LINE 6D OF ERROR MESSAGE
011330  004737  022264                   JSR     PC,LINE7                ;PRINT LINE 7 OF ERROR MESSAGE
011334  004737  006676                   JSR     PC,REFMT                ;REFORMAT THE ERROR SECTOR
011340  000207                   3$:     RTS     PC                      ;RETURN

                                 ;REPORT DRIVE ERROR

011342  004737  020502           DRVER:  JSR     PC,LINE1                ;PRINT LINE 1 OF ERROR MESSAGE
011346  104414  047427                   DISPLY  ,EM30                   ;REPORT DRIVE ERROR
011352  004737  020546                   JSR     PC,LINE2                ;PRINT LINE 2 OF ERROR MESSAGE
011356  004737  021154                   JSR     PC,LINE3                ;PRINT LINE 3 OF ERROR MESSAGE
011362  004737  023726                   JSR     PC,INCTOT               ;INCREMENT TOTAL ERROR COUNT
011366  004737  022264                   JSR     PC,LINE7                ;PRINT LINE 7 OF ERROR MESSAGE
011372  000207                           RTS     PC                      ;RETURN

                                 ;PROCESS FORMAT ('FER') ERROR

011374  032760  000400  000250   CKFMT:  BIT     #BIT8,$RPER1(R0) ;'HCRC' SET ON ORIGINAL ERROR ?
011402  001402                           BEQ     1$                      ;BR IF NOT SET
011404  000137  011212                   JMP     HCRCER                  ;REPORT HCRC ERROR
011410  004737  022616           1$:     JSR     PC,READDR               ;GET CORRECTED TRACK & SECTOR ADDRSSES
011414  004737  015620                   JSR     PC,READHD               ;READ HEADER
011420  032737  000400  046204           BIT     #BIT8,GENREG+RPER1 ;'HCRC' SET WHEN HEADER READ?
011426  001002                           BNE     2$                      ;BR IF 'HCRC' SET
011430  000137  012374                   JMP     FMTER                   ;NO, ERROR IS 'FMT' ONLY
011434  004737  020344           2$:     JSR     PC,SPOTCK               ;SEE IF ERROR AT BAD SPOT ON THE PACK
011440  000452                           BR      5$                      ;EXIT IF IT IS
011442  004737  020502                   JSR     PC,LINE1                ;PRINT LINE 1 OF ERROR MESSAGE
011446  104414  047216                   DISPLY  ,EM24                   ;HEADER READ ERROR - FMT BIT DROPPED UP
011452  004737  020546                   JSR     PC,LINE2                ;PRINT LINE 2 OF ERROR MESSAGE
011456  004737  021154                   JSR     PC,LINE3                ;PRINT LINE 3 OF ERROR MESSAGE
011462  004737  021630                   JSR     PC,LINE4                ;PRINT LINE 4 OF ERROR MESSAGE
011466  032760  040000  000244           BIT     #BIT14,$RPCS2(R0)  ;'WCE' ERROR ALSO ?
011474  001402                           BEQ     3$                      ;BR IF NOT
011476  004737  021720                   JSR     PC,LINE5                ;DISPLAY WORDS WHICH CAUSED 'WCE'
011502  004737  022020           3$:     JSR     PC,LINE5A               ;DISPLAY HEADER
011506  004737  023606                   JSR     PC,INCSOF               ;INCREMENT SOFT ERROR COUNT
011512  004737  023726                   JSR     PC,INCTOT               ;INCREMENT TOTAL ERROR COUNT
011516  012737  000020  001250           MOV     #BIT4,MASK              ;SET ERROR MASK
011524  012737  000003  001252           MOV     #3,RETRY                ;RETRY LIMIT
011532  004737  015702                   JSR     PC,$RETRY               ;RETRY THE ORDER
011536  000405                           BR      4$                      ;RETRY NOT SUCESSFUL
011540  004737  022202                   JSR     PC,LINE6C               ;PRINT LINE 6C OF ERROR MESSAGE
011544  004737  022264                   JSR     PC,LINE7                ;PRINT LINE 7 OF ERROR MESSAGE
011550  000406                           BR      5$                      ;EXIT
011552  004737  022210           4$:     JSR     PC,LINE6D               ;PRINT LINE 6D OF ERROR MESSAGE
```

```
011556  004737  022264                          JSR     PC,LINE7           ;PRINT LINE 7 OF ERROR MESSAGE
011562  004737  006676                          JSR     PC,REFMT           ;REFORMAT THE ERROR SECTOR
011566  000207                          5$:     RTS     PC                 ;RETURN

                                        ;PROCESS HEADER COMPARE ('HCE') ERROR

011570  032760  000400  000250  CKHCE:  BIT     #BIT8,$RPER1(R0) ;HCRC SET ON ORIGINAL ERROR ?
011576  001402                                  BEQ     1$                 ;BR IF NOT SET
011600  000137  011212                          JMP     HCRCER             ;REPORT HEADER CRC ERROR
011604  004737  022616                  1$:     JSR     PC,READDR          ;GET CURRENT SECTOR & TRACK ADDRS
011610  004737  015620                          JSR     PC,READHD          ;READ HEADER OF CURRENT SECTOR
011614  032737  000400  046204                  BIT     #BIT8,GENREG+RPER1 ;'HCRC' SET ?
011622  001016                                  BNE     3$                 ;BR IF SET
011624  042737  010000  055754                  BIC     #BIT12,CYLDER      ;CLEAR FORMAT BIT FROM HEADER
011632  026037  000272  055754                  CMP     $RPCC(R0),CYLDER   ;CORRECT CYLINDER ?
011640  001402                                  BEQ     2$                 ;BR IF IT IS
011642  000137  012014                          JMP     POSER              ;REPORT POSITIONING ERROR
011646  052737  010000  055754  2$:             BIS     #BIT12,CYLDER      ;RESTORE THE FORMAT BIT
011654  000137  012452                          JMP     HCEER              ;REPORT 'HCE' ERROR
011660  004737  020344                  3$:     JSR     PC,SPOTCK          ;SEE IF ERROR AT BAD SPOT
011664  000452                                  BR      6$                 ;EXIT IF IT IS
011666  004737  020502                          JSR     PC,LINE1           ;PRINT LINE 1 OF ERROR MESSAGE
011672  104414  047264                          DISPLY  ,EM25              ;HEADER READ ERROR - 'HCE' SET
011676  004737  020546                          JSR     PC,LINE2           ;PRINT LINE 2 OF ERROR MESSAGE
011702  004737  021154                          JSR     PC,LINE3           ;PRINT LINE 3 OF ERROR MESSAGE
011706  004737  021630                          JSR     PC,LINE4           ;PRINT LINE 4 OF ERROR MESSAGE
011712  032760  040000  000244                  BIT     #BIT14,$RPCS2(R0)  ;'WCE' ERROR ALSO ?
011720  001402                                  BEQ     4$                 ;BR IF NOT
011722  004737  021720                          JSR     PC,LINE5           ;DISPLAY WORDS WHICH CAUSED 'WCE'
011726  004737  022020                  4$:     JSR     PC,LINE5A          ;PRINT LINE 5 OF ERROR MESSAGE
011732  004737  023606                          JSR     PC,INCSOF          ;INCREMENT SOFT ERROR COUNT
011736  004737  023726                          JSR     PC,INCTOT          ;INCREMENT TOTAL ERROR COUNT
011742  012737  000200  001250                  MOV     #BIT7,MASK         ;SET ERROR MASK
011750  012737  000003  001252                  MOV     #3,RETRY           ;RETRY LIMIT
011756  004737  015702                          JSR     PC,$RETRY          ;RETRY THE ORDER
011762  000405                                  BR      5$                 ;RETRY NOT SUCESSFUL
011764  004737  022202                          JSR     PC,LINE6C          ;PRINT LINE 6C OF ERROR MESSAGE
011770  004737  022264                          JSR     PC,LINE7           ;PRINT LINE 7 OF ERROR MESSAGE
011774  000406                                  BR      6$                 ;EXIT
011776  004737  022210                  5$:     JSR     PC,LINE6D          ;PRINT LINE 6D OF ERROR MESSAGE
012002  004737  022264                          JSR     PC,LINE7           ;PRINT LINE 7 OF ERROR MESSAGE
012006  004737  006676                          JSR     PC,REFMT           ;REFORMAT THE ERROR SECTOR
012012  000207                          6$:     RTS     PC                 ;RETURN

                                        ;REPORT POSSIBLE POSITIONING ERROR

012014  004737  015542                  POSER:  JSR     PC,RECALT          ;RECALIBRATE
012020  004737  020502                          JSR     PC,LINE1           ;PRINT LINE 1 OF ERROR MESSAGE
012024  104414  050612                          DISPLY  ,EM51              ;PROGRAM DETECTED POSITIONING ERROR
012030  004737  020546                          JSR     PC,LINE2           ;PRINT LINE 2 OF ERROR MESSAGE
012034  004737  021202                          JSR     PC,LINE3C          ;PRINT LINE 3C OF ERROR MESSAGE
012040  052737  010000  055754                  BIS     #BIT12,CYLDER      ;RESTORE THE FORMAT BIT
012046  004737  022020                          JSR     PC,LINE5A          ;PRINT LINE 5A OF THE ERROR MESSAGE
012052  004737  023702                          JSR     PC,INCMIS          ;INCREMENT MISPOSITIONING COUNT
012056  004737  023726                          JSR     PC,INCTOT          ;INCREMENT TOTAL ERROR COUNT
012062  004737  022412                          JSR     PC,LINE7A          ;PRINT LINE 7A OF ERROR MESSAGE
012066  000207                                  RTS     PC                 ;EXIT
```

```

                                         ;REPORT 'OPI' ERROR

012070  004737  020344           OPIER:  JSR     PC,SPOTCK       ;SEE IF ERROR AT BAD SPOT
012074  000207                           RTS     PC              ;RETURN IF IT IS
012076  004737  020502                   JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
012102  104414  047461                   DISPLY  ,EM31           ;'OPI' ERROR
012106  004737  020546                   JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
012112  004737  021154                   JSR     PC,LINE3        ;PRINT LINE 3 OF ERROR MESSAGE
012116  004737  021630                   JSR     PC,LINE4        ;PRINT LINE 4 OF ERROR MESSAGE
012122  004737  023726                   JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
012126  012737  020000  001250           MOV     #BIT13,MASK     ;ERROR MASK
012134  012737  000003  001252   OPIER1: MOV     #3,RETRY        ;RETRY LIMIT
012142  004737  015702                   JSR     PC,$RETRY       ;RETRY THE ORDER
012146  000405                           BR      1$              ;RETRY UNSUCESSFUL
012150  004737  022202                   JSR     PC,LINE6C       ;PRINT LINE 6C OF ERROR MESSAGE
012154  004737  022264                   JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
012160  000207                           RTS     PC              ;EXIT
012162  004737  022210           1$:     JSR     PC,LINE6D       ;PRINT LINE 6D OF ERROR MESSAGE
012166  004737  022264                   JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
012172  004737  006676                   JSR     PC,REFMT        ;REFORMAT THE ERROR SECTOR
012176  000207                           RTS     PC              ;RETURN

                                         ;REPORT 'DTE' ERROR

012200  004737  020344           DTEER:  JSR     PC,SPOTCK       ;SEE IF ERROR AT BAD SPOT
012204  000207                           RTS     PC              ;RETURN IF IT IS
012206  004737  020502                   JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
012212  104414  047524                   DISPLY  ,EM32           ;'DTE' ERROR
012216  000137  010224                   JMP     DCKER1          ;FINISH PROCESSING THE 'DTE' ERROR

                                         ;REPORT 'PAR' ERROR

012222  004737  020502           PARER:  JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
012226  104414  047557                   DISPLY  ,EM33           ;REPORT 'PAR'
012232  004737  020546                   JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
012236  004737  021260                   JSR     PC,LINE3E       ;PRINT LINE 3E OF ERROR MESSAGE
012242  004737  021630                   JSR     PC,LINE4        ;PRINT LINE 4 OF ERROR MESSAGE
012246  004737  023726                   JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
012252  012737  000010  001250           MOV     #BIT03,MASK     ;ERROR MASK
012260  012737  000003  001252           MOV     #3,RETRY        ;RETRY LIMIT
012266  004737  015702                   JSR     PC,$RETRY       ;RETRY ORDER
012272  000405                           BR      2$              ;RETRY UNSUCESSFUL
012274  004737  022202                   JSR     PC,LINE6C       ;RETRY SUCESSFUL
012300  004737  022264           1$:     JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
012304  000207                           RTS     PC              ;EXIT
012306  004737  022210           2$:     JSR     PC,LINE6D       ;PRINT LINE 6D OF ERROR MESSAGE
012312  000772                           BR      1$              ;FINISH ERROR MESSAGE

                                         ;REPORT 'IAE' ERROR

012314  004737  020502           IAEER:  JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
012320  104414  047676                   DISPLY  ,EM35           ;REPORT 'IAE'
012324  004737  020546                   JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
012330  004737  021346                   JSR     PC,LINE3F       ;PRINT LINE 3F OF ERROR MESSAGE
012334  004737  023726                   JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
012340  004737  022264                   JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
```

```
012344  000207                           RTS     PC              ;RETURN

                                 ;REPORT 'WLE' ERROR

012346  004737  020502           WLEER:  JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
012352  104414  047734                   DISPLY  ,EM36           ;REPORT 'WLE'
012356  004737  020546                   JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
012362  004737  023726                   JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
012366  004737  022264                   JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
012372  000207                           RTS     PC              ;RETURN

                                 ;REPORT FORMAT ERROR

012374  004737  020502           FMTER:  JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
012400  104414  047345                   DISPLY  ,EM26           ;FORMAT ERROR
012404  004737  020546                   JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
012410  004737  021154                   JSR     PC,LINE3        ;PRINT LINE 3 OF ERROR MESSAGE
012414  004737  021630                   JSR     PC,LINE4        ;PRINT LINE 4 OF ERROR MESSAGE
012420  032760  040000  000244           BIT     #BIT14,$RPCS2(R0)  ;'WCE' ERROR ALSO ?
012426  001402                           BEQ     1$              ;BR IF NOT
012430  004737  021720                   JSR     PC,LINE5        ;DISPLAY WORDS WHICH CAUSED 'WCE'
012434  004737  022020           1$:     JSR     PC,LINE5A       ;PRINT LINE 5A OF ERROR MESSAGE
012440  004737  023726                   JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
012444  004737  022264                   JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
012450  000207                           RTS     PC

                                 ;REPORT HEADER COMPARE ERROR

012452  004737  020502           HCEER:  JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
012456  104414  047372                   DISPLY  ,EM27           ;HEADER COMPARE ERROR
012462  004737  020546                   JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
012466  004737  021154                   JSR     PC,LINE3        ;PRINT LINE 3 OF ERROR MESSAGE
012472  004737  021630                   JSR     PC,LINE4        ;PRINT LINE 4 OF ERROR MESSAGE
012476  032760  040000  000244           BIT     #BIT14,$RPCS2(R0)  ;'WCE' ERROR ALSO ?
012504  001402                           BEQ     1$              ;BR IF NOT
012506  004737  021720                   JSR     PC,LINE5        ;DISPLAY WORDS WHICH CAUSED 'WCE'
012512  004737  022020           1$:     JSR     PC,LINE5A       ;PRINT LINE 5A OF ERROR MESSAGE
012516  004737  023726                   JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
012522  004737  022264                   JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
012526  004737  006676                   JSR     PC,REFMT        ;REFORMAT THE ERROR SECTOR
012532  000207                           RTS     PC              ;RETURN

                                 ;PROCESS CONTROL/INTERFACE TRANSFER ERROR

012534  004737  020502           TRFER:  JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
012540  104414  050047                   DISPLY  ,EM40           ;RH11 OR UNIBUS TRANSFER ERROR
012544  004737  020546                   JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
012550  004737  021154                   JSR     PC,LINE3        ;PRINT LINE 3 OF ERROR MESSAGE
012554  004737  021630                   JSR     PC,LINE4        ;PRINT LINE 4 OF ERROR MESSAGE
012560  004737  023726                   JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
012564  032760  121400  000244           BIT     #BIT15!BIT13!BIT9!BIT8,$RPCS2(R0)  ;'DLT','UPE','MXF', 'MDPE' SET ?
012572  001415                           BEQ     2$              ;BR IF NONE SET
012574  012737  000003  001252           MOV     #3,RETRY        ;RETRY LIMIT
012602  005037  001250                   CLR     MASK            ;CLEAR ERROR MASK
012606  004737  015702                   JSR     PC,$RETRY       ;RETRY THE OPERATION
012612  000403                           BR      1$              ;RETURN HERE IF RETRY UNSUCESSFUL
012614  004737  022202                   JSR     PC,LINE6C       ;PRINT LINE 6C OF ERROR MESSAGE
```

```
012620  000402                           BR      2$              ;FINISH THE ERROR REPORT
012622  004737  022210           1$:     JSR     PC,LINE6D       ;PRINT LINE 6D OF ERROR MESSAGE
012626  004737  022264           2$:     JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
012632  000207                           RTS     PC


                                 ;PROCESS 'SKI' OR 'OCYL' ERRORS

012634  004737  020502           SKIER:  JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
012640  104414  050523                   DISPLAY ,EM50           ;'SKI' OR 'OCYL' ERROR
012644  004737  020546                   JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
012650  004737  021170                   JSR     PC,LINE3B       ;PRINT LINE 3B OF ERROR MESSAGE
012654  004737  023726                   JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
012660  004737  023656                   JSR     PC,INCSKI       ;INCREMENT 'SKI' OR 'OCYL' ERROR COUNT
012664  004737  022412                   JSR     PC,LINE7A       ;PRINT LINE 7A OF ERROR MESSAGE
012670  000207                           RTS     PC


                                 ;REPORT WRITE CLOCK FAILURE ('WCF')

012672  004737  020502           WCFER:  JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
012676  104414  047634                   DISPLAY ,EM34           ;REPORT WRITE CLOCK FAILURE
012702  004737  020546                   JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
012706  004737  021162                   JSR     PC,LINE3A       ;PRINT LINE 3A OF ERROR MESSAGE
012712  004737  021630                   JSR     PC,LINE4        ;PRINT LINE 4 OF ERROR MESSAGE
012716  004737  023726                   JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
012722  004737  015344                   JSR     PC,PRTBAD       ;SEE IF BAD SECTOR TO BE PRINTED
012726  012737  000003  001252           MOV     #3,RETRY        ;RETRY COUNT
012734  012737  000040  001250           MOV     #BIT05,MASK     ;ERROR MASK
012742  004737  015702                   JSR     PC,$RETRY       ;RETRY THE ORDER
012746  000405                           BR      2$              ;RETURN HERE IF RETRY UNSUCESSFUL
012750  004737  022202                   JSR     PC,LINE6C       ;PRINT LINE 6C OF ERROR MESSAGE
012754  004737  022264           1$:     JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
012760  000207                           RTS     PC
012762  004737  022210           2$:     JSR     PC,LINE6D       ;PRINT LINE 6D OF ERROR MESSAGE
012766  000772                           BR      1$


                                 ;PROCESS DRIVE UNSAFE ERROR

012770  004737  020502           UNSAF:  JSR     PC,LINE1        ;PRINT LINE 1 OF ERROR MESSAGE
012774  104414  050655                   DISPLAY ,EM60           ;REPORT DRIVE UNSAFE
013000  004737  020546                   JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
013004  004737  021154                   JSR     PC,LINE3        ;PRINT LINE 3 OF ERROR MESSAGE
013010  004737  023726                   JSR     PC,INCTOT       ;INCREMENT TOTAL ERROR COUNT
013014  012737  000003  001252           MOV     #3,RETRY        ;RETRY COUNT
013022  004737  015702                   JSR     PC,$RETRY       ;RETRY THE ORDER
013026  000403                           BR      1$              ;RETRY WAS UNSUCESSFUL
013030  004737  022202                   JSR     PC,LINE6C       ;PRINT LINE 6C OF ERROR MESSAGE
013034  000402                           BR      2$              ;CONTINUE WITH ERROR REPORT
013036  004737  022210           1$:     JSR     PC,LINE6D       ;PRINT LINE 6D OF ERROR MESSAGE
013042  004737  022264           2$:     JSR     PC,LINE7        ;PRINT LINE 7 OF ERROR MESSAGE
013046  000207                           RTS     PC              ;RETURN


                                 ;REPORT AN 'UNKNOWN' DATA PATTERN

013050  105737  001300           NOMTCH: TSTB    FRSTER          ;FIRST ERROR IN THE SECTOR ?
013054  001013                           BNE     1$              ;BR IF NOT OR IF PROCESSING 'DCKER'
013056  004737  020502                   JSR     PC,LINE1        ;TYPE LINE 1 OF ERROR MESSAGE
013062  104414  050232                   DISPLAY ,EM43           ;'CAN'T MATCH DATA WITH PATTERN'
```

```
    512 013066  004737  020546                  JSR     PC,LINE2            ;PRINT LINE 2 OF ERROR MESSAGE
    513 013072  004737  021162                  JSR     PC,LINE3A           ;PRINT LINE 3A OF ERROR MESSAGE
    514 013076  004737  021630                  JSR     PC,LINE4            ;PRINT LINE 4 OF ERROR MESSAGE
    515 013102  000404                          BR      2$                  ;CONTINUE PROCESSING ERROR
    516 013104  104414  050232          1$:     DISPLY  ,EM43               ;'CAN'T MATCH DATA WITH PATTERN'
    517 013110  104414  001165                  DISPLY  ,$CRLF              ;CR-LF
    518 013114  104414  052710          2$:     DISPLY  ,LIN9I              ;HEADER FOR DATA PRINTOUT
    526 013120  010146                          MOV     R1,-(SP)            ;ADDRESS OF WORD 1
        013122  004737  022544                  JSR     PC,LINOCT           ;TYPE WORD 1
        013126  104414  053363                  DISPLY  ,LINSP              ;SPACES
        013132  012146                          MOV     (R1)+,-(SP)         ;ADDRESS OF WORD 1
        013134  004737  022544                  JSR     PC,LINOCT           ;TYPE WORD 1
        013140  104414  001165                  DISPLY  ,$CRLF              ;CR-LF
        013144  010146                          MOV     R1,-(SP)            ;ADDRESS OF WORD 2
        013146  004737  022544                  JSR     PC,LINOCT           ;TYPE WORD 2
        013152  104414  053363                  DISPLY  ,LINSP              ;SPACES
        013156  012146                          MOV     (R1)+,-(SP)         ;ADDRESS OF WORD 2
        013160  004737  022544                  JSR     PC,LINOCT           ;TYPE WORD 2
        013164  104414  001165                  DISPLY  ,$CRLF              ;CR-LF
        013170  010146                          MOV     R1,-(SP)            ;ADDRESS OF WORD 3
        013172  004737  022544                  JSR     PC,LINOCT           ;TYPE WORD 3
        013176  104414  053363                  DISPLY  ,LINSP              ;SPACES
        013202  012146                          MOV     (R1)+,-(SP)         ;ADDRESS OF WORD 3
        013204  004737  022544                  JSR     PC,LINOCT           ;TYPE WORD 3
        013210  104414  001165                  DISPLY  ,$CRLF              ;CR-LF
        013214  010146                          MOV     R1,-(SP)            ;ADDRESS OF WORD 4
        013216  004737  022544                  JSR     PC,LINOCT           ;TYPE WORD 4
        013222  104414  053363                  DISPLY  ,LINSP              ;SPACES
        013226  012146                          MOV     (R1)+,-(SP)         ;ADDRESS OF WORD 4
        013230  004737  022544                  JSR     PC,LINOCT           ;TYPE WORD 4
        013234  104414  001165                  DISPLY  ,$CRLF              ;CR-LF
    527 013240  062701  000770                  ADD     #<252.*2.>,R1       ;INCREMENT BUFFER POINTER
    528 013244  005002                          CLR     R2                  ;CLEAR 'WORDS TO COMPARE' COUNT IN R2
    529 013246  112737  000001  001300          MOVB    #1,FRSTER           ;SET 'NOT FIRST ERROR' INDICATOR
    530 013254  112737  177777  001301          MOVB    #-1,FRSTER+1        ;SET ERROR FOUND INDICATOR
    531 013262  013737  001414  001310          MOV     CMPLMT,LIMIT        ;RESET THE COMPARE ERROR TYPEOUT LIMIT
    532 013270  000207                          RTS     PC                  ;RETURN
    533
    534                                 ;CHECK ERROR BITS IN THE RH11 & RP04/5/6 REGISTERS
    535
    536 013272  032760  060000  000234  CKERR:  BIT     #60000,$RPCS1(R0)   ;SEE IF 'TRE' OR 'MCPE' SET
    537 013300  001015                          BNE     1$                  ;BR IT EITHER SET
    538 013302  032760  177400  000244          BIT     #177400,$RPCS2(R0)  ;SEE IF ERROR BITS IN CS2 SET
    539 013310  001011                          BNE     1$                  ;BR IF ANY SET
    540 013312  005760  000250                  TST     $RPER1(R0)          ;ANY BITS SET IN ER1
    541 013316  001006                          BNE     1$                  ;BR IF ANY SET
    542 013320  005760  000274                  TST     $RPER2(R0)          ;ANY BITS SET IN ER2 ?
    543 013324  001003                          BNE     1$                  ;BR IF ANY SET
    544 013326  005760  000276                  TST     $RPER3(R0)          ;ANY BITS SET IN ER3 ?
    545 013332  001416                          BEQ     2$                  ;BR IF NONE SET
    546 013334  004737  020502          1$:     JSR     PC,LINE1            ;PRINT LINE 1 OF ERROR MESSAGE
    547 013340  104414  050277                  DISPLY  ,EM44               ;ERROR BITS SET, BUT 'SC' OR 'TRE' NOT SET
    548 013344  004737  020546                  JSR     PC,LINE2            ;PRINT LINE 2 OF ERROR MESSAGE
    549 013350  004737  021154                  JSR     PC,LINE3            ;PRINT LINE 3 OF ERROR MESSAGE
    550 013354  004737  021630                  JSR     PC,LINE4            ;PRINT LINE 4 OF ERROR MESSAGE
    551 013360  004737  023726                  JSR     PC,INCTOT           ;INCREMENT TOTAL ERROR COUNT
    552 013364  004737  022264                  JSR     PC,LINE7            ;PRINT LINE 7 OF ERROR MESSAGE
```

```
013370  000207                          2$:     RTS     PC                      ;RETURN

                                        ;CHECK BUS ADDRESS REGISTER & WORD COUNT REGISTER

013372  005760  000236                  CKBUS:  TST     $RPWC(R0)               ;CHECK WORD COUNT
013376  001010                                  BNE     1$                      ;BR IF NOT ZERO
013400  016046  000020                          MOV     $WRDL(R0),-(SP)         ;WORD LENGTH
013404  006316                                  ASL     (SP)                    ;CHANGE INTO BYTE COUNT
013406  066016  000006                          ADD     $BUF(R0),(SP)           ;ADD THE STARTING LOCATION
013412  022660  000240                          CMP     (SP)+,$RPBA(R0)         ;BUFFER ADDRESS PROPER ?
013416  001416                                  BEQ     2$                      ;BR IF OK
013420  004737  020502                  1$:     JSR     PC,LINE1                ;PRINT LINE 1 OF ERROR MESSAGE
013424  104414  050105                          DISPLY  ,EM41                   ;BUS ADDRESS OR WORD COUNT INCORRECT
013430  004737  020546                          JSR     PC,LINE2                ;PRINT LINE 2 OF ERROR MESSAGE
013434  004737  021212                          JSR     PC,LINE3D               ;PRINT LINE 3D OF ERROR MESSAGE
013440  004737  021630                          JSR     PC,LINE4                ;PRINT LINE 4 OF ERROR MESSAGE
013444  004737  023726                          JSR     PC,INCTOT               ;INCREMENT TOTAL ERROR COUNT
013450  004737  022264                          JSR     PC,LINE7                ;PRINT LINE 7 OF ERROR MESSAGE
013454  000207                          2$:     RTS     PC

                                        ;COMPARE THE BUFFER

013456  132760  000004  000024          CMPAR:  BITB    #BIT02,$CODE(R0)          ;SEE IF READ ORDER
013464  001001                                  BNE     1$                      ;BR IF IT IS
013466  000207                                  RTS     PC                      ;RETURN
013470  005037  001300                  1$:     CLR     FRSTER                  ;CLEAR 'FIRST ERROR' INDICATOR
013474  032777  000002  165436          CMPARD: BIT     #SW01,@SWR              ;IS SWITCH 1 SET?
013502  001401                                  BEQ     1$                      ;BR IF NOT
013504  000207                                  RTS     PC                      ;YES, DON'T COMPARE
013506  005037  001306                  1$:     CLR     ERCTR                   ;CLEAR THE ERROR COUNTER
013512  016001  000006                          MOV     $BUF(R0),R1             ;BUFFER ADDRESS
013516  016037  000020  001312                  MOV     $WRDL(R0),CMCNT          ;WORD COUNT TO WORKING LOCATION
013524  066037  000236  001312                  ADD     $RPWC(R0),CMCNT          ;CALCULATE ACTUAL WORDS TRANSFERED
013532  016037  000012  001314                  MOV     $CYL(R0),CMCYL          ;CYLINDER ADDRESS WORKING LOCATION
013540  052737  010000  001314                  BIS     #BIT12,CMCYL            ;SET FORMAT BIT
013546  016037  000010  001316                  MOV     $SEC(R0),CMSEC          ;SECTOR & TRACK ADDRESSES TO WORKING LOCNS
013554  013737  001414  001310                  MOV     CMPLMT,LIMIT            ;DISPLAY LIMIT
013562  005237  001310                          INC     LIMIT                   ;CONVERT PARAMETER INTO LIMIT VALUE
013566  012737  177777  001276          CMSTR:  MOV     #-1,ZROIND              ;CLEAR THE 'ZERO'S' INDICATOR
013574  005037  001302                          CLR     SAVER1                  ;CLEAR THE R1 SAVE WORD
013600  005037  001304                          CLR     SAVER5                  ;CLEAR THE R5 SAVE WORD
013604  023760  001312  000022                  CMP     CMCNT,$SSEC(R0)          ;IS BUFFER SIZE GREATER THAN ONE SECTOR ?
013612  101003                                  BHI     1$                      ;BR IF IT IS
013614  013702  001312                          MOV     CMCNT,R2                ;LESS THAN, USE REMAINING BUFFER
013620  000402                                  BR      2$                      ;
013622  016002  000022                  1$:     MOV     $SSEC(R0),R2            ;COMPARE SECTOR
013626  166037  000022  001312          2$:     SUB     $SSEC(R0),CMCNT         ;DECREMENT WORD COUNT
013634  126027  000024  000005                  CMPB    $CODE(R0),#5            ;READ HEADER & DATA?
013642  001036                                  BNE     CMDAT                   ;BR IF NOT
013644  023721  001314                  CMHED:  CMP     CMCYL,(R1)+             ;CHECK CYLINDER
013650  001402                                  BEQ     1$                      ;BR IF COMPARE OK
013652  004737  013726                          JSR     PC,5$                   ;REPORT ERROR
013656  023721  001316                  1$:     CMP     CMSEC,(R1)+             ;COMPARE SECTOR & TRACK
013662  001402                                  BEQ     2$                      ;BR IF EQ
013664  004737  013726                          JSR     PC,5$                   ;REPORT ERROR
013670  005721                          2$:     TST     (R1)+                   ;1ST KEY WORD ZERO?
013672  001402                                  BEQ     3$                      ;BR IF IT IS
```

```
013674  004737  013726                    JSR     PC,5$           ;REPORT ERROR
013700  005721                    3$:     TST     (R1)+           ;CHECK 2ND KEY WORD
013702  001402                            BEQ     4$              ;BR IF ZERO
013704  004737  013726                    JSR     PC,5$           ;REPORT THE ERROR
013710  162702  000004            4$:     SUB     #4,R2           ;SUBTRACT HEADER LENGTH FROM SIZE
013714  003530                            BLE     CMPRX           ;BR IF FINISHED
013716  022702  000004                    CMP     #4,R2           ;SEE IF AT LEAST 4 MORE WORDS TO CHECK
013722  101125                            BHI     CMPRX           ;BR IF NOT
013724  000405                            BR      CMDAT           ;COMPARE THE DATA PORTION
013726  005237  001306            5$:     INC     ERCTR           ;INCREMENT THE ERROR COUNT
013732  004737  014204                    JSR     PC,CMPRT        ;REPORT THE COMPARISON ERROR
013736  000207                            RTS     PC              ;CHECK THE REST OF THE HEADER
013740  004737  014526            CMDAT:  JSR     PC,MATCH        ;FIND THE PATTERN
013744  000403                            BR      2$              ;FOUND A PATTERN
013746  004737  013050                    JSR     PC,NOMTCH       ;RETURN HERE IF NO MATCH WITH PATTERN MADE
013752  000456                            BR      8$              ;BYPASS COMPARE ROUTINE
013754  011405                    2$:     MOV     (R4),R5         ;ADDRESS OF PATTERN ADDRESS IN R4
013756  012703  000020                    MOV     #20,R3          ;R3 IS PATTERN POS COUNTER
013762  022125                    3$:     CMP     (R1)+,(R5)+     ;COMPARE BUFFER WITH PATTERN
013764  001016                            BNE     5$              ;BR IF NOT EQUAL
013766  005737  001306                    TST     ERCTR           ;ERRORS DETECTED ?
013772  001406                            BEQ     4$              ;BR IF NO ERRORS
013774  032777  000010  165136            BIT     #SW3,@SWR       ;SWITCH 3 SET ?
014002  001402                            BEQ     4$              ;BR IF NOT SET
014004  004737  014204                    JSR     PC,CMPRT        ;DISPLAY THE WORD
014010  005302                    4$:     DEC     R2              ;DECREMENT SIZE COUNT
014012  001436                            BEQ     8$              ;BR WHEN AT END
014014  005303                            DEC     R3              ;DECREMENT PATT POS COUNT
014016  001361                            BNE     3$              ;BR IF NOT AT END OF PATT
014020  000755                            BR      2$              ;RESTART THE PATTERN
014022  005761  177776            5$:     TST     -2(R1)          ;IS MISCOMPARED CHARACTER=0
014026  001410                            BEQ     6$              ;BR IF YES
014030  012737  177777  001276            MOV     #-1,ZROIND      ;SET NON-ZERO MISCOMPARED INDCATOR
014036  005237  001306                    INC     ERCTR           ;INCREMENT THE ERROR COUNTER
014042  004737  014204                    JSR     PC,CMPRT        ;REPORT ERROR
014046  000760                            BR      4$              ;CONTINUE COMPARE
014050  105737  001300            6$:     TSTB    FRSTER          ;FIRST ERROR?
014054  100407                            BMI     7$              ;BR IF NOT
014056  005037  001276                    CLR     ZROIND          ;SET THE ZERO INDICATOR
014062  010137  001302                    MOV     R1,SAVER1       ;SAVE CURRENT R1
014066  010537  001304                    MOV     R5,SAVER5       ;SAVE CURRENT R5
014072  000746                            BR      4$              ;CONTINUE COMPARE
014074  005737  001276            7$:     TST     ZROIND          ;ANY MISCOMPARIONS NOT ZEROS ?
014100  001743                            BEQ     4$              ;BR IF NONE-ALL ERRORS=ZERO
014102  004737  014204                    JSR     PC,CMPRT        ;REPORT ERROR
014106  000740                            BR      4$              ;CONTINUE COMPARING
014110  005737  001312            8$:     TST     CMCNT           ;AT END OF BUFFER
014114  003430                            BLE     CMPRX           ;BR IF AT END
014116  126027  000024  000005            CMPB    $CODE(R0),#5    ;SEE IF READ HEADER & DATA
014124  001220                            BNE     CMSTR           ;BR IF NOT
014126  105237  001316                    INCB    CMSEC           ;INCREMENT SECTOR
014132  123727  001316  000026            CMPB    CMSEC,#22.      ;SECTOR GREATER THAN MAX ?
014140  103612                            BLO     CMSTR           ;BR IF NOT GREATER THAN MAX
014142  105037  001316                    CLRB    CMSEC           ;CLEAR SECTOR ADDRESS
014146  105237  001317                    INCB    CMTRK           ;INCREMENT TRACK
014152  123727  001317  000023            CMPB    CMTRK,#19.      ;TRACK GREATER THAN MAX ?
014160  103602                            BLO     CMSTR           ;BR IF NOT GREATER
```

```
    667 014162  105037  001317                    CLRB    CMTRK             ;RESET TRACK ADDRESS
    668 014166  005237  001314                    INC     CMCYL             ;INCREMENT CYLINDER ADDRESS
    669 014172  000137  013566                    JMP     CMSTR             ;CONTINUE WITH COMPARE
    670 014176  004737  014460            CMPRX:  JSR     PC,ENDCMP         ;PRINT LAST LINE IF ERRORS
    671 014202  000207                            RTS     PC
    672
    673                                   ;TYPE DATA COMPARE ERRORS
    674
    675 014204  005737  001302            CMPRT:  TST     SAVER1            ;PRINT SAVED VALUES ?
    676 014210  001010                            BNE     2$                ;BR IF NOT
    677 014212  105737  001300                    TSTB    FRSTER            ;FIRST ERROR?
    678 014216  100402                            BMI     1$                ;BR IF NOT
    679 014220  004737  014300                    JSR     PC,4$             ;PRINT INITIAL MESSAGE INFO
    680 014224  004737  014362            1$:     JSR     PC,8$             ;PRINT REMAINDER OF MESSAGE
    681 014230  000422                            BR      3$                ;EXIT
    682 014232                            2$:
        014232  010146                            MOV     R1,-(SP)          ;;PUSH R1 ON STACK
        014234  010546                            MOV     R5,-(SP)          ;;PUSH R5 ON STACK
    683 014236  013701  001302                    MOV     SAVER1,R1         ;DISPLAY SAVED R1
    684 014242  013705  001304                    MOV     SAVER5,R5         ;DISPLAY SAVED R5
    685 014246  004737  014300                    JSR     PC,4$             ;PRINT INITIAL MESSAGE INFO
    686 014252  004737  014362                    JSR     PC,8$             ;PRINT SAVED VALUES
    687 014256  005037  001302                    CLR     SAVER1            ;CLEAR SAVED REGISTER INDICATORS
    688 014262  005037  001304                    CLR     SAVER5            ;CLEAR THE OTHER ONE
    689 014266  012605                            MOV     (SP)+,R5          ;;POP STACK INTO R5
        014270  012601                            MOV     (SP)+,R1          ;;POP STACK INTO R1
    690 014272  004737  014362                    JSR     PC,8$             ;PRINT REMAINDER OF MESSAGE
    691 014276  000207                    3$:     RTS     PC                ;RETURN
    692 014300  105737  001300            4$:     TSTB    FRSTER            ;FIRST ERROR ?
    693 014304  100425                            BMI     7$                ;BR IF NOT
    694 014306  001013                            BNE     5$                ;BR IF FIRST ERROR AND PROCESSING 'DCK' ERROR
    695 014310  004737  020502                    JSR     PC,LINE1          ;PRINT LINE 1 OF ERROR MESSAGE
    696 014314  104414  050151                    DISPLY  ,EM42             ;DATA COMPARE ERROR
    697 014320  004737  020546                    JSR     PC,LINE2          ;PRINT LINE 2 OF ERROR MESSAGE
    698 014324  004737  021162                    JSR     PC,LINE3A         ;PRINT LINE 3A OF ERROR MESSAGE
    699 014330  004737  021630                    JSR     PC,LINE4          ;PRINT LINE 4 OF ERROR MESSAGE
    700 014334  000404                            BR      6$                ;GO TO TYPE HEADER
    701 014336  104414  052605            5$:     DISPLY  ,LIN9B            ;HEADER MESSAGE OF PROCESSING 'DCK' ERROR
    702 014342  104414  001165                    DISPLY  ,$CRLF            ;CR-LF
    703 014346  104414  052634            6$:     DISPLY  ,LIN9H            ;DISPLAY HEADER
    704 014352  012737  177777  001300            MOV     #-1,FRSTER        ;SET ERROR FLAG
    705 014360  000207                    7$:     RTS     PC                ;RETURN
    706 014362  005737  001310            8$:     TST     LIMIT             ;TYPEOUT LIMIT REACHED ?
    707 014366  001403                            BEQ     9$                ;BR IF IT HAS
    708 014370  005337  001310                    DEC     LIMIT             ;DECREMENT LIMIT COUNTER
    709 014374  001005                            BNE     10$               ;BR IF NOT AT LIMIT
    710 014376  032777  000200  164534    9$:     BIT     #SW07,@SWR        ;PRINT ALL DATA COMPARE ERRORS ?
    711 014404  001001                            BNE     10$               ;BR IF YES
    712 014406  000207                            RTS     PC                ;RETURN
    713 014410  010146                    10$:    MOV     R1,-(SP)          ;BUFFER ADDRESS
    714 014412  162716  000002                    SUB     #2,(SP)           ;ADJUST ADDRESS
    715 014416  004737  022544                    JSR     PC,LINOCT         ;TYPE IT
    716 014422  104414  053363                    DISPLY  ,LINSP            ;2 SPACES
    717 014426  016546  177776                    MOV     -2(R5),-(SP)      ;PUT GOOD DATA ON THE STACK
    718 014432  004737  022544                    JSR     PC,LINOCT         ;TYPE IT
    719 014436  104414  053363                    DISPLY  ,LINSP            ;2 SPACES
    720 014442  016146  177776                    MOV     -2(R1),-(SP)      ;BAD DATA
```

```
014446  004737  022544                  JSR     PC,LINOCT           ;TYPE IT
014452  104414  001165                  DISPLY  ,$CRLF              ;CR-LF
014456  000207                          RTS     PC                  ;RETURN

                                ;LAST LINE OF COMPARE ERROR REPORTING

014460  105737  001301          ENDCMP: TSTB    FRSTER+1            ;ANY COMPARE ERRORS FOUND ?
014464  001417                          BEQ     2$                  ;BR IF NOT
014466  005737  001306                  TST     ERCTR               ;SEE HOW MANY ERRORS
014472  001410                          BEQ     1$                  ;BR IF ONLY CAN'T MATCH PATTERN
014474  104414  052727                  DISPLY  ,LIN9E              ;'NUMBER OF ERRORS='
014500  013746  001306                  MOV     ERCTR,-(SP)         ;NUMBER OF ERRORS
014504  004737  022576                  JSR     PC,LINDEC           ;TYPE IT
014510  104414  001165                  DISPLY  ,$CRLF              ;CR-LF
014514  004737  023726          1$:     JSR     PC,INCTOT           ;INCREMENT TOTAL ERROR COUNT
014520  004737  022264                  JSR     PC,LINE7            ;PRINT LINE 7 OF ERROR MESSAGE
014524  000207                  2$:     RTS     PC                  ;RETURN


                                ;ROUTINE TO MATCH THE DATA WITH A PATTERN
                                ;CALL:
                                ;       MOV     #BUFFER,R1          ;BUFFER ADDRESS
                                ;       JSR     PC,MATCH
                                ;       RETURN1                     ;PATTERN ADDRESS IN R4
                                ;       RETURN2                     ;COULDN'T MATCH PATTERN

014526  010146                  MATCH:  MOV     R1,-(SP)            ;SAVE R1 ON THE STACK
014530  012704  000044                  MOV     #44,R4              ;PATTERN TABLE INDEX
014534  011601                  1$:     MOV     (SP),R1             ;RELOAD R1
014536  162704  000002                  SUB     #2,R4               ;DECREMENT INDEX
014542  016405  002762                  MOV     STNDAT(R4),R5       ;ADDRESS OF PATTERN ADDRESS
014546  001411                          BEQ     3$                  ;BR IF ALL PATTERNS CHECKED AND NO MATCH
                                                                    ;FOUND
014550  012703  000004                  MOV     #4,R3               ;NUMBER OF LOCATIONS TO CHECK
014554  022125                  2$:     CMP     (R1)+,(R5)+         ;COMPARE THE BUFFER AGAINST THE PATTERN
014556  001366                          BNE     1$                  ;BR IF NOT EQUAL, TRY NEXT PATTERN
014560  005303                          DEC     R3                  ;FINISHED CHECKING?
014562  001374                          BNE     2$                  ;BR IF NOT FINISHED
014564  062704  002762                  ADD     #STNDAT,R4          ;MAKE PATTERN ADDRESS ABSOLUTE
014570  000403                          BR      4$                  ;EXIT
014572  062766  000002  000002  3$:     ADD     #2,2(SP)            ;INCREMENT RETURN ADDRESS
014600  012601                  4$:     MOV     (SP)+,R1            ;RESTORE R1
014602  000207                          RTS     PC                  ;RETURN

                                ;USE ECC TO CORRECT THE DATA ERROR

014604  016037  000240  001322  ECC:    MOV     $RPBA(R0),ECSEC     ;ADDRESS OF LAST LOCN XFERED
014612  016046  000236                  MOV     $RPWC(R0),-(SP)     ;ACT WORDS XFERED (2'S COMP)
014616  066016  000020                  ADD     $WRDL(R0),(SP)      ;ADD WORDS REQUESTED
014622  005046                          CLR     -(SP)               ;CLEAR NEXT STACK LOCN
014624  016046  000022                  MOV     $SSEC(R0),-(SP)     ;SECTOR SIZE
014630  004737  027366                  JSR     PC,LINKDV           ;DIVIDE WORDS XFERED BY SECTOR SIZE
014634  005716                          TST     (SP)                ;PARTIAL SECTOR XFERED ?
014636  001413                          BEQ     1$                  ;BR IF NOT
014640  006316                          ASL     (SP)                ;CONVERT INTO NUMBER OF BYTES
014642  161637  001322                  SUB     (SP),ECSEC          ;SUBTRACT SECTOR RESIDUE
014646  126027  000024  000005          CMPB    $CODE(R0),#5        ;WAS OP READ HEAD & DATA
```

```
014654  001007                          BNE     2$                      ;BR IF NOT
014656  062737  000010  001322          ADD     #8.,ECSEC               ;ADD HEADER SIZE (IN BYTES) BACK IN
014664  000403                          BR      2$                      ;GO ADJUST THE STACK POINTER
014666  162737  001000  001322  1$:     SUB     #1000,ECSEC             ;SUBTRACT SECTOR DATA FIELD SIZE (IN BYTES)
014674  062706  000004          2$:     ADD     #4,SP                   ;ADJUST THE STACK POINTER
014700  016037  000300  001320          MOV     $RPEC1(R0),ECBIT  ;ECC POSITION COUNT
014706  005337  001320                  DEC     ECBIT                   ;ADJUST THE POSITION COUNT
014712  013737  001320  001330          MOV     ECBIT,ECWRD             ;LOAD THE WORD COUNT LOCATION
014720  042737  177760  001320          BIC     #^C17,ECBIT             ;SAVE THE BIT OFFSET COUNT
014726  042737  000017  001330          BIC     #17,ECWRD               ;CLEAR THE BIT OFFSET
014734  006237  001330                  ASR     ECWRD                   ;CHANGE TO BYTE COUNT
014740  006237  001330                  ASR     ECWRD                   ;CHANGE TO BYTE COUNT
014744  006237  001330                  ASR     ECWRD                   ;CHANGE TO BYTE COUNT
014750  104414  053005                  DISPLY  .LIN10A                 ;'ERROR BURST BEGINS AT '
014754  013746  001330                  MOV     ECWRD,-(SP)             ;PUT THE WORD COUNT ON THE STACK
014760  006216                          ASR     (SP)                    ;CONVERT TO WORD COUNT FOR MESSAGE
014762  004737  030302                  JSR     PC,$SB2D                ;CONVERT THE WORD COUNT
014766  004737  027702                  JSR     PC,$SUPRS               ;PRINT IT
014772  104414  053041                  DISPLY  .LIN10B                 ;' IN DATA FIELD OF ERROR SECTOR'
014776  063737  001322  001330          ADD     ECSEC,ECWRD             ;FIND THE BEGINNING OF THE ERROR BURST
015004  026037  000240  001330          CMP     $RPBA(R0),ECWRD  ;SEE IF BURST WAS IN DATA READ
015012  101002                          BHI     .+6                     ;BR IF IN DATA READ
015014  000137  015332                  JMP     ECC2                    ;NOT IN DATA READ - REPORT IT
015020  016037  000302  001324          MOV     $RPEC2(R0),ECMSK0  ;GET THE ERROR MASK
015026  005037  001326                  CLR     ECMSK1                  ;CLEAR THE UPPER MASK WORD
015032  005737  001320          3$:     TST     ECBIT                   ;BIT OFFSET EQUAL ZERO
015036  001407                          BEQ     4$                      ;BR IF IT IS
015040  005337  001320                  DEC     ECBIT                   ;DECREMENT THE BIT OFFSET COUNT
015044  006337  001324                  ASL     ECMSK0                  ;SHIFT THE ERROR MASK
015050  006137  001326                  ROL     ECMSK1                  ;SHIFT THE LOWER INTO THE UPPER
015054  000766                          BR      3$                      ;CONTINUE THE SHIFT
015056  017737  164246  001334  4$:     MOV     @ECWRD,ECBAD0           ;SAVE THE INCORRECT WORD
015064  005037  001336                  CLR     ECWRD1                  ;CLEAR SECOND INCORRECT WORD ADDRESS
015070  013746  001324                  MOV     ECMSK0,-(SP)            ;PUT LOWER MASK ON STACK
015074  047716  164230                  BIC     @ECWRD,(SP)             ;CLEAR ERRONEOUS ONE BITS FROM MASK
015100  043777  001324  164222          BIC     ECMSK0,@ECWRD           ;CLEAR ERRONEOUS ONE BITS FROM BAD WORD
015106  052677  164216                  BIS     (SP)+,@ECWRD            ;SET DROPPED BITS
015112  005737  001326                  TST     ECMSK1                  ;DOES BURST GO INTO NEXT WORD ?
015116  001431                          BEQ     ECC1                    ;BR IF BURST ONLY IN ONE WORD
015120  013737  001330  001336          MOV     ECWRD,ECWRD1            ;DUPLICATE ADDRESS
015126  062737  000002  001336          ADD     #2,ECWRD1               ;INCREMENT ERROR ADDRESS
015134  026037  000240  001336          CMP     $RPBA(R0),ECWRD1  ;IS NEXT WORD IN THE BUFFER
015142  101003                          BHI     5$                      ;BR IF IT IS
015144  005037  001336                  CLR     ECWRD1                  ;CLEAR 2ND WORD ADDRESS
015150  000414                          BR      ECC1                    ;PRINT WORD CORRECTED
015152  017737  164160  001342  5$:     MOV     @ECWRD1,ECBAD1          ;SAVE THE SECOND BAD WORD
015160  013746  001326                  MOV     ECMSK1,-(SP)            ;PUT THE UPPER MASK ON THE STACK
015164  047716  164146                  BIC     @ECWRD1,(SP)            ;CLEAR ERRONEOUS ONE BITS FROM UPPER MASK
015170  043777  001326  164140          BIC     ECMSK1,@ECWRD1          ;CLEAR ERRONEOUS ONE BITS FROM DATA WORD
015176  052677  164134                  BIS     (SP)+,@ECWRD1           ;SET DROPPED BITS
015202  104414  053205          ECC1:   DISPLY  .LIN10H                 ;HEADER
015206  013746  001330                  MOV     ECWRD,-(SP)             ;PUT ECWRD ON THE STACK
015212  004737  022544                  JSR     PC,LINOCT               ;TYPE ECWRD
015216  104414  053363                  DISPLY  .LINSP                  ;SPACES
015222  013746  001334                  MOV     ECBAD0,-(SP)            ;PUT ECBAD0 ON THE STACK
015226  004737  022544                  JSR     PC,LINOCT               ;TYPE ECBAD0
015232  104414  053363                  DISPLY  .LINSP                  ;SPACES
```

```
015236  017746  164066                    MOV     @ECWRD,-(SP)       ;PUT @ECWRD ON THE STACK
015242  004737  022544                    JSR     PC,LINOCT          ;TYPE @ECWRD
015246  104414  053363                    DISPLY  ,LINSP             ;SPACES
015252  005737  001336                    TST     ECWRD1             ;PRINT THE NEXT WORD ?
015256  001427                            BEQ     ECCX               ;BR IF NOT
015260  104414  001165                    DISPLY  ,$CRLF             ;CR-LF
015264  013746  001336                    MOV     ECWRD1,-(SP)       ;PUT ECWRD1 ON THE STACK
015270  004737  022544                    JSR     PC,LINOCT          ;TYPE ECWRD1
015274  104414  053363                    DISPLY  ,LINSP             ;SPACES
015300  013746  001342                    MOV     ECBAD1,-(SP)       ;PUT ECBAD1 ON THE STACK
015304  004737  022544                    JSR     PC,LINOCT          ;TYPE ECBAD1
015310  104414  053363                    DISPLY  ,LINSP             ;SPACES
015314  017746  164016                    MOV     @ECWRD1,-(SP)      ;PUT @ECWRD1 ON THE STACK
015320  004737  022544                    JSR     PC,LINOCT          ;TYPE @ECWRD1
015324  104414  053363                    DISPLY  ,LINSP             ;SPACES
015330  000402                            BR      ECCX               ;EXIT
015332  104414  053101            ECC2:   DISPLY  ,LIN10C            ;ERROR BURST WAS NOT TRANSFERED TO MEMORY
015336  104414  001165            ECCX:   DISPLY  ,$CRLF             ;CR-LF
015342  000207                            RTS     PC                 ;RETURN

                                  ;ROUTINE TO DISPLAY THE SECTOR WHICH GAVE THE HARD ERROR

015344  032777  000010  163566    PRTBAD: BIT     #SW3,@SWR          ;PRINT THE BAD SECTOR ?
015352  001460                            BEQ     6$                 ;BR IF NOT
015354  016001  000240                    MOV     $RPBA(R0),R1       ;PUT THE END ADDRESS INTO R1
015360  016046  000020                    MOV     $WRDL(R0),-(SP)    ;FIND THE BEGINNING OF THE SECTOR
015364  066016  000236                    ADD     $RPWC(R0),(SP)     ;SUBTRACT THE WORDS NOT TRANSFERED
015370  005046                            CLR     -(SP)              ;MAKE THE UPPER DIVIDEND 0
015372  016046  000022                    MOV     $SSEC(R0),-(SP)     ;DIVDE THE WORDS TRANSFERED BY THE SECTOR SIZE
015376  004737  027366                    JSR     PC,LINKDV          ;DIVIDE
015402  005716                            TST     (SP)               ;REMANDER = 0 ?
015404  001403                            BEQ     1$                 ;BR IF IT IS - COMPLETE SECTOR TRANSFERED
015406  006316                            ASL     (SP)               ;CONVERT THE RESIDUAL SECTOR SIZE INTO BYTE COUNT
015410  161601                            SUB     (SP),R1            ;SUBTRACT IT FROM THE END ADDRESS
015412  000410                            BR      2$                 ;FINISH THE SIZING
015414  162701  001000            1$:     SUB     #1000,R1           ;SUBTRACT FULL SECTOR SIZE FROM END ADDR
015420  126027  000024  000005            CMPB    $CODE(R0),#5       ;WAS OPERATION READ HEADER & DATA ?
015426  001002                            BNE     2$                 ;BR IF NOT
015430  162701  000010                    SUB     #10,R1             ;SUBTRACK HEADER SIZE FROM ADDR
015434  062706  000004            2$:     ADD     #4,SP              ;RESTORE THE STACK POINTER
015440  104414  053270                    DISPLY  ,LIN11H            ;PRINT THE HEADER
015444  012702  000007            3$:     MOV     #7,R2              ;R2 CONTAINS THE WORDS/LINE COUNT
015450  010146                            MOV     R1,-(SP)           ;PUT THE ADDRESS ON THE STACK
015452  004737  022544                    JSR     PC,LINOCT          ;TYPE THE ADDRESS
015456  020160  000240            4$:     CMP     R1,$RPBA(R0)       ;PRINTED ALL THE SECTOR ?
015462  001412                            BEQ     5$                 ;BR IF ALL PRINTED
015464  104414  053363                    DISPLY  ,LINSP             ;SPACES
015470  012146                            MOV     (R1)+,-(SP)        ;PUT THE DATA ON THE STACK
015472  004737  022544                    JSR     PC,LINOCT          ;TYPE THE DATA
015476  005302                            DEC     R2                 ;DECREMENT THE HORIZONTAL COUNT
015500  001366                            BNE     4$                 ;BR IF NOT AT THE END OF THE LINE
015502  104414  001165                    DISPLY  ,$CRLF             ;CR-LF
015506  000756                            BR      3$                 ;RESTORE THE WORDS/LINE COUNT
015510  104414  001165            5$:     DISPLY  ,$CRLF             ;PRINT WHAT REMAINS IN THE BUFFER
015514  000207                    6$:     RTS     PC                 ;RETURN

                                  ;ROUTINE TO DO AN RTC - DRIVE SELECTED IN R0
```

```
                                        ;CALL:
                                        ;       MOV     #DPB,R0         ;DPB ADDRCSS
                                        ;       JSR     PC,RTNCTR
                                        ;       RETURN

015516  111037  046150                  RTNCTR: MOVB    (R0),GENDPB     ;MOVE THE DRIVE # TO THE GENERAL DPB
015522  112737  000117  046152                  MOVB    #RTC,GENDPB+$COMND  ;COMMAND CODE
015530  004037  035340                  1$:     JSR     R0,RP04         ;DRIVER ENTRANCE
015534  046150                                  GENDPB                  ;DPB ADDRESS FOR ORDER
015536  000774                                  BR      1$              ;DRIVER DIDN'T ACCEPT ORDER
015540  000207                                  RTS     PC              ;RETURN

                                        ;ROUTINE TO DO A RECALIBRATE - DRIVE SELECTED IN R0
                                        ;CALL:
                                        ;       MOV     #DPB,R0         ;DPB ADDRESS
                                        ;       JSR     PC,RECALT
                                        ;       RETURN
                                        ;
                                        ;OR
                                        ;
                                        ;       MOV     #DPB,R0         ;DPB ADDRESS
                                        ;       MOVB    #DRIVE,GENDPB   ;DRIVE ADDRESS
                                        ;       JSR     PC,RECALT0
                                        ;       RETURN

015542  111037  046150                  RECALT: MOVB    (R0),GENDPB     ;MOVE THE DRIVE # TO THE GENERAL DPB
015546  112737  000107  046152          RECAL0: MOVB    #RECAL,GENDPB+$COMND  ;RELCALIBRATE COMMAND
015554  004037  035340                  1$:     JSR     R0,RP04         ;DRIVER ENTRANCE
015560  046150                                  GENDPB                  ;DPB ADDRESS FOR ORDER
015562  000774                                  BR      1$              ;DRIVER DIDN'T ACCEPT THE ORDER
015564  005737  046166                  2$:     TST     GENDPB+$STATUS          ;SEE IF FINISHED
015570  001775                                  BEQ     2$              ;BR IF NOT FINISHED
015572  000207                                  RTS     PC              ;RETURN

                                        ;OFFSET THE DRIVE IN R0 (OFFSET CODE PRELOADED INTO 'RPOF')
                                        ;CALL:
                                        ;       MOVB    #OFFSET,GENDPB+$FMT  ;OFFSET CODE
                                        ;       MOV     #DPB,R0         ;DPB ADDRESS
                                        ;       JSR     PC,OFFST
                                        ;       RETURN

015574  111037  046150                  OFFST:  MOVB    (R0),GENDPB     ;DRIVE # TO GENERAL DPB
015600  112737  000115  046152                  MOVB    #OFFSET,GENDPB+$COMND ;COMMAND
015606  004037  035340                  1$:     JSR     R0,RP04         ;DRIVER ENTRANCE
015612  046150                                  GENDPB                  ;DPB ADDRESS FOR ORDER
015614  000774                                  BR      1$              ;DRIVER DIDN'T ACCEPT ORDER
015616  000207                                  RTS     PC

                                        ;UTILITY READ HEADER ROUTINE
                                        ;CALL:
                                        ;       MOV     #DPB,R0         ;DPB ADDRESS
                                        ;       MOV     #SECTOR,-(SP)   ;SECTOR ADDRESS
                                        ;       MOV     #TRACK,-(SP)    ;TRACK ADDRESS
                                        ;       JSR     PC,READDR
                                        ;       RETURN

015620  116637  000002  046161          READHD: MOVB    2(SP),GENDPB+$TRK ;TRACK ADDRESS
```

```
015626  116637  000004  046160           MOVB    4(SP),GENDPB+$SEC ;SECTOR ADDRESS
015634  111037  046150                   MOVB    (R0),GENDPB       ;DRIVE NUMBER
015640  016037  000272  046162           MOV     $RPCC(R0),GENDPB+$CYL ;CYLINDER ADDRESS
015646  112737  000173  046152           MOVB    #RDHD,GENDPB+$COMND ;COMMAND
015654  004037  035340          1$:      JSR     R0,RP04           ;DRIVER ENTRANCE
015660  046150                           GENDPB                    ;DPB ADDRESS FOR ORDER
015662  000774                           BR      1$                ;DRIVER DIDN'T ACCEPT COMMAND
015664  005737  046166          2$:      TST     GENDPB+$STATUS    ;FINISHED?
015670  001775                           BEQ     2$                ;BR IF NOT
015672  012666  000002                   MOV     (SP)+,2(SP)       ;ADJUST STACK FOR RETURN
015676  005726                           TST     (SP)+
015700  000207                           RTS     PC                ;RETURN

                                ;RETRY THE PRESENT OPERATION
                                ;CALL:
                                ;        MOV     #COUNT,RETRY      ;RETRY COUNT
                                ;        JSR     PC,$RETRY
                                ;        RETURN1                   ;RETRY UNSUCESSFUL
                                ;        RETURN2                   ;SUCESSFUL RETRY
                                ;                                  ;NOTE: IF A DIFFERENT ERROR OCCURS DURING
                                ;                                  ;RETRY, THE ROUTINE EXITS TO 'ERPRC1'

015702  004737  016672          $RETRY:  JSR     PC,GODRIV         ;RE-START ORDER
015706  005760  000016          1$:      TST     $STATUS(R0)       ;ORDER FINISHED?
015712  001775                           BEQ     1$                ;BR IF NOT
015714  100405                           BMI     2$                ;BR IF ERROR
015716  105237  001253                   INCB    RETRY+1           ;INCREMENT RETRY COUNT
015722  062716  000002                   ADD     #2,(SP)           ;INCREMENT RETURN
015726  000425                           BR      5$                ;GO TO EXIT
015730  032760  000200  000016  2$:      BIT     #BIT7,$STATUS(R0)  ;DID ORDER TERMINATE NORMALLY ?
015736  001430                           BEQ     7$                ;BR IF NOT
015740  005737  001250                   TST     MASK              ;IS ERROR MASK 0 ?
015744  001004                           BNE     3$                ;BR IF NOT
015746  005760  000250                   TST     $RPER1(R0)        ;MAKE SURE THAT THE DRIVE ERROR REG IS CLEAR
015752  001014                           BNE     6$                ;BR IF NOT
015754  000404                           BR      4$                ;CONTINUE RETRY
015756  033760  001250  000250  3$:      BIT     MASK,$RPER1(R0)   ;SAME ERROR?
015764  001407                           BEQ     6$                ;BR IF NOT
015766  105237  001253          4$:      INCB    RETRY+1           ;INCREMENT RETRY COUNT
015772  123737  001252  001253           CMPB    RETRY,RETRY+1     ;DONE ?
016000  001340                           BNE     $RETRY            ;BR IF NOT DONE
016002  000207                  5$:      RTS     PC                ;RETURN
016004  004737  022532          6$:      JSR     PC,LINE8          ;REPORT DIFFERENT ERROR
016010  004737  022264                   JSR     PC,LINE7          ;PRINT LINE 7
016014  005726                           TST     (SP)+             ;ADJUST STACK POINTER FOR DIRECT RETURN
016016  000207                           RTS     PC                ;RETURN
016020  104414  052550          7$:      DISPLY  ,LIN8M            ;'DIFFERENT ERROR DURING RETRY'
016024  000137  007146                   JMP     ERPRC1            ;REPORT THE ERROR

                                ;ROUTINE TO UPDATE THE PERFORMANCE SUMMARY STATISTICS
                                ;CALL:
                                ;        MOV     #DPB,R0           ;DPB ADDRESS
                                ;        JSR     PC,STATIS
                                ;        RETURN

016030  032760  000300  000016  STATIS:  BIT     #BIT07!BIT06,$STATUS(R0) ;CHECK FOR DATA TERMINATION
016036  001454                           BEQ     3$                ;BR IF NOT DATA TERMINATION
```

```
016040 016037  000240  016172          MOV     $RPBA(R0),FACTOR  ;STORE THE FINAL BUFFER ADDRESS
016046 166037  000006  016172          SUB     $BUF(R0),FACTOR ;SUBTRACT THE INITIAL ADDRESS
016054 001434                          BEQ     2$              ;BR IF NO DATA TRANSFER
016056 006237  016172                  ASR     FACTOR          ;CONVERT TO A WORD COUNT
016062 063760  016172  000046          ADD     FACTOR,$TRANS(R0)  ;UPDATE WORD COUNT
016070 005560  000050                  ADC     $TRANS+2(R0)    ;ADD ANY CARRY
016074 132760  000002  000024          BITB    #BIT01,$CODE(R0)  ;SEE IF ORDER READ OR WRITE
016102 001021                          BNE     2$              ;BRANCH IF ORDER WRITE
016104 005737  001424                  TST     AUTOCK          ;AUTO WRITE CHECKS BEING PERFORMED
016110 001411                          BEQ     1$              ;BR IF NOT
016112 126027  000024  000001          CMPB    $CODE(R0),#1    ;PRESENT OPERATION AN AUTOMATIC WRITE CHECK ?
016120 101005                          BHI     1$              ;BR IF NOT
016122 066060  000020  000046          ADD     $WRDL(R0),$TRANS(R0)  ;ADD WORDS WRITTEN
016130 005560  000050                  ADC     $TRANS+2(R0)    ;ADD A CARRY
016134 063760  016172  000052  1$:     ADD     FACTOR,$READ(R0)  ;UPDATE THE READ WORD COUNT
016142 005560  000054                  ADC     $READ+2(R0)     ;ADD ANY CARRY
016146 026060  000012  000270  2$:     CMP     $CYL(R0),$RPCA(R0)  ;DID MID-TRANSFER SEEK OCCUR
016154 001405                          BEQ     3$              ;BR IF NOT
016156 062760  000001  000042          ADD     #1,$POSIT(R0)   ;INCREMENT SEEK COUNT
016164 005560  000044                  ADC     $POSIT+2(R0)    ;ADD CARRY TO UPPER WORD
016170 000207                  3$:     RTS     PC

016172 000000                  FACTOR: .WORD   0               ;USED FOR WORDS TRANSFERED

                               ;ROUTINE TO GET A BUFFER
                               ;CALL:
                               ;       MOV     #DPB,R0         ;DPB ADDRESS
                               ;       CLR     -(SP)           ;CLEAR THE STACK
                               ;       JSR     PC,GETBUF
                               ;       RETURN                  ;BUFFER ADDRESS WILL BE ON THE STACK
                               ;                               ;STACK WILL BE ZERO IF NO BUFFER AVAILABLE

016174 010146                  GETBUF: MOV     R1,-(SP)        ;SAVE R1
016176 010246                          MOV     R2,-(SP)        ;SAVE R2
016200 010346                          MOV     R3,-(SP)        ;SAVE R3
016202 013702  001616                  MOV     BUFTBL,R2       ;NUMBER OF SEPARATE BUFFERS
016206 001444                          BEQ     6$              ;BR IF NONE AVAILABLE
016210 012701  001620                  MOV     #BUFTBL+2,R1    ;FIRST ADDRESS OF ALLOCATION TABLE
016214 026061  000020  000002  1$:     CMP     $WRDL(R0),2(R1) ;SEE IF THERE IS A BLOCK LARGE ENOUGH
016222 101405                          BLOS    3$              ;BRANCH IF IT IS
016224 005302                          DEC     R2              ;DECREMENT TABLE COUNT
016226 001434                          BEQ     6$              ;BR IF THROUGH TABLE
016230 062701  000004                  ADD     #4,R1           ;INCREMENT TABLE POINTER
016234 000767                          BR      1$              ;CONTINUE LOOKING
016236 011166  000010          3$:     MOV     (R1),10(SP)     ;BUFFER ADDRESS TO STACK
016242 166061  000020  000002          SUB     $WRDL(R0),2(R1) ;ADJUST BUFFER SIZE
016250 001407                          BEQ     4$              ;BR IF DIFFERENCE IS ZERO
016252 006360  000020                  ASL     $WRDL(R0)       ;CONVERT # WORDS TO BYTES
016256 066011  000020                  ADD     $WRDL(R0),(R1)  ;MAKE NEW STARTING ADDRESS
016262 006260  000020                  ASR     $WRDL(R0)       ;RETURN # BYTES TO WORDS
016266 000414                          BR      6$              ;RETURN
016270 005337  001616          4$:     DEC     BUFTBL          ;DECREMENT ENTRIES COUNT
016274 001411                          BEQ     6$              ;BR IF ALLOCATION TABLE EMPTY
016276 005302                          DEC     R2              ;DECREMENT TABLE COUNT
016300 001407                          BEQ     6$              ;BR IF ITEM WERE LAST ENTRY
016302 010103                          MOV     R1,R3           ;MOVE TABLE POINTER
016304 062703  000004                  ADD     #4,R3           ;POINT TO NEXT ENTRY
```

```
016310  012321                          5$:     MOV     (R3)+,(R1)+        ;MOVE ITEMS
016312  012321                                  MOV     (R3)+,(R1)+
016314  005302                                  DEC     R2                 ;DECREMENT TABLE COUNT
016316  001374                                  BNE     5$                 ;CONTINUE IF NOT AT END OF TABLE
016320  012603                          6$:     MOV     (SP)+,R3           ;RESTORE R3
016322  012602                                  MOV     (SP)+,R2           ;RESTORE R2
016324  012601                                  MOV     (SP)+,R1           ;RESTORE R1
016326  000207                                  RTS     PC                 ;RETURN


                                        ;ROUTINE TO PUT BUFFER BACK IN TABLE
                                        ;CALL:
                                        ;       MOV     #DPB,R0            ;DPB ADDRESS
                                        ;       JSR     PC,RELBUF
                                        ;       RETURN

016330  010146                          RELBUF: MOV     R1,-(SP)           ;SAVE R1
016332  012701  001620                          MOV     #BUFTBL+2,R1       ;BEGINNING OF TABLE
016336  013702  001616                          MOV     BUFTBL,R2          ;ENTRY COUNT
016342  001424                                  BEQ     2$                 ;BR IF EMPTY TABLE
016344  016003  000020                          MOV     $WRDL(R0),R3       ;TRIAL ADDRESS
016350  006303                                  ASL     R3                 ;CHANGE TO BYTE COUNT
016352  066003  000006                          ADD     $BUF(R0),R3        ;ADDRESS OF HIGHER ADJACENT BLOCK
016356  021103                          1$:     CMP     (R1),R3            ;UPPER ADJACENT BLOCK
016360  001424                                  BEQ     4$                 ;BR IF YES
016362  062701  000004                          ADD     #4,R1              ;INCREMENT POINTER
016366  005302                                  DEC     R2                 ;DECREMENT ENTRY COUNT
016370  001372                                  BNE     1$                 ;CONTINUE SEARCHING
016372  016011  000006                          MOV     $BUF(R0),(R1)      ;PUT THE BUFFER BLOCK INTO THE TABLE
016376  016061  000020  000002                  MOV     $WRDL(R0),2(R1)     ;BLOCK SIZE
016404  005237  001616                          INC     BUFTBL             ;INCREMENT ENTRY COUNT
016410  005202                                  INC     R2                 ;INCREMENT R2 FOR USE LATER
016412  000414                                  BR      5$                 ;SEE IF A LOWER ADJACENT BLOCK IS IN THE TABLE
016414  016021  000006                  2$:     MOV     $BUF(R0),(R1)+     ;BLOCK ADDRESS TO TABLE
016420  016021  000020                          MOV     $WRDL(R0),(R1)+    ;SIZE TO TABLE
016424  005237  001616                          INC     BUFTBL             ;INCREMENT ENTRY COUNT
016430  000443                                  BR      10$                ;EXIT
016432  016011  000006                  4$:     MOV     $BUF(R0),(R1)      ;RELEASED BUFFER IS LOWER ADJACENT
016436  066061  000020  000002                  ADD     $WRDL(R0),2(R1)    ;INCREMENTED SIZE
016444  010246                          5$:     MOV     R2,-(SP)           ;SAVE R2
016446  013702  001616                          MOV     BUFTBL,R2          ;ENTRY COUNT
016452  012705  001620                          MOV     #BUFTBL+2,R5       ;BEGINNING OF TABLE
016456  016504  000002                  6$:     MOV     2(R5),R4           ;BLOCK SIZE (IN WORDS)
016462  006304                                  ASL     R4                 ;CHANGE TO BYTE COUNT
016464  061504                                  ADD     (R5),R4            ;ADD BLOCK BEGINNING ADDRESS
016466  020411                                  CMP     R4,(R1)            ;R1 STILL POINTS TO INSERTED ENTRY
016470  001406                                  BEQ     8$                 ;LOWER ADJACENT IN TABLE
016472  062705  000004                          ADD     #4,R5              ;INCREMENT POINTER
016476  005302                                  DEC     R2                 ;DECREMENT ENTRY COUNT
016500  001366                                  BNE     6$                 ;CONTINUE LOOKING
016502  005726                                  TST     (SP)+              ;RESTORE STACK POINTER
016504  000415                                  BR      10$                ;END
016506  012602                          8$:     MOV     (SP)+,R2           ;RESTORE R2
016510  066165  000002  000002                  ADD     2(R1),2(R5)        ;INCREMENT LOWER BLOCK LENGTH
016516  005337  001616                          DEC     BUFTBL             ;DECREMENT ENTRY COUNT
016522  010105                                  MOV     R1,R5              ;GET READY TO COMPRESS
016524  062705  000004                          ADD     #4,R5              ;INCREMENT TO NEXT ENTRY
```

```
016530  012521                      9$:     MOV     (R5)+,(R1)+             ;COMPRESS TABLE
016532  012521                              MOV     (R5)+,(R1)+             ;MOVE SIZE FIELD DOWN
016534  005302                              DEC     R2                      ;DECREMENT ENTRY COUNT
016536  001374                              BNE     9$                      ;BR IF NOT FINISHED
016540  012601                      10$:    MOV     (SP)+,R1                ;RESTORE R1
016542  000207                              RTS     PC                      ;RETURN


                                    ;FILL THE ASSIGNED BUFFER (IF WRITE OR WRITE CHECK ORDER)
                                    ;CALL:
                                    ;       MOV     #DPB,R0                 ;DPB ADDRESS
                                    ;       MOV     #BUFADR,$BUF(R0)  ;LOAD BUFFER ADDRESS INTO THE DPB
                                    ;       MOVB    #PATTERN,$PATTC(R0)  ;PATTERN CODE
                                    ;       JSR     PC,FILBUF
                                    ;       RETURN

016544  104412                      FILBUF: SAVREG                          ;SAVE THE REGISTERS
016546  132760  000004  000024              BITB    #BIT02,$CODE(R0)  ;SEE IF READ ORDER
016554  001044                              BNE     4$                      ;BR IF READ
016556  016001  000006              1$:     MOV     $BUF(R0),R1             ;BUFFER ADDRESS
016562  016002  000020                      MOV     $WRDL(R0),R2            ;POSITIVE WORD COUNT
016566  132760  000001  000024              BITB    #BIT00,$CODE(R0)  ;SEE IF WRITE HEADER TYPE ORDER
016574  001413                              BEQ     2$                      ;BR IF NOT
016576  016011  000012                      MOV     $CYL(R0),(R1)           ;CYLINDER ADDRESS
016602  052721  010000                      BIS     #BIT12,(R1)+            ;SET FMT22 BIT
016606  016021  000010                      MOV     $SEC(R0),(R1)+          ;MOVE SECTOR & TRACK
016612  005021                              CLR     (R1)+                   ;CLEAR FIRST KEY WORD
016614  005021                              CLR     (R1)+                   ;CLEAR THE SECOND
016616  162702  000004                      SUB     #4,R2                   ;ADJUST THE WORD COUNT
016622  003421                              BLE     4$                      ;BR IF END OF PATTERN
016624  005004                      2$:     CLR     R4                      ;CLEAR R4
016626  116004  000030                      MOVB    $PATTC(R0),R4           ;RELATIVE PATTERN ADDRESS
016632  016405  002762                      MOV     STNDAT(R4),R5           ;PATTERN ADDRESS
016636  012703  000020                      MOV     #20,R3                  ;PATTERN COUNT
016642  012521                      3$:     MOV     (R5)+,(R1)+             ;MOVE THE PATTERN INTO THE BUFFER
016644  005302                              DEC     R2                      ;DECREMENT THE WORD COUNT
016646  001407                              BEQ     4$                      ;BR IF DONE (WORD COUNT = 0)
016650  005303                              DEC     R3                      ;DECREMENT THE PATTERN COUNT
016652  001373                              BNE     3$                      ;BR IF MORE PATTERN
016654  012703  000020                      MOV     #20,R3                  ;RESTORE PATTERN COUNT
016660  016405  002762                      MOV     STNDAT(R4),R5           ;RESTORE THE ADDRESS
016664  000766                              BR      3$                      ;CONTINUE DISTRIBUTING THE PATTERN
016666  104413                      4$:     RESREG                          ;RESTORE THE REGISTERS
016670  000207                              RTS     PC                      ;RETURN

                                    ;START THE ORDER FOR THE DPB IN R0
                                    ;CALL:
                                    ;       MOV     #DPB,R0                 ;DPB ADDRESS
                                    ;       JSR     PC,GODRIV
                                    ;       RETURN

016672  010046                      GODRIV: MOV     R0,-(SP)                ;SAVE R0
016674  010037  016704                      MOV     R0,2$                   ;CURRENT DPB ADDRESS
016700  004037  035340              1$:     JSR     R0,RP04                 ;CALL THE DRIVE HANDLER
016704  000000                      2$:     .WORD   0                       ;DRIVE BLOCK ADDRESS GOES HERE
016706  000000                              HALT                            ;DRIVER REJECTED REQUEST
016710  012600                              MOV     (SP)+,R0                ;RESTORE R0
```

```
016712  062760  000001  000036           ADD     #1,$OPERC(R0)    ;INCREMENT THE OPERATION COUNT
016720  005560  000040                   ADC     $OPERC+2(R0)
016724  026060  000034  000012           CMP     $PREVA+2(R0),$CYL(R0)  ;DID ORDER REQUIRE A CYLINDER CHANGE
016732  001405                           BEQ     3$               ;BR IF NOT
016734  062760  000001  000042           ADD     #1,$POSIT(R0)    ;INCREMENT SEEK COUNT
016742  005560  000044                   ADC     $POSIT+2(R0)     ;ADD ANY CARRY
016746  000207                   3$:     RTS     PC

                                 ;GENERATE PARAMETERS FOR THE OPERATION
                                 ;CALL:
                                 ;       MOV     #DPB,R0          ;DPB ADDRESS
                                 ;       JSR     PC,SELPAR
                                 ;       RETURN

016750  004737  033534           SELPAR: JSR     PC,$RAND         ;CYCLE THE RANDOM NUMBER GENERATOR
016754  032777  000001  162156           BIT     #SW0,@SWR        ;SEE IF SW0 SET
016762  001012                           BNE     2$               ;BR IF SET - READ ONLY
016764  012705  000010           1$:     MOV     #10,R5           ;READ/WRITE SELECTION DIVISOR
016770  004737  027340                   JSR     PC,GETREM        ;GET SELECTION VALUE
016774  020537  001422                   CMP     R5,RATIO         ;DETERMINE IF READ OR WRITE
017000  103003                           BHIS    2$               ;BR IF READ
017002  004737  017472                   JSR     PC,RANWRT        ;SELECT A WRITE ORDER
017006  000406                           BR      3$               ;CONTINUE WITH THE SELECTION
017010  013705  033634           2$:     MOV     $LONUM,R5        ;SELECT READ OPERATION CODE
017014  042705  177776                   BIC     #^C1,R5          ;MASK OUT ALL BUT BIT 0
017020  062705  000004                   ADD     #4,R5            ;TABLE OFFSET FOR READ CODE
017024  110560  000074           3$:     MOVB    R5,$NCODE(R0)    ;ORDER SELECTION CODE TO CONTROL BLOCK

                                 ;GENERATE A RANDOM SECTOR ADDRESS BETWEEN VALUES 'MINSEC' & 'MAXSEC'

017030  016005  000116           RANSEC: MOV     MAXSEC(R0),R5    ;GET MAXIMUM SECTOR ADDRESS
017034  026005  000120                   CMP     MINSEC(R0),R5    ;'MINSEC' AND 'MAXSEC' THE SAME ?
017040  001417                           BEQ     2$               ;BR IF THEY ARE
017042  166005  000120                   SUB     MINSEC(R0),R5    ;SUBTRACT MINIMUM SECTOR ADDRESS
017046  100002                           BPL     1$               ;BR IF MAX LARGER THAN MIN
017050  062705  000026                   ADD     #22.,R5          ;CORRECT THE NUMBER
017054  005205                   1$:     INC     R5               ;INCREMENT DIFFERENCE TO USE AS DIVISOR
017056  004737  027340                   JSR     PC,GETREM        ;GET THE RANDOM AUGMENT
017062  066005  000120                   ADD     MINSEC(R0),R5    ;NEW ADDRESS
017066  020527  000025                   CMP     R5,#21.          ;IS VALUE TOO LARGE ?
017072  101402                           BLOS    2$               ;BR IF NOT
017074  162705  000026                   SUB     #22.,R5          ;CORRECT VALUE
017100  110560  000076           2$:     MOVB    R5,$NSEC(R0)     ;STORE SECTOR ADDRESS IN DPB

                                 ;GENERATE A RANDOM TRACK ADDRESS BETWEEN VALUES 'MINTRK' & 'MAXTRK'

017104  016005  000112           RANTRK: MOV     MAXTRK(R0),R5    ;GET MAXIMUM TRACK ADDRESS
017110  026005  000114                   CMP     MINTRK(R0),R5    ;'MINTRK' AND 'MAXTRK' THE SAME ?
017114  001417                           BEQ     2$               ;BR IF THEY ARE
017116  166005  000114                   SUB     MINTRK(R0),R5    ;SUBTRACT MINIMUM TRACK ADDRESS
017122  100002                           BPL     1$               ;BR IF MAX LARGER THAN MIN
017124  062705  000023                   ADD     #19.,R5          ;CORRECT THE NUMBER
017130  005205                   1$:     INC     R5               ;INCREMENT DIFFERENCE TO USE AS DIVISOR
017132  004737  027340                   JSR     PC,GETREM        ;GET THE RANDOM AUGMENT
017136  066005  000114                   ADD     MINTRK(R0),R5    ;NEW TRACK ADDRESS
017142  020527  000022                   CMP     R5,#18.          ;IS VALUE TOO LARGE ?
017146  101402                           BLOS    2$               ;BR IF NOT
```

```
017150  162705  000023                      SUB     #19.,R5             ;CORRECT VALUE
017154  110560  000077          2$:         MOVB    R5,$NTRK(R0)        ;STORE TRACK ADDRESS IN DPB

                                ;GENERATE A RANDOM CYLINDER ADDRESS BETWEEN VALUES 'MINCYL' & 'MAXCYL'

017160  012737  000633  001350              MOV     #411.,CYLIMT        ;ASSUME AN RP04/5
017166  032760  000002  000262              BIT     #BIT01,$RPDT(R0)      ;SEE IF RP06
017174  001403                              BEQ     RANCYL              ;BR IF NOT
017176  012737  001457  001350              MOV     #815.,CYLIMT        ;CHANGE CYLINDER LIMIT
017204  016005  000106          RANCYL:     MOV     MAXCYL(R0),R5       ;GET MAXIMUM CYLINDER ADDRESS
017210  026005  000110                      CMP     MINCYL(R0),R5       ;'MINCYL' AND 'MAXCYL' THE SAME ?
017214  001417                              BEQ     2$                  ;BR IF THEY ARE
017216  166005  000110                      SUB     MINCYL(R0),R5       ;SUBTRACT MINIMUM CYLINDER ADDRESS
017222  100002                              BPL     1$                  ;BR IF MAX LARGER THAN MIN
017224  063705  001350                      ADD     CYLIMT,R5           ;CORRECT THE NUMBER
017230  005205                  1$:         INC     R5                  ;INCREMENT DIFFERENCE TO USE AS DIVISOR
017232  004737  027340                      JSR     PC,GETREM           ;GET THE RANDOM AUGMENT
017236  066005  000110                      ADD     MINCYL(R0),R5       ;NEW CYLINDER ADDRESS
017242  023705  001350                      CMP     CYLIMT,R5           ;IS VALUE TOO LARGE ?
017246  101002                              BHI     2$                  ;BR IF NOT
017250  163705  001350                      SUB     CYLIMT,R5           ;CORRECT VALUE
017254  010560  000100          2$:         MOV     R5,$NCYL(R0)        ;STORE CYLINDER ADDRESS IN DPB
017260  122760  000003  000074              CMPB    #3,$NCODE(R0)       ;WRITE HEADER & DATA ?
017266  001013                              BNE     RANSIZ              ;BR IF NOT
017270  012760  000404  000102              MOV     #260.,$NWRDL(R0)      ;CHANGE WORD LENGTH TO 260 FOR WRTHD ORDER
017276  023727  001404  000404              CMP     MAXDL,#260.         ;CAN A FULL SECTOR BE WRITTEN ?
017304  103062                              BHIS    RANPAT              ;BR IF IT CAN
017306  013760  001404  000102              MOV     MAXDL,$NWRDL(R0)      ;CHANGE TRANSFER SIZE
017314  000456                              BR      RANPAT              ;CONTINUE WITH THE SELECTION

                                ;GENERATE A RANDOM BUFFER LENGTH BETWEEN 4 & THE VALUE IN 'MAXDL'

017316  013705  001404          RANSIZ:     MOV     MAXDL,R5            ;GET BUFFER SIZE
017322  005737  001420                      TST     WCSEL               ;SELECT A RANDOM WORD COUNT ?
017326  001010                              BNE     1$                  ;BR IF NOT
017330  005205                              INC     R5                  ;INCREMENT THE MAXIMUM SIZE
017332  004737  027340                      JSR     PC,GETREM           ;DIVIDE BY MAX VALUE
017336  005705                              TST     R5                  ;IS THE REMAINDER 0 ?
017340  001003                              BNE     1$                  ;NOT 0, CONTINUE
017342  004737  033534                      JSR     PC,$RAND            ;CYCLE THE RANDOM NUMBER GENERATOR
017346  000763                              BR      RANSIZ              ;TRY AGAIN
017350  010560  000102          1$:         MOV     R5,$NWRDL(R0)       ;WORD LENGTH TO CONTROL BLOCK
017354  010546                              MOV     R5,-(SP)            ;NEW WORD LENGTH ON STACK FOR CHECK
017356  005046                              CLR     -(SP)               ;MAKE UPPER DIVIDEND ZERO
017360  012746  000400                      MOV     #256.,-(SP)         ;SECTOR SIZE IS THE DIVISOR
017364  132760  000001  000074              BITB    #1,$NCODE(R0)       ;SEE IF NEXT ORDER IS A HEADER ORDER
017372  001402                              BEQ     2$                  ;BR IF NOT
017374  062716  000004                      ADD     #4,(SP)             ;ADD HEADER SIZE TO SECTOR SIZE
017400  004737  027366          2$:         JSR     PC,LINKDV           ;DIVIDE BUFFER SIZE BY SECTOR SIZE
017404  012616                              MOV     (SP)+,(SP)          ;MOV REMAINDER UP THE STACK
017406  021627  000004                      CMP     (SP),#4 ;SEE IF REMAINDER LESS THAN 4
017412  103012                              BHIS    4$                  ;BR IF NOT
017414  005737  001420                      TST     WCSEL               ;SELECTING RANDOM TRANSFER SIZES ?
017420  001403                              BEQ     3$                  ;BR IF YES
017422  161660  000102                      SUB     (SP),$NWRDL(R0)       ;ADJUST WORD LENGTH DOWNWARD
017426  000404                              BR      4$                  ;CONTINUE
017430  005726                  3$:         TST     (SP)+               ;CORRECT THE STACK POINTER
```

```
017432  004737  033534                  JSR     PC,$RAND          ;CYCLE THE RANDOM NUMBER GENERATOR
017436  000727                          BR      RANSIZ            ;TRY AGAIN
017440  005726                  4$:     TST     (SP)+             ;CORRECT THE STACK POINTER
017442  122760  000002  000074          CMPB    #2,$NCODE(R0)     ;SEE IF WRITE DATA
017450  001004                          BNE     RANXIT            ;BR IF NOT WRITE DATA

                                ;GET A RANDOM PATTERN NUMBER

017452  004737  017576          RANPAT: JSR     PC,GETPAT         ;GET PATTERN CODE
017456  110560  000075                  MOVB    R5,$NPATC(R0)     ;MOVE PATTERN CODE TO CONTROL BLOCK
017462  012760  177777  000104  RANXIT: MOV     #-1,$NEXT(R0)    ;SET PARAMETERS SELECTED INDICATOR
017470  000207                          RTS     PC                ;RETURN

                                ;ROUTINE TO SELECT A WRITE (OR WRITE CHECK) OPERATION

017472  012705  000004          RANWRT: MOV     #4,R5             ;WRITE OPERATION SELECTION DIVISOR
017476  004737  027340                  JSR     PC,GETREM         ;GET SELECTION CODE
017502  005737  001424                  TST     AUTOCK            ;ARE WRITE CHECK ORDERS TO BE SELECTED
                                                                  ;RANDOMLY ?
017506  001403                          BEQ     1$                ;BR IF THEY ARE
017510  152705  000002                  BISB    #2,R5             ;SET CODE TO EXCLUDE WRITE CHECK ORDERS
017514  000420                          BR      3$                ;COMPLETE SELECTION
017516  020527  000001          1$:     CMP     R5,#1             ;WRITE CHECK SELECTED ?
017522  101015                          BHI     3$                ;BR IF NOT
017524  132760  000002  000024          BITB    #2,$CODE(R0)      ;PREVIOUS WRITE OPERATION ?
017532  001407                          BEQ     2$                ;BR IF PREVIOUS WAS READ OR WRITE CHECK
017534  116060  000024  000074          MOVB    $CODE(R0),$NCODE(R0)  ;MOVE CODE TO 'NEXT CODE'
017542  142760  000002  000074          BICB    #2,$NCODE(R0)     ;CHANGE WRITE TO WRITE CHECK
017550  000411                          BR      5$                ;EXIT
017552  052705  000002          2$:     BIS     #2,R5             ;CHANGE WRITE CHECK TO WRITE
017556  005737  001416          3$:     TST     FORMAT            ;WRITE HEADER ORDERS ALLOWED ?
017562  001002                          BNE     4$                ;BR IF THEY ARE
017564  042705  000001                  BIC     #1,R5             ;ALTER POSSIBLE WRITE HEADER
017570  110560  000074          4$:     MOVB    R5,$NCODE(R0)     ;SETUP 'NEXT' CODE
017574  000207                  5$:     RTS     PC                ;RETURN

                                ;ROUTINE TO SELECT A PATTERN

017576  012705  000020          GETPAT: MOV     #20,R5            ;SELECT PATTERN
017602  004737  027340                  JSR     PC,GETREM         ;GET CODE
017606  005705                          TST     R5                ;WAS PATTERN ZERO SELECTED ?
017610  001003                          BNE     1$                ;BR IF NOT ZERO
017612  004737  033534                  JSR     PC,$RAND          ;CYCLE THE RANDOM NUMBER GENERATOR
017616  000767                          BR      GETPAT            ;TRY AGAIN
017620  006305                  1$:     ASL     R5                ;MAKE CODE INTO TABLE INDEX
017622  000207                          RTS     PC

                                ;ROUTINE TO GET THE PREVIOUSLY SELECTED PARAMETER VALUES
                                ;CALL:
                                ;       MOV     #DPB,R0           ;DPB ADDRESS
                                ;       JSR     PC,SELPAR         ;SELECT THE PARAMETERS
                                ;       JSR     PC,GETPAR
                                ;       RETURN

017624  010546                  GETPAR: MOV     R5,-(SP)          ;SAVE R5
017626  116060  000234  000027          MOVB    $RPCS1(R0),$PREVO(R0)  ;SAVE CURRENT PARAMETERS
017634  032760  000006  000074          BIT     #6,$NCODE(R0)     ;SEE IF NEXT OPERATION IS READ OR WRITE
```

```
017642  001007                          BNE     1$                  ;BR IF EITHER
017644  016060  000012  000034          MOV     $CYL(R0),$PREVA+2(R0)   ;SAVE STARTING CYLINDER
017652  016060  000010  000032          MOV     $SEC(R0),$PREVA(R0)  ;SAVE STARTING SECTOR AND TRACK
017660  000411                          BR      2$
017662  004737  022616          1$:     JSR     PC,READDR           ;GET THE DECREMENTED SECTOR AND TRACK ADDRESSES
017666  112660  000033                  MOVB    (SP)+,$PREVA+1(R0)   ;TRACK ADDRESS
017672  112660  000032                  MOVB    (SP)+,$PREVA(R0)   ;SECTOR ADDRESS
017676  016060  000272  000034          MOV     $RPCC(R0),$PREVA+2(R0)   ;CURRENT CYLINDER
017704  032777  000100  161226  2$:     BIT     #SW06,@SWR          ;SWITCH 6 SET ?
017712  001043                          BNE     3$                  ;BR IF SET
017714  116060  000074  000024          MOVB    $NCODE(R0),$CODE(R0)   ;LOGICAL CODE FOR OPERATION
017722  116005  000074                  MOVB    $NCODE(R0),R5    ;LOAD R5 FOR USE AS TABLE INDEX
017726  116560  001760  000002          MOVB    COMTBL(R5),$COMND(R0)   ;RP04 COMMAND CODE
017734  116060  000075  000030          MOVB    $NPATC(R0),$PATTC(R0)   ;PATTERN CODE
017742  016060  000076  000010          MOV     $NSEC(R0),$SEC(R0)   ;TRACK AND SECTOR ADDRESSES
017750  016060  000100  000012          MOV     $NCYL(R0),$CYL(R0)   ;CYLINDER ADDRESS
017756  016060  000102  000020          MOV     $NWRDL(R0),$WRDL(R0)   ;BUFFER SIZE
017764  016060  000102  000004          MOV     $NWRDL(R0),$WRDM(R0)   ;WORD COUNT FOR THE RH11
017772  005460  000004                  NEG     $WRDM(R0)           ;COMPLEMENT IT
017776  012760  000400  000022          MOV     #256.,$SSEC(R0)     ;INITIAL VALUE OF SECTOR SIZE
020004  032760  000001  000024          BIT     #1,$CODE(R0)        ;HEADER OPERATION ?
020012  001403                          BEQ     3$                  ;BR IF NOT
020014  062760  000004  000022          ADD     #4,$SSEC(R0)        ;ADD HEADER SIZE
020022  005060  000104          3$:     CLR     $NEXT(R0)           ;RESET 'PARAMETERS LOADED' INDICATOR
020026  012605                          MOV     (SP)+,R5            ;RESTORE R5
020030  000207                          RTS     PC                  ;RETURN

                                ;ROUTINE TO COMPRESS A LIST
                                ;CALL:
                                ;       MOV     #ADDRS,R1           ;COMPRESS LIST STARTING AT THIS ADDRESS
                                ;       JSR     PC,CMPRES
                                ;       RETURN

020032  016111  000002          CMPRES: MOV     2(R1),(R1)          ;COMPRESS THE TABLE IN R1
020036  001403                          BEQ     1$                  ;BR WHEN ZERO FOUND
020040  062701  000002                  ADD     #2,R1               ;INCREMENT R1
020044  000772                          BR      CMPRES              ;CONTINUE COMPRESSING TABLE
020046  000207                  1$:     RTS     PC                  ;RETURN

                                ;ROUTINE TO SETUP PARAMETERS FOR A SEQUENTIAL READ OR WRITE OF THE DISK
                                ;CALL:
                                ;       MOV     #DPB,R0             ;DPB ADDRESS
                                ;       MOV     #-1,$PACK(R0)       ;'WRITE PACK' FLAG
                                ;  OR
                                ;       MOV     #1,$PACK(R0)        ;'READ PACK' FLAG
                                ;       JSR     PC,WRTPK
                                ;       RETURN

020050  004737  033534          WRTPK:  JSR     PC,$RAND            ;CYCLE THE RANDOM NUMBER GENERATOR
020054  005760  000040                  TST     $OPERC+2(R0)        ;SEE IF FIRST OPERATION
020060  001007                          BNE     WRTPK1              ;BR IF UPPER WORD OF COUNTER NOT ZERO
020062  005760  000036                  TST     $OPERC(R0)          ;LOWER WORD ZERO ?
020066  001004                          BNE     WRTPK1              ;BR IF NOT 1ST OPERATION
020070  105760  000026                  TSTB    $PACK(R0)           ;SEE WHICH - 'R' OR 'W'
020074  100503                          BMI     WRTPK3              ;BR IF 'W'
020076  000470                          BR      WRTPK2              ;'R' OPERATION
020100  116060  000234  000027  WRTPK1: MOVB    $RPCS1(R0),$PREVO(R0)   ;SAVE CURRENT PARAMETERS
```

```
020106  004737  022616                   JSR     PC,READDR             ;GET THE DECREMENTED SECTOR AND TRACK ADDRESSES
020112  112660  000033                   MOVB    (SP)+,$PREVA+1(R0)    ;TRACK ADDRESS
020116  112660  000032                   MOVB    (SP)+,$PREVA(R0)      ;SECTOR ADDRESS
020122  016060  000272  000034           MOV     $RPCC(R0),$PREVA+2(R0)  ;CURRENT CYLINDER
020130  016060  000242  000010           MOV     $RPDA(R0),$SEC(R0)    ;NEW SECTOR & TRACK ADDRESS
020136  016060  000270  000012           MOV     $RPCA(R0),$CYL(R0)    ;NEW CYLINDER ADDRESS
020144  026060  000012  000106           CMP     $CYL(R0),MAXCYL(R0)   ;SEE IF AT END
020152  103427                           BLO     2$                    ;BR IF LESS THAN 'MAXCYL'
020154  101004                           BHI     1$                    ;BR IF GREATER THAN 'MAXCYL'
020156  126060  000011  000112           CMPB    $TRK(R0),MAXTRK(R0)   ;SEE IF AT MAX TRACK
020164  101422                           BLOS    2$                    ;BR IF NOT GREATER
020166  116060  000114  000011  1$:      MOVB    MINTRK(R0),$TRK(R0)   ;RESET TRACK ADDRESS
020174  116060  000120  000010           MOVB    MINSEC(R0),$SEC(R0)   ;RESET SECTOR ADDRESS
020202  016060  000110  000012           MOV     MINCYL(R0),$CYL(R0)   ;RESET CYLINDER ADDRESS
020210  004737  027140                   JSR     PC,EOP2               ;DROP THE DRIVE (NORMAL TERMINATION)
020214  032777  000020  160716           BIT     #SW04,@SWR            ;IS SWITCH 4 SET ?
020222  001003                           BNE     2$                    ;BR IF SET
020224  005726                           TST     (SP)+                 ;INCREMENT THE STACK POINTER
020226  000137  005676                   JMP     MAIN                  ;RETURN DIRECTLY TO 'MAIN'
020232  013760  001404  000020  2$:      MOV     MAXDL,$WRDL(R0)       ;BUFFER SIZE IS MAXIMUM
020240  013760  001404  000004           MOV     MAXDL,$WRDM(R0)       ;WORD COUNT
020246  005460  000004                   NEG     $WRDM(R0)             ;CHANGE WORD COUNT TO 2'S COMPLEMENT
020252  105760  000026                   TSTB    $PACK(R0)             ;READ OR WRITE ?
020256  100412                           BMI     WRTPK3                ;BR IF WRITE
020260  012760  000404  000022  WRTPK2:  MOV     #260.,$SSEC(R0)       ;SECTOR SIZE FOR READ
020266  112760  000005  000024           MOVB    #5,$CODE(R0)          ;CODE FOR READ HEADER & DATA
020274  112760  000173  000002           MOVB    #RDHD,$COMND(R0)      ;DRIVE CODE FOR OPERATION
020302  000415                           BR      WRTPK4                ;SET UP FOR EXIT
020304  012760  000400  000022  WRTPK3:  MOV     #256.,$SSEC(R0)       ;SECTOR SIZE
020312  112760  000002  000024           MOVB    #2,$CODE(R0)          ;CODE FOR WRTDAT
020320  112760  000161  000002           MOVB    #WRTDAT,$COMND(R0)    ;OP CODE
020326  004737  017576                   JSR     PC,GETPAT             ;GET PATTERN CODE
020332  110560  000030                   MOVB    R5,$PATTC(R0)         ;PATTERN CODE
020336  005060  000104          WRTPK4:  CLR     $NEXT(R0)             ;CLEAR 'PARAMETER SELECTED' INDICATOR
020342  000207                           RTS     PC                    ;RETURN

                                ;ROUTINE TO DETERMINE OF ERROR IS AT A LOCATION ON THE PACK DEFINED
                                ;        IN THE BAD TRACK/SECTOR TABLE FOR THE DRIVE.
                                ;CALL:
                                ;        JSR     PC,SPOTCK
                                ;        RETURN1                       ;ERROR AT AN ADDRESS IN TABLE
                                ;        RETURN2                       ;NO TABLE ENTRY FOR ERROR ADDRESS OR
                                ;                                      ;PARAMETER 'NOTPRT' IS 0

020344                          SPOTCK:
020344  010146                           MOV     R1,-(SP)              ;;PUSH R1 ON STACK
020346  010246                           MOV     R2,-(SP)              ;;PUSH R2 ON STACK
020350  012701  000124                   MOV     #$BDSEC,R1            ;INCREMENT FOR BAD SECTOR TABLE
020354  060001                           ADD     R0,R1                 ;ADD THE BLOCK'S STARTING ADDRESS
020356  012702  000020                   MOV     #16.,R2               ;BAD SECTOR TABLE SIZE COUNT
020362  021160  000272          1$:      CMP     (R1),$RPCC(R0)        ;IS CYLINDER IN THE TABLE ?
020366  001022                           BNE     4$                    ;BR IF NOT
020370  105761  000003                   TSTB    3(R1)                 ;TRACK ENTRY ?
020374  100426                           BMI     5$                    ;BR IF NOT
020376  004737  022616                   JSR     PC,READDR             ;DECREMENT THE SECTOR/TRACK ADDRESS
020402  122661  000003                   CMPB    (SP)+,3(R1)           ;COMPARE THE TRACK ADDRESS
020406  001011                           BNE     3$                    ;BR IF IT IS NOT EQUAL
```

```
   1452 020410  105761  000002                 TSTB    2(R1)               ;IS A SECTOR ADDRESS IN THE TABLE ?
   1453 020414  100002                         BPL     2$                  ;BR IF ONE IS
   1454 020416  005726                         TST     (SP)+               ;INCREMENT THE STACK POINTER
   1455 020420  000414                         BR      5$                  ;DISPLAY THE MESSAGE
   1456 020422  122661  000002          2$:    CMPB    (SP)+,2(R1)         ;COMPARE THE SECTOR ADDRESS
   1457 020426  001002                         BNE     4$                  ;BR IF NOT EQUAL
   1458 020430  000410                         BR      5$                  ;CHECK 'NOTPRT'
   1459 020432  005726                  3$:    TST     (SP)+               ;INCREMENT THE STACK POINTER
   1460 020434  062701  000004          4$:    ADD     #4,R1               ;GO TO THE NEXT LOCATION IN THE TABLE
   1461 020440  005711                         TST     (R1)                ;PAST THE TABLE ENTRIES ?
   1462 020442  100411                         BMI     6$                  ;BR IF PAST
   1463 020444  005302                         DEC     R2                  ;DECREMENT THE MAXIMUM ENTRY COUNT
   1464 020446  001345                         BNE     1$                  ;BR IF MORE TO CHECK
   1465 020450  000406                         BR      6$                  ;END, EXIT
   1466 020452  005737  001426          5$:    TST     NOTPRT              ;PRINT THE ERROR ANYWAY ?
   1467 020456  001006                         BNE     7$                  ;BR IF NOT
   1468 020460  012737  177777  001264         MOV     #-1,BADSEC          ;SET THE INDICATOR FOR THE IDENTIFICATION LINE
   1469 020466  062766  000002  000004  6$:    ADD     #2,4(SP)            ;INCREMENT THE RETURN
   1470 020474                          7$:
        020474  012602                         MOV     (SP)+,R2            ;;POP STACK INTO R2
        020476  012601                         MOV     (SP)+,R1            ;;POP STACK INTO R1
   1471 020500  000207                         RTS     PC                  ;RETURN
```

```
                                        .SBTTL  ERROR MESSAGE GENERATION ROUTINES

                                        ;PRINT LINE 1 OF ERROR MESSAGE:
                                        ;'HH:MM:SS'

020502  032777  002000  160430  LINE1:  BIT     #SW10,@SWR      ;SWITCH 10 SET ?
020510  001402                          BEQ     1$              ;BR IF NOT
020512  104401  001160                  TYPE    ,$BELL          ;RING THE BELL
020516  032777  020000  160414  1$:     BIT     #SW13,@SWR      ;INHIBIT TYPEOUT ?
020524  001403                          BEQ     2$              ;BR IF NOT
020526  104414  001165                  DISPLY  ,$CRLF          ;CR-LF
020532  000404                          BR      3$              ;EXIT
020534  004737  023752          2$:     JSR     PC,$TIME        ;TYPE THE TIME
020540  104414  053364                  DISPLY  ,LINSPO         ;SPACES
020544  000207                  3$:     RTS     PC              ;RETURN & TYPE DESCRIPTION

                                        ;PRINT LINE 2 OF ERROR MESSAGE
                                        ;'PRESENT ORDER = XXXX  PREVIOUS ORDER = XXXX'
                                        ;'* ERROR AT BAD TRACK/SECTOR'
                                        ;'DRV RPCS1   RPCS2   RPDS1   RPER1   RPER2   RPER3   RPEC1   RPEC2'
                                        ;'RPWC    RPBA    RPDA    RPAS    RPLA    RPDB   RPMR   RPDT'
                                        ;'RPSN    RPOF    RPCA    RPCC    STATUS'
                                        ;'BUS ADDRESS OR WORD COUNT NOT CONSISTENT'
                                        ;'RPBA = XXXXXX   RPWC = XXXXXX'
                                        ;'BUFFER ADR = XXXXXX   SIZE = XXXX   ACTUAL NMBR WRDS XFRD = XXX'

020546                          LINE2:
020546  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
020550  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
020552  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
020554  104414  001165                  DISPLY  ,$CRLF          ;CR-LF
020560  005037  020706                  CLR     4$              ;CLEAR MESSAGE ADDRESS STORAGE
020564  005004                          CLR     R4              ;WORKING REGISTER
020566  012737  051462  020706          MOV     #LIN2C,4$       ;ADDRESS OF 'PRESENT ORDER = ' MSG
020574  116004  000234                  MOVB    $RPCS1(R0),R4   ;GET THE OPCODE
020600  042704  177701                  BIC     #^C76,R4        ;SAVE ONLY SIGNIFICANT BITS
020604  004737  020642                  JSR     PC,1$           ;TYPE THE FIRST MNEMONIC
020610  005737  020712                  TST     5$              ;SEE IF MNEMONIC ENTRY FOUND
020614  001440                          BEQ     LINE2A          ;BR IF NOT
020616  012737  051503  020706          MOV     #LIN2P,4$       ;ADDRESS OF 'PREVIOUS ORDER = ' MSG
020624  116004  000027                  MOVB    $PREVO(R0),R4   ;PREVIOUS OPERATION CODE
020630  042704  177701                  BIC     #^C76,R4        ;SAVE ONLY SIGNIFICANT BITS
020634  004737  020642                  JSR     PC,1$           ;TYPE THE PREVIOUS MNEMONIC
020640  000426                          BR      LINE2A          ;CONTINUE
020642  005005                  1$:     CLR     R5              ;CLEAR THE TABLE INDEX
020644  126504  001766          2$:     CMPB    OPTBL(R5),R4    ;LOOK FOR THE OPCODE
020650  001405                          BEQ     3$              ;BR WHEN OPCODE COUNT EQUALS OPCODE
020652  105765  001766                  TSTB    OPTBL(R5)       ;LOOK FOR END OF TABLE
020656  100402                          BMI     3$              ;BR IF END
020660  005205                          INC     R5              ;INCREMENT THE POINTER
020662  000770                          BR      2$              ;CONTINUE - NOT END OF TABLE
020664  006305                  3$:     ASL     R5              ;SHIFT INDEX
020666  006305                          ASL     R5              ;SHIFT THE INDEX
020670  006305                          ASL     R5              ;SHIFT THE INDEX
020672  012737  002010  020712          MOV     #MNTBL,5$       ;ADDRESS OF ASCII TEXT TABLE
020700  060537  020712                  ADD     R5,5$           ;ADD THE INDEX
020704  104414                          DISPLY                  ;TYPE IT
```

```
020706  000000                  4$:     .WORD   0                 ;ADDRESS OF 'PRESENT' OR 'PREVIOUS' MESSAGE
020710  104414                          DISPLY                    ;TYPE THE OPERATION MNEMONIC
020712  000000                  5$:     .WORD   0                 ;ADDRESS OF MESSAGE
020714  000207                          RTS     PC                ;RETURN TO MAIN ROUTINE
020716  005737  001264          LINE2A: TST     BADSEC            ;PRINT THE BAD SECTOR LINE ?
020722  001404                          BEQ     LINE2B            ;BR IF NOT
020724  104414  001165                  DISPLY  .$CRLF            ;CR-LF
020730  104414  051527                  DISPLY  .LIN2S            ;ERROR ADDRESS DEFINED AS BAD AREA
020734  104414  001165          LINE2B: DISPLY  .$CRLF            ;CR-LF
020740  104414  051056                  DISPLY  .DH14             ;STANDARD RP04/5/6 REGISTER HEADER
020744  104414  053364                  DISPLY  .LINSPO           ;TYPE A SPACE
020750  013746  001246                  MOV     UNIT,-(SP)        ;PUT THE DRIVE NUMBER ON THE STACK
020754  004737  022576                  JSR     PC,LINDEC         ;TYPE DRIVE NUMBER
020760  104414  053363                  DISPLY  .LINSP            ;SPACES
020764  012705  051402                  MOV     #DT14,R5          ;REGISTER INDEXES
020770  004737  021120                  JSR     PC,3$             ;PRINT THE REGISTERS
020774  032777  000040  160136          BIT     #SW05,@SWR        ;PRINT THE OPTIONAL REGISTERS ?
021002  001014                          BNE     1$                ;BR IF NOT
021004  104414  051162                  DISPLY  .DH15
021010  012705  051424                  MOV     #DT15,R5          ;SECOND DATA LINE
021014  004737  021120                  JSR     PC,3$             ;PRINT THEM
021020  104414  051261                  DISPLY  .DH16
021024  012705  051446                  MOV     #DT16,R5          ;THIRD DATA LINE
021030  004737  021120                  JSR     PC,3$             ;PRINT THE REGISTERS
021034  032760  000100  000016  1$:     BIT     #BIT6,$STATUS(R0)   ;DATA ERROR ?
021042  001422                          BEQ     2$                ;BR IF NOT
021044  016046  000020                  MOV     $WRDL(R0),-(SP)   ;TRANSFER SIZE
021050  066016  000236                  ADD     $RPWC(R0),(SP)    ;ADD REMAINING WORD COUNT
021054  006316                          ASL     (SP)              ;CONVERT TO AN BYTE INCREMENT
021056  066016  000006                  ADD     $BUF(R0),(SP)     ;BUFFER STARTING ADDRESS
021062  022660  000240                  CMP     (SP)+,$RPBA(R0)   ;CORRECT BUFFER ADDRESS ?
021066  001410                          BEQ     2$                ;BR IF YES
021070  104414  050451                  DISPLY  .EM46             ;'BUS ADDRESS AND WORD COUNT ARE NOT CONSISTENT'
021074  104414  001165                  DISPLY  .$CRLF            ;CR-LF
021100  004737  021212                  JSR     PC,LINE3D         ;PRINT LINE 3D OF ERROR MESSAGE
021104  004737  021630                  JSR     PC,LINE4          ;PRINT LINE 4 OF ERROR MESSAGE
021110                          2$:
021110  012605                          MOV     (SP)+,R5          ;;POP STACK INTO R5
021112  012604                          MOV     (SP)+,R4          ;;POP STACK INTO R4
021114  012603                          MOV     (SP)+,R3          ;;POP STACK INTO R3
021116  000207                          RTS     PC                ;RETURN TO ERROR PROCESSING ROUTINE
021120  012546                  3$:     MOV     (R5)+,-(SP)       ;PUT THE REGISTER INDEX ON THE STACK
021122  060016                          ADD     R0,(SP)           ;ADD DRIVE'S TABLE ADDRESS
021124  017646  000000                  MOV     @(SP),-(SP)       ;VALUE
021130  004737  022544                  JSR     PC,LINOCT         ;TYPE IT
021134  005726                          TST     (SP)+             ;CORRECT THE STACK POINTER
021136  104414  053363                  DISPLY  .LINSP            ;PRINT 2 SPACES
021142  005715                          TST     (R5)              ;AT END OF LINE ?
021144  001365                          BNE     3$                ;BR IF NOT
021146  104414  001165          4$:     DISPLY  .$CRLF            ;CR-LF
021152  000207                          RTS     PC                ;RETURN

                                ;PRINT LINE 3 OF ERROR MESSAGE
                                ;'ERROR AT CCC TT SS   PREVIOUS ADR = CCC TT SS'

021154  104414  051563          LINE3:  DISPLY  .LINM3            ;LINE 3 ENTRANCE
021160  000517                          BR      LIN3.1            ;FINISH PRINTOUT
```

```

                                        ;PRINT LINE 3A OF ERROR MESSAGE
                                        ;'START CYL = CCC   END CYL = CCC'

021162  104414  051601          LINE3A: DISPLY  ,LINN3          ;LINE 3A ENTRANCE
021166  000514                          BR      LIN3.1          ;FINISH ERROR LINE

                                        ;PRINT LINE 3B OF ERROR MESSAGE
                                        ;'START CYL = CCC   END CYL = CCC   ACTUAL CYL = CCC'

021170  004737  021532          LINE3B: JSR     PC,LIN3.3       ;LINE 3B ENTRANCE
021174  104414  001165                  DISPLY  ,$CRLF
021200  000207                          RTS     PC

                                        ;PRINT LINE 3C OF ERROR MESSAGE
                                        ;'START CYL = CCC   END CYL = CCC   ACTUAL CYL = CCC   TRK = TT'

021202  004737  021532          LINE3C: JSR     PC,LIN3.3       ;LINE 3C ENTRANCE
021206  000137  021564                  JMP     LIN3.4          ;FINISH MESSAGE

                                        ;PRINT LINE 3D OF ERROR MESSAGE
                                        ;'RPBA = XXXXXX   RPWC = XXXXXX'

021212  032777  000040  157720  LINE3D: BIT     #SW05,@SWR      ;SWITCH 5 SET ?
021220  001416                          BEQ     1$              ;BR IF IT IS
021222  104414  051752                  DISPLY  ,LINB3          ;'RPBA = '
021226  016046  000240                  MOV     $RPBA(R0),-(SP)  ;BUFFER ADDR REG CONTENTS
021232  004737  022544                  JSR     PC,LINOCT       ;CONVERT TO OCTAL AND TYPE IT
021236  104414  051762                  DISPLY  ,LINW3          ;'  RPWC = '
021242  016046  000236                  MOV     $RPWC(R0),-(SP) ;WORD COUNT REGISTER CONTENTS
021246  004737  022544                  JSR     PC,LINOCT       ;CONVERT TO OCTAL AND TYPE IT
021252  104414  001165                  DISPLY  ,$CRLF
021256  000207                  1$:     RTS     PC

                                        ;PRINT LINE 3E OF ERROR MESSAGE
                                        ;'START CYL = CCC   START TRK = TT   START SEC = SS'

021260  104414  051646          LINE3E: DISPLY  ,LINS3          ;'START CYL = '
021264  016046  000012                  MOV     $CYL(R0),-(SP)  ;MOVE CYL TO STACK
021270  004737  022576                  JSR     PC,LINDEC       ;TYPE IT IN DECIMAL
021274  104414  053363                  DISPLY  ,LINSP          ;SPACES
021300  104414  051774                  DISPLY  ,LINST3         ;'START TRK = '
021304  005046                          CLR     -(SP)           ;CLEAR STACK
021306  116016  000011                  MOVB    $TRK(R0),(SP)   ;TRACK TO STACK
021312  004737  022576                  JSR     PC,LINDEC       ;TYPE IT IN DECIMAL
021316  104414  053363                  DISPLY  ,LINSP          ;SPACES
021322  104414  052011                  DISPLY  ,LINSS3         ;'START SEC = '
021326  005046                          CLR     -(SP)           ;CLEAR STACK
021330  116016  000010                  MOVB    $SEC(R0),(SP)   ;SECTOR ADDR TO STACK
021334  004737  022576                  JSR     PC,LINDEC       ;TYPE IT IN DECIMAL
021340  104414  001165                  DISPLY  ,$CRLF
021344  000207                          RTS     PC

                                        ;PRINT LINE 3F OF ERROR MESSAGE
                                        ;'RPDA = XXXXXX   RPCA = XXXXXX'

021346  032777  000040  157564  LINE3F: BIT     #SW5,@SWR       ;SWITCH 5 SET ?
```

```
021354  001420                   BEQ     1$              ;BR IF NOT
021356  104414  051742           DISPLY  ,LINDA3         ;'RPDA = '
021362  016046  000242           MOV     $RPDA(R0),-(SP) ;PUT SECTOR/TRACK ADDRESS ON THE STACK
021366  004737  022544           JSR     PC,LINOCT       ;TYPE IT
021372  104414  053363           DISPLY  ,LINSP          ;SPACES
021376  104414  051731           DISPLY  ,LINCA3         ;' RPCA = '
021402  016046  000270           MOV     $RPCA(R0),-(SP) ;PUT DESIRED CYLINDER ADDRESS ON THE STACK
021406  004737  022544           JSR     PC,LINOCT       ;TYPE IT
021412  104414  001165           DISPLY  ,$CRLF
021416  000207          1$:      RTS     PC

                        ;'CCC TT SS   PREV ADR = CCC TT SS'

021420  016046  000272  LIN3.1:  MOV     $RPCC(R0),-(SP) ;PUT CYLINDER ADDR ON STACK
021424  004737  022576           JSR     PC,LINDEC       ;TYPE IT IN DECIMAL
021430  104414  051576           DISPLY  ,T              ;PRINT ' T'
021434  004737  022616           JSR     PC,READDR       ;DECREMENT TRACK AND SECTOR ADDRESSES
021440  004737  022576           JSR     PC,LINDEC       ;TYPE TRACK IN DECIMAL
021444  104414  051622           DISPLY  ,S              ;PRINT ' S'
021450  004737  022576           JSR     PC,LINDEC       ;TYPE SECTOR ADDRESS
021454  104414  051625           DISPLY  ,LINP3          ;PRINT 'PREV ADDR'
021460  016046  000034           MOV     $PREVA+2(R0),-(SP) ;PREVIOUS CYLINDER
021464  004737  022576           JSR     PC,LINDEC       ;TYPE IT IN DECIMAL
021470  104414  051576           DISPLY  ,T              ;PRINT ' T'
021474  005046                   CLR     -(SP)           ;MAKE ROOM ON THE STACK
021476  116016  000033           MOVB    $PREVA+1(R0),(SP)  ;PREVIOUS TRACK ADDRESS
021502  004737  022576           JSR     PC,LINDEC       ;TYPE IT IN DECIMAL
021506  104414  051622           DISPLY  ,S              ;PRINT ' S'
021512  005046                   CLR     -(SP)           ;MAKE ROOM ON THE STACK
021514  116016  000032           MOVB    $PREVA(R0),(SP)  ;PREVIOUS SECTOR DDRESS
021520  004737  022576           JSR     PC,LINDEC       ;TYPE IT IN DECIMAL
021524  104414  001165           DISPLY  ,$CRLF
021530  000207                   RTS     PC

                        ;'START CYL = CCC   END CYL = CCC'

021532  104414  051646  LIN3.3:  DISPLY  ,LINS3          ;LINE '3B & 3C' ENTRANCE
021536  016046  000034           MOV     $PREVA+2(R0),-(SP) ;PREVIOUS CYLINDER
021542  004737  022576           JSR     PC,LINDEC       ;TYPE IT IN DECIMAL
021546  104414  051663           DISPLY  ,LINEN3         ;PRINT 'END CYL'
021552  016046  000272           MOV     $RPCC(R0),-(SP) ;PRESENT CYLINDER
021556  004737  022576           JSR     PC,LINDEC       ;TYPE IT IN DECIMAL
021562  000207                   RTS     PC

                        ;'ACTUAL CYL = CCC   TRK = TT'

021564  104414  051700  LIN3.4:  DISPLY  ,LINA3          ;PRINT 'ACTUAL'
021570  013746  055754           MOV     CYLDER,-(SP)    ;ACTUAL CYLINDER
021574  042716  010000           BIC     #BIT12,(SP)     ;CLEAR THE FORMAT BIT
021600  004737  022576           JSR     PC,LINDEC       ;TYPE IT IN DECIMAL
021604  104414  051720           DISPLY  ,LINT3          ;PRINT TRACK
021610  005046                   CLR     -(SP)           ;CLEAR STACK WORD
021612  116016  000243           MOVB    $RPDA+1(R0),(SP) ;PUT TRACK ON STACK
021616  004737  022576           JSR     PC,LINDEC       ;TYPE IT IN DECIMAL
021622  104414  001165           DISPLY  ,$CRLF
021626  000207                   RTS     PC

```

```
                                        ;PRINT LINE 4 OF ERROR MESSAGE
                                        ;'BUFFER ADR = XXXXXX   SIZE = XXXX   ACTUAL NMBR WRDS XFRD = XXX'

021630  032760  000100  000016  LINE4:  BIT     #BIT06,$STATUS(R0)  ;DATA ERROR ?
021636  001427                          BEQ     1$                  ;BR IF NOT
021640  104414  052026                  DISPLY  ,LINM4              ;'PRINT BUFFER'
021644  016046  000006                  MOV     $BUF(R0),-(SP)      ;BUFFER ADDR ON STACK
021650  004737  022544                  JSR     PC,LINOCT           ;CONVERT TO OCTAL & PRINT
021654  104414  052045                  DISPLY  ,LINS4              ;PRINT 'SIZE'
021660  016046  000020                  MOV     $WRDL(R0),-(SP)     ;BUFFER SIZE
021664  004737  022576                  JSR     PC,LINDEC           ;TYPE IT IN DECIMAL
021670  104414  052057                  DISPLY  ,LINX4              ;'ACTUAL NMBR WRDS XFRD = '
021674  016046  000240                  MOV     $RPBA(R0),-(SP)     ;VALUE IN BUFFER ADDR REGISTER
021700  166016  000006                  SUB     $BUF(R0),(SP)       ;SUBTRACT STARTING ADDRESS
021704  006216                          ASR     (SP)                ;CONVERT INTO A WORD COUNT
021706  004737  022576                  JSR     PC,LINDEC           ;TYPE IT IN DECIMAL
021712  104414  001165                  DISPLY  ,$CRLF              ;CR-LF
021716  000207                  1$:     RTS     PC                  ;RETURN

                                        ;PRINT LINE 5 OF ERROR MESSAGE
                                        ;'GOOD DATA = XXXXXX   BAD DATA = XXXXXX   SECT POS = XXX'

021720  104414  052112          LINE5:  DISPLY  ,LIND5              ;PRINT 'GOOD DATA'
021724  162760  000002  000240          SUB     #2,$RPBA(R0)        ;BACK THE ADDRESS UP
021732  017046  000240                  MOV     @$RPBA(R0),-(SP)    ;'GOOD' DATA - AT THE BUFFER LOCATION
021736  004737  022544                  JSR     PC,LINOCT           ;TYPE IT
021742  104414  052127                  DISPLY  ,LINB5              ;PRINT 'BAD DATA'
021746  016046  000256                  MOV     $RPDB(R0),-(SP)     ;BAD DATA FROM BUFFER
021752  004737  022544                  JSR     PC,LINOCT           ;TYPE IT
021756  016046  000236                  MOV     $RPWC(R0),-(SP)     ;WORD LENGTH ON STACK
021762  066016  000020                  ADD     $WRDL(R0),(SP)      ;MAKE INTO A POSITIVE NUMBER
021766  005046                          CLR     -(SP)               ;UPPER DIVIDEND TO ZERO
021770  016046  000022                  MOV     $SSEC(R0),-(SP)     ;SECTOR SIZE ON THE STACK
021774  004737  027366                  JSR     PC,LINKDV           ;DIVIDE WORDS XFERED BY SECTOR SIZE
022000  012616                          MOV     (SP)+,(SP)          ;MOVE REMAINDER UP THE STACK
022002  104414  052145                  DISPLY  ,LINP5              ;PRINT 'SECT POS'
022006  004737  022576                  JSR     PC,LINDEC           ;TYPE THE POSITION
022012  104414  001165                  DISPLY  ,$CRLF
022016  000207                          RTS     PC

                                        ;PRINT LINE 5A OF THE ERROR MESSAGE
                                        ;'HEADER FROM ERROR SECTOR  XXXXXX  XXXXXX  XXXXXX  XXXXXX'

022020  104414  052163          LINE5A: DISPLY  ,LINS5              ;'HEADER CONTENTS OF ERROR SECTOR'
022024  013746  055754                  MOV     CYLDER,-(SP)        ;HEADER POSITION
022030  004737  022544                  JSR     PC,LINOCT           ;TYPE IT
022034  104414  053363                  DISPLY  ,LINSP              ;SPACES
022040  013746  055756                  MOV     CYLDER+2,-(SP)      ;HEADER POSITION +2
022044  004737  022544                  JSR     PC,LINOCT           ;TYPE IT
022050  104414  053363                  DISPLY  ,LINSP              ;SPACES
022054  013746  055760                  MOV     CYLDER+4,-(SP)      ;HEADER POSITION +4
022060  004737  022544                  JSR     PC,LINOCT           ;TYPE IT
022064  104414  053363                  DISPLY  ,LINSP              ;SPACES
022070  013746  055762                  MOV     CYLDER+6,-(SP)      ;HEADER POSITION +6
022074  004737  022544                  JSR     PC,LINOCT           ;TYPE IT
022100  104414  053363                  DISPLY  ,LINSP              ;SPACES
022104  104414  001165                  DISPLY  ,$CRLF
```

```
022110  000207                          RTS     PC

                                ;PRINT LINE 5B OF ERROR MESSAGE
                                ;'RPEC1 = XXXXXX   RPEC2 = XXXXXX'

022112  104414  052217          LINE5B: DISPLY  ,LINEP5         ;'RPEC1 = '
022116  016046  000300                  MOV     $RPEC1(R0),-(SP)  ;PUT REGISTER CONTENTS ON THE STACK
022122  004737  022544                  JSR     PC,LINOCT       ;TYPE IT
022126  104414  053363                  DISPLY  ,LINSP          ;SPACES
022132  104414  052230                  DISPLY  ,LINE05         ;' RPEC2 = '
022136  016046  000302                  MOV     $RPEC2(R0),-(SP)  ;PUT REGISTER CONTENTS ON THE STACK
022142  004737  022544                  JSR     PC,LINOCT       ;TYPE IT
022146  104414  001165                  DISPLY  ,$CRLF
022152  000207                          RTS     PC              ;RETURN

                                ;PRINT LINE 6 OF ERROR MESSAGE
                                ;'SECTOR IS ECC CORRECTABLE'

022154  104414  052242          LINE6:  DISPLY  ,LINB6          ;ECC CORRECTABLE
022160  104414  001165                  DISPLY  ,$CRLF
022164  000207                          RTS     PC

                                ;PRINT LINE 6A OF THE ERROR MESSAGE
                                ;'SECTOR READ CORRECTLY AT OFFSET N'

022166  104414  052275          LINE6A: DISPLY  ,LINC6          ;PRINT 'READ CORRECTLY AT OFFSET N'
022172  000411                          BR      LIN6.1          ;TYPE THE REST OF THE LINE

                                ;PRINT LINE 6B OF THE ERROR MESSAGE
                                ;'SECTOR IS ECC CORRECTABLE AT OFFSET N'

022174  104414  052242          LINE6B: DISPLY  ,LINB6          ;PRINT 'SECTOR IS ECC CORRECTABLE '
022200  000406                          BR      LIN6.1

                                ;PRINT LINE 6C OF THE ERROR MESSAGE
                                ;'CORRECTED ON NTH RETRY'

022202  104414  052324          LINE6C: DISPLY  ,LING6          ;'CORRECTED ON NTH RETRY'
022206  000414                          BR      LIN6.2          ;TYPE THE REST OF THE LINE

                                ;PRINT LINE 6D OF THE ERROR MESSAGE
                                ;'UNCORRECTABLE AFTER N RETRIES'

022210  104414  052353          LINE6D: DISPLY  ,LINU06         ;'UNCORRECTABLE AFTER N RETRIES'
022214  000411                          BR      LIN6.2          ;FINISH

                                ;TYPE THE OFFSET VALUE IN MICRO-INCHES

022216  006301                  LIN6.1: ASL     R1              ;DOUBLE THE OFFSET TABLE INDEX
022220  016137  002240  022230          MOV     OFMTBL(R1),1$   ;ADDRESS OF OFFSET POSITION MESSAGE
022226  104414                          DISPLY
022230  000000                  1$:     .WORD   0               ;OFFSET VALUE
022232  104414  001165                  DISPLY  ,$CRLF
022236  000207                          RTS     PC

                                ;RETRY COUNT TYPEOUT

```

```
022240  005046                  LIN6.2: CLR     -(SP)                   ;CLEAR STACK
022242  113716  001253                  MOVB    RETRY+1,(SP)            ;RETRY COUNT
022246  004737  022576                  JSR     PC,LINDEC               ;TYPE IT IN DECIMAL
022252  104414  052342                  DISPLY  ,LINR6                  ;'RETRY'
022256  104414  001165                  DISPLY  ,$CRLF
022262  000207                          RTS     PC


                                ;PRINT LINE 7 OF THE ERROR MESSAGE
                                ;'ORDERS:XXXXX   TOTAL ERRORS:XXX   WRDS XFRD:XXXXXXX   WRDS READ:XXXXXXX'

022264  104414  052426          LINE7:  DISPLY  ,LIN7O                  ;PRINT ORDER COUNT
022270  012746  000036                  MOV     #$OPERC,-(SP)           ;TO STACK
022274  060016                          ADD     R0,(SP)                 ;ADD THE BASE ADDRESS
022276  004737  033732                  JSR     PC,$DB2D                ;CONVERT IT
022302  004737  027702                  JSR     PC,$SUPRS               ;PRINT IT
022306  104414  052505                  DISPLY  ,LIN7T                  ;TOTAL ERRORS
022312  016046  000056                  MOV     $TOTAL(R0),-(SP)                ;TO STACK
022316  004737  022576                  JSR     PC,LINDEC               ;TYPE IT IN DECIMAL
022322  104414  052517                  DISPLY  ,LIN7X                  ;PRINT 'WRDS XFR'
022326  012746  000046                  MOV     #$TRANS,-(SP)           ;ADDRESS OF LOW WORD ON STACK
022332  060016                          ADD     R0,(SP)
022334  004737  033732                  JSR     PC,$DB2D                ;CONVERT
022340  004737  027702                  JSR     PC,$SUPRS               ;PRINT
022344  104414  052533                  DISPLY  ,LIN7R                  ;'BITS READ'
022350  012746  000052                  MOV     #$READ,-(SP)            ;LOW WORD ADDRESS
022354  060016                          ADD     R0,(SP)
022356  004737  033732                  JSR     PC,$DB2D                ;CONVERT
022362  004737  027702                  JSR     PC,$SUPRS               ;PRINT
022366  104414  001165                  DISPLY  ,$CRLF
022372  104414  001165                  DISPLY  ,$CRLF
022376  032777  100000  156534          BIT     #SW15,@SWR              ;SEE IF 'HALT ON ERROR' - SWITCH 15
022404  001401                          BEQ     1$                      ;BR IF NOT
022406  000000                          HALT                            ;SWITCH 15 HALT
022410  000207                  1$:     RTS     PC


                                ;PRINT LINE 7A OF ERROR MESSAGE
                                ;'ORDERS:XXXXX   TOTAL SEEKS=XXXXX   TOTAL MISPOS ERR = XXX   TOTAL SKI,OCYL ERR = XXX'

022412  104414  052426          LINE7A: DISPLY  ,LIN7O                  ;'ORDERS = '
022416  012746  000036                  MOV     #$OPERC,-(SP)   ;ORDER COUNT INCREMENT
022422  060016                          ADD     R0,(SP)                 ;ADD BASE ADDRESS
022424  004737  033732                  JSR     PC,$DB2D                ;CONVERT THE COUNT
022430  004737  027702                  JSR     PC,$SUPRS               ;PRINT IT
022434  104414  052436                  DISPLY  ,LIN7P                  ;'TOTAL SEEKS = '
022440  012746  000042                  MOV     #$POSIT,-(SP)           ;TOTAL SEEKS
022444  060016                          ADD     R0,(SP)                 ;DEVICE TABLE ADDRESS
022446  004737  033732                  JSR     PC,$DB2D                ;CONVERT THE SEEK COUNT
022452  004737  027702                  JSR     PC,$SUPRS               ;PRINT IT
022456  104414  052400                  DISPLY  ,LIN7M                  ;'  TOTAL MISPOS ERR = '
022462  016046  000066                  MOV     $MISPO(R0),-(SP)  ;TOTAL ERRORS
022466  004737  022576                  JSR     PC,LINDEC               ;TYPE IT IN DECIMAL
022472  104414  052456                  DISPLY  ,LIN7S                  ;'  TOTAL SKI,OCYL ERR = '
022476  016046  000064                  MOV     $SKI(R0),-(SP)          ;CONVERT & PRINT IT
022502  004737  022576                  JSR     PC,LINDEC               ;TYPE IT IN DECIMAL
022506  104414  001165                  DISPLY  ,$CRLF
022512  104414  001165                  DISPLY  ,$CRLF
022516  032777  100000  156414          BIT     #SW15,@SWR              ;SEE IF HALT ON ERROR - SWITCH 15 SET
```

```
022524  001401                                  BEQ     1$              ;BR IF NOT
022526  000000                                  HALT                    ;SWITCH 15 HALT
022530  000207                          1$:     RTS     PC

                                        ;PRINT LINE 8 OF THE ERROR MESSAGE
                                        ;'DIFFERENT ERROR DURING RETRY'

022532  104414  052550                  LINE8:  DISPLY  ,LIN8M
022536  004737  020546                          JSR     PC,LINE2        ;PRINT LINE 2 OF ERROR MESSAGE
022542  000207                                  RTS     PC

                                        ;OCTAL TYPEOUT ROUTINE
                                        ;CALL:
                                        ;       MOV     NUM,-(SP)       ;PUT THE NUMBER ON THE STACK
                                        ;       JSR     PC,LINOCT
                                        ;       RETURN

022544  016646  000002                  LINOCT: MOV     2(SP),-(SP)     ;PUT NUMBER IN PROPER LOCATION ON STACK
022550  004737  030332                          JSR     PC,$SB2O        ;CONVERT THE NUMBER TO OCTAL
022554  012637  022570                          MOV     (SP)+,1$        ;GET THE ADDRESS OF THE ASCII STRING
022560  062737  000005  022570                  ADD     #5.,1$          ;ADDRESS THE LAST 6 ASCII DIGITS
022566  104414                                  DISPLY                  ;TYPE IT
022570  000000                          1$:     .WORD   0               ;ADDRESS
022572  012616                                  MOV     (SP)+,(SP)      ;CORRECT THE STACK
022574  000207                                  RTS     PC              ;RETURN

                                        ;ROUTINE TO CONVERT THE INPUT NUMBER TO DECIMAL AND TYPE IT WITH
                                        ;LEADING ZERO SUPRESSION
                                        ;CALL:
                                        ;       MOV     NUM,-(SP)       ;PUT THE NUMBER ON THE STACK
                                        ;       JSR     PC,LINDEC
                                        ;       RETURN

022576  016646  000002                  LINDEC: MOV     2(SP),-(SP)     ;SET UP STACK FOR CONVERT
022602  004737  030302                          JSR     PC,$SB2D        ;CONVERT IT TO DECIMAL
022606  004737  027702                          JSR     PC,$SUPRS       ;TYPE IT (WITH LEADING ZEROS SUPRESSED)
022612  012616                                  MOV     (SP)+,(SP)      ;RESTORE STACK POINTER
022614  000207                                  RTS     PC
```

```
                                        .SBTTL  GENERAL SUPPORT SUBROUTINES

                                        ;DECREMENT THE SECTOR-TRACK ADDRESS
                                        ;CALL:
                                        ;       MOV     #DPB,R0         ;DPB ADDRESS
                                        ;       JSR     PC,READDR
                                        ;       RETURN
                                        ;       (SP) CONTAINS THE TRACK ADDRESS
                                        ;       2(SP) CONTAINS THE SECTOR ADDRESS

022616  162706  000004                  READDR: SUB     #4,SP           ;DECREMENT THE STACK POINTER
022622  016616  000004                          MOV     4(SP),(SP)      ;MOVE THE RETURN ADDR DOWN THE STACK
022626  005066  000004                          CLR     4(SP)           ;CLEAR STACK FOR SECTOR
022632  005066  000002                          CLR     2(SP)           ;CLEAR STACK FOR TRACK
022636  116066  000242  000004                  MOVB    $RPDA(R0),4(SP)  ;INCREMENTED SECTOR ON STACK
022644  005366  000004                          DEC     4(SP)           ;DECREMENT THE SECTOR ADDRESS
022650  100015                                  BPL     1$              ;BR IF SECTOR GREATER THAN 0
022652  012766  000025  000004                  MOV     #21.,4(SP)      ;JAM SECTOR ADDRESS TO 21(10)
022660  116066  000243  000002                  MOVB    $RPDA+1(R0),2(SP)  ;TRACK ADDRESS
022666  005366  000002                          DEC     2(SP)           ;DECREMENT TRACK ADDRESS
022672  100007                                  BPL     2$              ;BR IF IT DIDN'T GO NEG
022674  012766  000022  000002                  MOV     #18.,2(SP)      ;RESET TRACK TO 18(10)
022702  000403                                  BR      2$
022704  116066  000243  000002  1$:     MOVB    $RPDA+1(R0),2(SP)  ;TRACK ADDRESS
022712  000207                          2$:     RTS     PC              ;RETURN

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

022714  012737  177777  001210  CKCLK:  MOV     #-1,CLKFLG      ;CLEAR CLOCK AVAILABILITY FLAG
022722  012737  177777  001206          MOV     #-1,PCLOCK      ;CLEAR KW11-P CLOCK AVAILABILITY FLAG
022730  012737  023010  000004          MOV     #CKCLK1,ERRVEC  ;SET UP VECTOR FOR CLOCK CHECK
022736  005037  000006                          CLR     @#ERRVEC+2        ;NEW PSW
022742  005777  156226                          TST     @$LKCSR         ;CHECK FOR KW11-P
022746  005037  001210                          CLR     CLKFLG          ;SET CLOCK AVAILABILITY FLAG
022752  005037  001206                          CLR     PCLOCK          ;SET KW11-P CLOCK FLAG
022756  013701  001200                          MOV     $LPVEC,R1       ;KW11-P VECTOR ADDRESS
022762  012721  024050                          MOV     #CLOCK,(R1)+    ;SET UP KW11-P VECTOR
022766  012711  000300                          MOV     #300,(R1)       ;PSW - PRI 6
022772  012777  174575  156176                  MOV     #-1667.,@$LKCSB ;LOAD COUNTER BUFFER WITH 16.67
023000  012777  000131  156166                  MOV     #131,@$LKCSR    ;SET CLOCK - CNT UP, 10US, CONT INT
023006  000437                                  BR      CKCLK3
023010  062706  000004                  CKCLK1: ADD     #4,SP           ;RESTORE THE STACK POINTER
023014  012737  023056  000004                  MOV     #CKCLK2,@#ERRVEC  ;CHANGE ERROR VECTOR TO CHECK FOR KW11-L
023022  005777  156154                          TST     @$LKS           ;LOOK FOR KW11-L
023026  005037  001210                          CLR     CLKFLG          ;SET CLOCK FLAG
023032  013701  001204                          MOV     $LLVEC,R1       ;KW11-L VECTOR ADDRESS
023036  012721  024050                          MOV     #CLOCK,(R1)+    ;SET UP KW11-L VECTOR
023042  012711  000300                          MOV     #300,(R1)       ;PSW - PRI 6
023046  012777  000100  156126                  MOV     #100,@$LKS      ;SET KW11-L INTERRUPT
023054  000414                                  BR      CKCLK3
023056  062706  000004                  CKCLK2: ADD     #4,SP           ;RESTORE THE STACK POINTER
023062  104401  054251                          TYPE    ,NEDCLK         ;'P OR L CLOCK MUST BE ON SYSTEM'
023066  005737  000042                          TST     42              ;UNDER MONITOR CONTROL ?
023072  001402                                  BEQ     1$              ;BR IF NOT
023074  000137  005646                          JMP     $GET42          ;ABORT PROGRAM
023100  000000                          1$:     HALT                    ;HALT
023102  000137  004200                          JMP     START1          ;TRY AGAIN
```

```
023106  012737  000006  000004  CKCLK3: MOV     #6,@#ERRVEC         ;RESTORE THE ERROR VECTOR
023114  000207                          RTS     PC

                                ;ROUTINE TO DISPLAY STATISTICS FOR ALL DRIVES ASSIGNED
                                ;CALL:
                                ;       JSR     PC,STATPR
                                ;       RETURN

023116  010046                  STATPR: MOV     R0,-(SP)            ;SAVE R0
023120  010446                          MOV     R4,-(SP)            ;SAVE R4
023122  005737  001462                  TST     ASNLST              ;ANY DRIVES ASSIGNED ?
023126  001421                          BEQ     3$                  ;BR IF NOT
023130  004737  023226                  JSR     PC,SHDTYP           ;TYPE THE HEADING
023134  005004                          CLR     R4                  ;CLEAR THE DRIVE INDEX
023136  006304                  1$:     ASL     R4                  ;CHANGE TO INDEX WORDS
023140  016400  001740                  MOV     BLKADR(R4),R0       ;GET THE DRIVE'S BLOCK ADDRESS
023144  006204                          ASR     R4                  ;RESTORE R4
023146  136437  034470  001462          BITB    ATABIT(R4),ASNLST   ;IS THIS DRIVE ASSIGNED ?
023154  001402                          BEQ     2$                  ;BR IF NOT
023156  004737  023250                  JSR     PC,SDETAL           ;TYPE THE PERFORMANCE SUMMARY
023162  005204                  2$:     INC     R4                  ;INCREMENT THE INDEX
023164  020427  000010                  CMP     R4,#8.              ;FINISHED ?
023170  001362                          BNE     1$                  ;BR IF NOT
023172  012604                  3$:     MOV     (SP)+,R4            ;RESTORE R4
023174  012600                          MOV     (SP)+,R0            ;RESTORE R0
023176  000207                          RTS     PC                  ;RETURN

                                ;ROUTINE TO TYPE STATISTICS FOR AN INDIVIDUAL DRIVE
                                ;CALL:
                                ;       MOV     #DPB,R0             ;DPB ADDRESS
                                ;       JSR     PC,TYPEST
                                ;       RETURN

023200  010046                  TYPEST: MOV     R0,-(SP)            ;SAVE R0
023202  010446                          MOV     R4,-(SP)            ;SAVE R4
023204  004737  023226                  JSR     PC,SHDTYP           ;TYPE THE HEADING
023210  005004                          CLR     R4                  ;CLEAR R4 FOR DRIVE NUMBER
023212  111004                          MOVB    (R0),R4             ;DRIVE NUMBER
023214  004737  023250                  JSR     PC,SDETAL           ;TYPE THE STATISTICS
023220  012604                          MOV     (SP)+,R4            ;RESTORE R4
023222  012600                          MOV     (SP)+,R0            ;RESTORE R0
023224  000207                          RTS     PC                  ;RETURN

                                ;TYPE THE HEADER FOR THE DRIVE PERFORMANCE SUMMARY TYPEOUT
                                ;CALL:
                                ;       JSR     PC,SHDTYP
                                ;       RETURN

023226  004737  023752          SHDTYP: JSR     PC,$TIME            ;TYPE THE TIME OF DAY
023232  004537  027742                  JSR     R5,TYPRI4           ;TYPE AT PRIORITY 4
023236  001165                          $CRLF                       ;CR-LF
023240  004537  027742                  JSR     R5,TYPRI4           ;TYPE THE HEADER
023244  053730                          STATHD                      ;HEADER
023246  000207                          RTS     PC                  ;RETURN

                                ;TYPE THE PERFORMANCE SUMMARY DATE LINE
                                ;CALL:
```

```
                                ;       MOV     #DRIVE,R4       ;DRIVE NUMBER
                                ;       MOV     #DPB,R0         ;DPB ADDRESS
                                ;       RETURN

023250  010246                  SDETAL: MOV     R2,-(SP)        ;SAVE R2
023252  010002                          MOV     R0,R2           ;DPB ADDRESS
023254  062702  000036                  ADD     #$OPERC,R2      ;FIRST STATISTICAL FIELD
023260  010446                          MOV     R4,-(SP)        ;;SAVE R4 FOR TYPEOUT
                                                                ;;TYPE DRIVE NUMBER
023262  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
023264     002                          .BYTE   2               ;;TYPE 2 DIGIT(S)
023265     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
023266  104401  053363                  TYPE    ,LINSP          ;SPACES
023272  016046  000070                  MOV     $PASSC(R0),-(SP)   ;PUT THE PASS COUNT ON THE STACK
023276  004737  030302                  JSR     PC,$SB2D        ;CONVERT IT
023302  004537  027612                  JSR     R5,REPLZ        ;TYPE IT
023306  000003                          .WORD   3               ;TYPE 3 DIGITS
023310  104401  053363                  TYPE    ,LINSP          ;SPACES
023314  010246                          MOV     R2,-(SP)        ;PUT $OPERC ON THE STACK
023316  004737  033732                  JSR     PC,$DB2D        ;CONVERT IT
023322  004537  027612                  JSR     R5,REPLZ        ;TYPE $OPERC
023326  000006                          .WORD   6               ;TYPE 6 DIGITS
023330  104401  053363                  TYPE    ,LINSP          ;SPACES
023334  062702  000004                  ADD     #4,R2           ;INCREMENT R2
023340  010246                          MOV     R2,-(SP)        ;PUT $POSIT ON THE STACK
023342  004737  033732                  JSR     PC,$DB2D        ;CONVERT IT
023346  004537  027612                  JSR     R5,REPLZ        ;TYPE $POSIT
023352  000006                          .WORD   6               ;TYPE 6 DIGITS
023354  104401  053363                  TYPE    ,LINSP          ;SPACES
023360  062702  000004                  ADD     #4,R2           ;INCREMENT R2
023364  010246                          MOV     R2,-(SP)        ;PUT $TRANS ON THE STACK
023366  004737  033732                  JSR     PC,$DB2D        ;CONVERT $TRANS
023372  004537  027612                  JSR     R5,REPLZ        ;TYPE IT
023376  000012                          .WORD   10.             ;TYPE 10 DIGITS
023400  104401  053363                  TYPE    ,LINSP          ;SPACES
023404  062702  000004                  ADD     #4,R2           ;INCREMENT R2
023410  010246                          MOV     R2,-(SP)        ;PUT $READ ON THE STACK
023412  004737  033732                  JSR     PC,$DB2D        ;CONVERT $READ
023416  004537  027612                  JSR     R5,REPLZ        ;TYPE IT
023422  000012                          .WORD   10.             ;TYPE 10 DIGITS
023424  104401  053364                  TYPE    ,LINSPO         ;1 SPACE
023430  062702  000004                  ADD     #4,R2           ;INCREMENT R2
023434  062702  000002                  ADD     #2,R2           ;INCREMENT R2 AGAIN
023440  012246                          MOV     (R2)+,-(SP)     ;PUT $SOFT ON THE STACK
023442  004737  030302                  JSR     PC,$SB2D        ;CONVERT $SOFT
023446  004537  027612                  JSR     R5,REPLZ        ;TYPEOUT $SOFT
023452  000004                          .WORD   4               ;TYPE 4 DIGITS
023454  104401  053364                  TYPE    ,LINSPO         ;SPACES
023460  012246                          MOV     (R2)+,-(SP)     ;PUT $HARD ON THE STACK
023462  004737  030302                  JSR     PC,$SB2D        ;CONVERT $HARD
023466  004537  027612                  JSR     R5,REPLZ        ;TYPEOUT $HARD
023472  000004                          .WORD   4               ;TYPE 4 DIGITS
023474  104401  053364                  TYPE    ,LINSPO         ;SPACES
023500  012246                          MOV     (R2)+,-(SP)     ;PUT $SKI ON THE STACK
023502  004737  030302                  JSR     PC,$SB2D        ;CONVERT $SKI
023506  004537  027612                  JSR     R5,REPLZ        ;TYPEOUT $SKI
023512  000004                          .WORD   4               ;TYPE 4 DIGITS
```

```
     023514  104401  053364                  TYPE    ,LINSPO          ;SPACES
     023520  012246                          MOV     (R2)+,-(SP)      ;PUT $MISPO ON THE STACK
     023522  004737  030302                  JSR     PC,$SB2D         ;CONVERT $MISPO
     023526  004537  027612                  JSR     R5,REPLZ         ;TYPEOUT $MISPO
     023532  000004                          .WORD   4                ;TYPE 4 DIGITS
     023534  104401  053364                  TYPE    ,LINSPO          ;SPACES
     023540  016046  000056                  MOV     $TOTAL(R0),-(SP)  ;CALCULATE NUMBER OF OTHER ERRORS
     023544  166016  000060                  SUB     $SOFT(R0),(SP)   ;SUBTRACT $SOFT FROM $TOTAL
     023550  166016  000062                  SUB     $HARD(R0),(SP)   ;SUBTRACT $HARD FROM $TOTAL
     023554  166016  000064                  SUB     $SKI(R0),(SP)    ;SUBTRACT $SKI FROM $TOTAL
     023560  166016  000066                  SUB     $MISPO(R0),(SP)  ;SUBTRACT $MISPO FROM $TOTAL
     023564  004737  030302                  JSR     PC,$SB2D         ;CONVERT 'OTHER' COUNT
     023570  004537  027612                  JSR     R5,REPLZ         ;TYPE IT
     023574  000004                          .WORD   4                ;TYPE 4 DIGITS
     023576  104401  001165                  TYPE    ,$CRLF
     023602  012602                          MOV     (SP)+,R2         ;RESTORE R2
     023604  000207                          RTS     PC



                                     ;ROUTINE TO INCREMENT $SOFT
                                     ;
                                     ;NOTE:  $SOFT WILL NOT BE INCREMENTED BEYOND 9999 (10)

     023606  005737  001264          INCSOF: TST     BADSEC           ;SEE IF BAD TRK/SEC INDICATOR SET
     023612  001006                          BNE     1$               ;BR IF IT'S SET, DON'T INCREMENT COUNT
     023614  026027  000060  023417          CMP     $SOFT(R0),#9999.  ;IS $SOFT ALREADY AT MAXIMUM ?
     023622  103002                          BHIS    1$               ;BR IF IT IS
     023624  005260  000060                  INC     $SOFT(R0)                ;INCREMENT $SOFT
     023630  000207                  1$:     RTS     PC               ;RETURN


                                     ;ROUTINE TO INCREMENT $HARD
                                     ;
                                     ;NOTE:  $HARD WILL NOT BE INCREMENTED BEYOND 9999 (10)

     023632  005737  001264          INCHRD: TST     BADSEC           ;SEE IF BAD TRK/SEC INDICATOR SET
     023636  001006                          BNE     1$               ;BR IF IT'S SET, DON'T INCREMENT COUNT
     023640  026027  000062  023417          CMP     $HARD(R0),#9999.  ;IS $HARD ALREADY AT MAXIMUM ?
     023646  103002                          BHIS    1$               ;BR IF IT IS
     023650  005260  000062                  INC     $HARD(R0)                ;INCREMENT $HARD
     023654  000207                  1$:     RTS     PC               ;RETURN


                                     ;ROUTINE TO INCREMENT $SKI
                                     ;
                                     ;NOTE:  $SKI WILL NOT BE INCREMENTED BEYOND 9999 (10)

     023656  005737  001264          INCSKI: TST     BADSEC           ;SEE IF BAD TRK/SEC INDICATOR SET
     023662  001006                          BNE     1$               ;BR IF IT'S SET, DON'T INCREMENT COUNT
     023664  026027  000064  023417          CMP     $SKI(R0),#9999.  ;IS $SKI ALREADY AT MAXIMUM ?
     023672  103002                          BHIS    1$               ;BR IF IT IS
     023674  005260  000064                  INC     $SKI(R0)                 ;INCREMENT $SKI
     023700  000207                  1$:     RTS     PC               ;RETURN


                                     ;ROUTINE TO INCREMENT $MISPO
```

```
                                             ;
                                             ;NOTE:  $MISPO WILL NOT BE INCREMENTED BEYOND 9999 (10)

    023702  005737  001264                   INCMIS: TST     BADSEC                 ;SEE IF BAD TRK/SEC INDICATOR SET
    023706  001006                                   BNE     1$                     ;BR IF IT'S SET, DON'T INCREMENT COUNT
    023710  026027  000066  023417                   CMP     $MISPO(R0),#9999.      ;IS $MISPO ALREADY AT MAXIMUM ?
    023716  103002                                   BHIS    1$                     ;BR IF IT IS
    023720  005260  000066                           INC     $MISPO(R0)                     ;INCREMENT $MISPO
    023724  000207                           1$:     RTS     PC                     ;RETURN

                                             ;ROUTINE TO INCREMENT $TOTAL
                                             ;
                                             ;NOTE:  $TOTAL WILL NOT BE INCREMENTED BEYOND 9999 (10)

    023726  005737  001264                   INCTOT: TST     BADSEC                 ;SEE IF BAD TRK/SEC INDICATOR SET
    023732  001006                                   BNE     1$                     ;BR IF IT'S SET, DON'T INCREMENT COUNT
    023734  026027  000056  023417                   CMP     $TOTAL(R0),#9999.      ;IS $TOTAL ALREADY AT MAXIMUM ?
    023742  103002                                   BHIS    1$                     ;BR IF IT IS
    023744  005260  000056                           INC     $TOTAL(R0)                     ;INCREMENT $TOTAL
    023750  000207                           1$:     RTS     PC                     ;RETURN


                                             ;ROUTINE TO TYPE THE TIME

    023752  005737  001210                   $TIME:  TST     CLKFLG                 ;CLOCK ON THE SYSTEM ?
    023756  001033                                   BNE     1$                     ;BR IF NOT
    023760  104401  001165                           TYPE    ,$CRLF                 ;CR-LF
    023764  013746  001266                           MOV     HOUR,-(SP)             ;PUT 'HOURS' ON THE STACK
    023770  004737  030302                           JSR     PC,$SB2D               ;CONVERT TO DECIMAL
    023774  004537  027612                           JSR     R5,REPLZ               ;TYPE IT
    024000  000002                                   .WORD   2                      ;TYPE 2 DIGITS
    024002  104401  054516                           TYPE    ,COLON                 ;':'
    024006  013746  001270                           MOV     MINUTE,-(SP)           ;PUT 'MINUTES' ON THE STACK
    024012  004737  030302                           JSR     PC,$SB2D               ;CONVERT TO DECIMAL
    024016  004537  027612                           JSR     R5,REPLZ               ;TYPE IT
    024022  000002                                   .WORD   2                      ;TYPE 2 DIGITS
    024024  104401  054516                           TYPE    ,COLON                 ;':'
    024030  013746  001272                           MOV     SECOND,-(SP)           ;PUT SECONDS ON THE STACK
    024034  004737  030302                           JSR     PC,$SB2D               ;CONVERT TO DECIMAL
    024040  004537  027612                           JSR     R5,REPLZ               ;TYPE IT
    024044  000002                                   .WORD   2                      ;TYPE 2 DIGITS
    024046  000207                           1$:     RTS     PC

                                             ;CLOCK HANDLER ROUTINE

    024050  005337  001274                   CLOCK:  DEC     SIXTEE                 ;INCREMENT THE 1/60 SECOND COUNTER
    024054  001035                                   BNE     1$                     ;BR IF A SECOND NOT COUNTED
    024056  013737  001212  001274                   MOV     HZ,SIXTEE              ;RESTORE THE VALUE
    024064  005237  001272                           INC     SECOND                 ;COUNT THE SECOND
    024070  022737  000074  001272                   CMP     #60.,SECOND            ;AT MAXIMUM ?
    024076  001024                                   BNE     1$                     ;BR IF NOT
    024100  005037  001272                           CLR     SECOND                 ;CLEAR THE SECOND'S COUNTER
    024104  005237  001412                           INC     INTRVL+2               ;COUNT THE PERFORMANCE SUMMARY INTERVAL
    024110  005237  001270                           INC     MINUTE                 ;COUNT THE MINUTE
    024114  022737  000074  001270                   CMP     #60.,MINUTE            ;AT MAXIMUM ?
    024122  001012                                   BNE     1$                     ;BR IF NOT
```

```
024124  005037  001270                  CLR     MINUTE          ;CLEAR THE MINUTE'S COUNTER
024130  005237  001266                  INC     HOUR            ;COUNT THE HOURS
024134  022737  001747  001266          CMP     #999.,HOUR      ;AT MAXIMUM
024142  103002                          BHIS    1$              ;BR IF NOT
024144  005037  001266                  CLR     HOUR            ;CLEAR THE HOURS
024150  012746  000021          1$:     MOV     #17.,-(SP)      ;17 MS ON THE STACK
024154  004737  041100                  JSR     PC,RPTMR        ;DRIVER TIMER ROUTINE
024160  005737  001410                  TST     INTRVL          ;DISPLAY THE PERFORMANCE SUMMARY ?
024164  001411                          BEQ     2$              ;BR IF NOT
024166  023737  001410  001412          CMP     INTRVL,INTRVL+2 ;DISPLAY INTERVAL FINISHED ?
024174  001005                          BNE     2$              ;BR IF NOT
024176  012737  177777  001214          MOV     #-1,STATIN      ;SET PERFORMANCE SUMMARY DISPLAY FLAG
024204  005037  001412                  CLR     INTRVL+2        ;CLEAR THE PERFORMANCE INTERVAL COUNTER
024210  000002                  2$:     RTI                     ;RETURN
```

```
                                        ;COMMAND DECODE ROUTINE
                                        ;CALL:
                                        ;       MOV     #-1,CFLAG       ;'CFLAG' IS NORMALLY SET BY THE TTY SERVICE
                                        ;                               ;ROUTINE IN INTERRUPT MODE
                                        ;       JSR     PC,KSR
                                        ;       RETURN1                 ;SYSTEM BUSY RETURN
                                        ;       RETURN2                 ;RETURN AFTER KEYBOARD SERVICED

024212  005737  001440                  KSR:    TST     ORDERQ          ;ANY OPERATIONS ACTIVE ?
024216  001003                                  BNE     1$              ;BR IF SOME ARE
024220  005037  024244                          CLR     3$              ;CLEAR THE LOOP COUNTER
024224  000410                                  BR      KSR1            ;PROCESS THE KEYBOARD REQUEST
024226  062737  000001  024244          1$:     ADD     #1,3$           ;COUNT THE TIMES THROUGH THE LOOP
024234  001002                                  BNE     2$              ;BR IF NOT ENOUGH
024236  104401  054621                          TYPE    ,BUSY           ;'SYSTEM BUSY...'
024242  000207                          2$:     RTS     PC              ;PROCESS ANY COMPLETED DRIVES
024244  000000                          3$:     .WORD   0               ;LOOP COUNTER

024246  104412                          KSR1:   SAVREG                  ;SAVE THE REGISTERS
024250  012737  000200  177776                  MOV     #PR4,PS         ;SET PRIORITY TO 4
024256  005037  001262                          CLR     CFLAG           ;CLEAR THE 'CONTROL C' FLAG
024262  004737  023752                          JSR     PC,$TIME        ;TYPE THE TIME
024266  004737  030400                          JSR     PC,$TKINT       ;INITIALIZE THE KEYBOARD
024272  104401  054346                          TYPE    ,ENTCOM         ;'ENTER COMMANDS'
024276  104411                                  RDLIN                   ;READ THE KEYBOARD
024300  012605                                  MOV     (SP)+,R5        ;GET ADDRESS OF INPUT STRING
024302  005737  001262                          TST     CFLAG           ;CHECK THE CONTROL C FLAG
024306  001065                                  BNE     7$              ;EXIT IF 'CONTROL C' ENTERED
024310  005205                                  INC     R5              ;POINT TO SECOND CHARACTER
024312  122715  000101                          CMPB    #'A,(R5)        ;EQ TO AN 'A'
024316  001410                                  BEQ     1$              ;BR IF IT IS
024320  121527  000067                          CMPB    (R5),#'7        ;DRIVE NUMBER GREATER THAN AN ASCII 7 ?
024324  101054                                  BHI     6$              ;BR IF IT IS
024326  121527  000060                          CMPB    (R5),#'0        ;DRIVE NUMBER LESS THAN AN ASCII 0 ?
024332  103451                                  BLO     6$              ;BR IF IT IS
024334  142715  177770                          BICB    #^C7,(R5)       ;LEAVE ONLY LOWER 3 BITS IF CHAR NOT 'A'
024340  122765  000124  177777          1$:     CMPB    #'T,-1(R5)      ;EQ TO 'T'
024346  001003                                  BNE     2$              ;BR IF NOT EQ
024350  004737  025054                          JSR     PC,NEWASN       ;ASSIGN DRIVE FOR TEST
024354  000442                                  BR      7$              ;EXIT
024356  122765  000104  177777          2$:     CMPB    #'D,-1(R5)      ;EQ TO 'D' ?
024364  001003                                  BNE     3$              ;BR IF NOT EQ
024366  004737  025064                          JSR     PC,DEASGN       ;DEASSIGN DRIVE
024372  000433                                  BR      7$              ;EXIT
024374  122765  000123  177777          3$:     CMPB    #'S,-1(R5)      ;EQ TO 'S'
024402  001003                                  BNE     4$              ;BR IF NOT EQ
024404  004737  025172                          JSR     PC,SCMND        ;TYPE STATISTICS
024410  000424                                  BR      7$              ;EXIT
024412  122765  000127  177777          4$:     CMPB    #'W,-1(R5)      ;EQ TO 'W'
024420  001007                                  BNE     5$              ;BR IF NOT EQ
024422  032777  000001  154510                  BIT     #SW0,@SWR       ;IS SWITCH 0 SET ?
024430  001012                                  BNE     6$              ;BR IF SET, CAN'T DO 'W' COMMAND
024432  004737  025442                          JSR     PC,DATAPK       ;WRITE A DATA PACK
024436  000411                                  BR      7$              ;EXIT
024440  122765  000122  177777          5$:     CMPB    #'R,-1(R5)      ;EQ TO 'R' ?
024446  001003                                  BNE     6$              ;BR IF NOT EQ
024450  004737  025454                          JSR     PC,REDAPK       ;READ A DATA PACK
```

```
024454  000402                                  BR      7$              ;EXIT
024456  104401  054325                  6$:     TYPE    ,INVLD          ;TYPE 'INVALID COMMAND' MESSAGE
024462  104413                          7$:     RESREG                  ;RESTORE R0 - R5
024464  062716  000002                          ADD     #2,(SP)         ;INCREMENT THE RETURN ADDRESS
024470  005777  154452                          TST     @$TKB           ;CLEAR THE TTY BUFFER
024474  052777  000100  154442                  BIS     #BIT06,@$TKS    ;SET TTY INTERRUPT ENABLE
024502  005037  177776                          CLR     PS              ;SET PRIORITY BACK TO ZERO
024506  000207                                  RTS     PC              ;RETURN

                                        ;ROUTINE TO PROCESS THE ASSIGN REQUEST ('T', 'R', OR 'W' COMMANDS)

024510  122715  000101                  ASSIGN: CMPB    #'A,(R5)        ;ASSIGN ALL DRIVES?
024514  001430                                  BEQ     ASGN2           ;BR IF ALL DRIVES
024516  111504                          ASGN1:  MOVB    (R5),R4         ;PUT DRIVE # IN R4
024520  012737  053557  026750                  MOV     #UNTASN,ASNMSG  ;'DRIVE ASSIGNED' MESSAGE ADDRESS
024526  012703  000001                          MOV     #1,R3           ;RELOAD R3 FOR 1 UNIT
024532  136437  034470  001462                  BITB    ATABIT(R4),ASNLST  ;DRIVE ALREADY ASSIGNED ?
024540  001013                                  BNE     2$              ;BR IF IT IS
024542  005704                                  TST     R4              ;TRYING TO ASSIGN DRIVE 0 ?
024544  001007                                  BNE     1$              ;BR IF NOT
024546  012737  053643  026750                  MOV     #NOTAVL,ASNMSG  ;'NOT AVAILABLE' MESSAGE ADDRESS
024554  122737  000011  000041                  CMPB    #11,41          ;SEE IF LOADED FROM AN RP0/4/5/6
024562  001402                                  BEQ     2$              ;BR IF RP04/5/6 IS THE LOAD DEVICE
024564  004737  024650                  1$:     JSR     PC,ASGN3        ;SEE IF DRIVE ON THE SYSTEM
024570  000137  026730                  2$:     JMP     ASNERR          ;RETURN HERE IF DRIVE NOT AVAIL
024574  000207                                  RTS     PC              ;EXIT
024576  122737  000011  000041          ASGN2:  CMPB    #11,41          ;LOADED FROM AN RP04/5/6 ?
024604  001005                                  BNE     1$              ;BR IF NOT
024606  012704  000001                          MOV     #1,R4           ;START WITH DRIVE 1
024612  012703  000007                          MOV     #7.,R3          ;SETUP FOR ONLY 7 DRIVES
024616  000403                                  BR      2$              ;CONTINUE
024620  005004                          1$:     CLR     R4              ;START WITH DRIVE 0
024622  012703  000010                          MOV     #8.,R3          ;DRIVE COUNT FOR 8 DRIVES
024626  004737  024650                  2$:     JSR     PC,ASGN3        ;ASSIGN ALL UNASSIGNED, AVAIL DRIVES
024632  000137  024640                  3$:     JMP     4$              ;DRIVE NOT ON SYSTEM
024636  000207                                  RTS     PC              ;RETURN
024640  012746  024632                  4$:     MOV     #3$,-(SP)       ;PUT RETURN ADDRESS ON THE STACK
024644  000137  024766                          JMP     ASGN4           ;LOOK FOR MORE DRIVES
024650  136437  034470  001462          ASGN3:  BITB    ATABIT(R4),ASNLST  ;DRIVE ALREADY ASSIGNED ?
024656  001043                                  BNE     ASGN4           ;BR IF IT IS
024660  005737  034466                  1$:     TST     DTUW            ;DATA TRANSFER UNDER WAY ?
024664  100375                                  BPL     1$              ;BR IF IT IS
024666  110437  046150                          MOVB    R4,GENDPB       ;DRIVE NUMBER
024672  004737  015546                          JSR     PC,RECALO       ;RECALIBRATE DRIVE
024676  105764  034354                          TSTB    DRVSTA(R4)      ;DRIVE AVAILABLE?
024702  001444                                  BEQ     ASGN7           ;BR IF DRIVE OFFLINE OR NONEXISTENT
024704  100437                                  BMI     ASGN6           ;BR IF DRIVE UNSAFE
024706  006304                                  ASL     R4              ;MAKE R4 INTO WORD INDEX
024710  016464  001740  001506                  MOV     BLKADR(R4),NEWUNT(R4)  ;DPB ADDRESS
024716  016400  001740                          MOV     BLKADR(R4),R0   ;PUT BLOCK'S ADDR INTO R0
024722  004737  025466                          JSR     PC,CLRDPB       ;CLEAR BLOCK FOR DRIVE JUST ASSIGNED
024726  004737  025672                          JSR     PC,DRVPRM       ;GET THE DRIVE'S ADDRESS LIMITS
024732  004737  026154                          JSR     PC,GETID        ;GET DRIVE I.D.
024736  004737  026264                          JSR     PC,GETADR       ;GET BAD SECTOR ADDRESSES
024742  012760  000001  000070                  MOV     #1,$PASSC(R0)   ;PRESET PASS COUNT TO 1
024750  005737  001216                          TST     PACK            ;WRITE DATA PACK ?
024754  001403                                  BEQ     2$              ;BR IF NOT
```

```
024756  113760  001216  000026            MOVB    PACK,$PACK(R0)   ;SET READ/WRITE DATA PACK INDICATOR
024764  006204                    2$:     ASR     R4               ;RESTORE DRIVE ADDRESS
024766  005303                    ASGN4:  DEC     R3               ;DECREMENT DRIVE COUNT
024770  001402                            BEQ     ASGN5            ;BR IF FINISHED
024772  005204                            INC     R4               ;INCREMENT DRIVE NUMBER
024774  000725                            BR      ASGN3            ;CONTINUE
024776  062716  000004            ASGN5:  ADD     #4,(SP)          ;INCREMENT RETURN
025002  000207                            RTS     PC               ;RETURN
025004  012737  053662  026750    ASGN6:  MOV     #NOTSAF,ASNMSG   ;'UNSAFE' MESSAGE ADDRESS
025012  000207                            RTS     PC               ;RETURN
025014  105764  034364            ASGN7:  TSTB    DRVTYP(R4)       ;DRIVE PRESENT?
025020  001405                            BEQ     1$               ;BR IF NOT
025022  100010                            BPL     2$               ;BR IF DRIVE OFFLINE
025024  012737  053605  026750            MOV     #NOTRP,ASNMSG    ;ADDRESS OF 'NOT RP04/5/6' MSG
025032  000407                            BR      3$               ;EXIT
025034  012737  053626  026750    1$:     MOV     #NOTPRS,ASNMSG   ;ADDRESS OF 'NOT PRESENT' MSG
025042  000403                            BR      3$               ;EXIT
025044  012737  053514  026750    2$:     MOV     #UNTOFF,ASNMSG   ;ADDRESS OF 'DRIVE OFFLINE' MESSAGE
025052  000207                    3$:     RTS     PC               ;ERROR RETURN

                                  ;'T' COMMAND (ROUTINE TO ASSIGN A DRIVE)

025054  005037  001216            NEWASN: CLR     PACK             ;CLEAR 'W' COMMAND INDICATOR
025060  000137  024510                    JMP     ASSIGN           ;GO TO THE ASSIGN ROUTINE

                                  ;'D' COMMAND (ROUTINE TO DEASSIGN A DRIVE)

025064  005004                    DEASGN: CLR     R4
025066  122715  000101                    CMPB    #'A,(R5)         ;DEASSIGN ALL DRIVES ?
025072  001434                            BEQ     5$               ;BR IF YES
025074  012703  000001                    MOV     #BIT00,R3        ;SET R3 FOR ONE UNIT
025100  111504                            MOVB    (R5),R4          ;GET DRIVE NUMBER
025102  136437  034470  001462    1$:     BITB    ATABIT(R4),ASNLST  ;DRIVE ASSIGNED ?
025110  001414                            BEQ     3$               ;BR IF NOT
025112  146437  034470  001462            BICB    ATABIT(R4),ASNLST  ;DELETE THE DRIVE FROM THE ASSIGNED LIST
025120  006304                            ASL     R4               ;MAKE ADDR INTO A WORD INDEX
025122  016464  001740  001464            MOV     BLKADR(R4),DUNIT(R4)  ;PUT ADDRESS IN DEASSIGN LIST
025130  006204                            ASR     R4
025132  005303                    2$:     DEC     R3               ;ANY MORE DRIVES ?
025134  001412                            BEQ     4$               ;BR IF NOT
025136  005204                            INC     R4
025140  000760                            BR      1$
025142  122715  000101            3$:     CMPB    #'A,(R5)         ;DEASSIGN ALL DRIVES ?
025146  001771                            BEQ     2$               ;BR IF YES
025150  012737  053535  026750            MOV     #UNTNOT,ASNMSG   ;ADDR OF 'NOT ASSIGNED' MESSAGE
025156  004737  026730                    JSR     PC,ASNERR        ;REPORT IT
025162  000207                    4$:     RTS     PC
025164  012703  000010            5$:     MOV     #8.,R3           ;SET UNIT COUNT TO 8
025170  000744                            BR      1$

                                  ;'S' COMMAND (ROUTINE TO TYPE DRIVE PERFORMANCE SUMMARY)

025172  005004                    SCMND:  CLR     R4
025174  122715  000101                    CMPB    #'A,(R5)         ;ALL STATISTICS ?
025200  001421                            BEQ     2$               ;BR IF YES
025202  111504                            MOVB    (R5),R4          ;GET DRIVE NUMBER
025204  136437  034470  001462            BITB    ATABIT(R4),ASNLST  ;SEE IF DRIVE ASSIGNED
```

```
025212  001406                          BEQ     1$                      ;BR IF NOT
025214  006304                          ASL     R4                      ;MAKE DRIVE ADDR INTO WORD INDEX
025216  016400  001740                  MOV     BLKADR(R4),R0           ;ADDR OF BLOCK
025222  004737  023200                  JSR     PC,TYPEST               ;TYPE DRIVE STATISTICS
025226  000504                          BR      9$                      ;EXIT
025230  012737  053535  026750  1$:     MOV     #UNTNOT,ASNMSG          ;ADDR OF 'NOT ASSIGNED' MSG
025236  004737  026730                  JSR     PC,ASNERR               ;TYPE ERROR MESSAGE
025242  000476                          BR      9$                      ;EXIT
025244  012703  000010          2$:     MOV     #8.,R3                  ;DRIVE COUNT
025250  136437  034470  001462  3$:     BITB    ATABIT(R4),ASNLST  ;SEE IF DRIVE ASSIGNED
025256  001004                          BNE     4$                      ;BR IF YES
025260  005204                          INC     R4                      ;INCREMENT DRIVE ADDRESS
025262  005303                          DEC     R3                      ;DECREMENT COUNTER
025264  001371                          BNE     3$                      ;MORE TO CHECK
025266  000464                          BR      9$                      ;NONE ASSINGED, RETURN
025270  004737  023116          4$:     JSR     PC,STATPR               ;TYPE ALL STATISTICS
025274  105737  001220                  TSTB    DATE                    ;SEE IF 'DATE' ENTERED
025300  001404                          BEQ     11$                     ;BR IF NOT
025302  104401  054520                  TYPE    ,DATEIS                 ;'DATE: '
025306  104401  001220                  TYPE    ,DATE                   ;THE OPERATOR ENTERED DATE
025312  105737  001232          11$:    TSTB    OPERID                  ;SEE IF OPERATOR I.D. ENTERED
025316  001404                          BEQ     12$                     ;BR IF NOT
025320  104401  054530                  TYPE    ,IDIS                   ;'OPERATOR I.D.: '
025324  104401  001232                  TYPE    ,OPERID                 ;THE OPERATOR I.D.
025330  104401  054551          12$:    TYPE    ,HEDLIN                 ;HEADER LINE
025334  012737  043334  025410          MOV     #DRIVE0+$DRVID,6$  ;DRIVE I.D. FIELD ADDRESS
025342  005004                          CLR     R4                      ;DRIVE ADDRESS
025344  012703  000010                  MOV     #8.,R3                  ;COUNTER
025350  136437  034470  001462  5$:     BITB    ATABIT(R4),ASNLST  ;SEE IF DRIVE ASSIGNED
025356  001417                          BEQ     7$                      ;BR IF NOT ASSIGNED
025360  010446                          MOV     R4,-(SP)                ;;SAVE R4 FOR TYPEOUT
                                                                        ;;TYPE DRIVE NUMBER
025362  104403                          TYPOS                           ;;GO TYPE--OCTAL ASCII
025364     002                          .BYTE   2                       ;;TYPE 2 DIGIT(S)
025365     000                          .BYTE   0                       ;;SUPPRESS LEADING ZEROS
025366  104401  053361                  TYPE    ,LIN4SP                 ;4 SPACES
025372  105777  000012                  TSTB    @6$                     ;SEE IF DRIVE I.D. ENTERED
025376  001003                          BNE     10$                     ;BR IF DRIVE I.D. PRESENT
025400  104401  054572                  TYPE    ,NONE                   ;TYPE 'NONE'
025404  000404                          BR      7$                      ;CONTINUE
025406  104401                  10$:    TYPE                            ;TYPE THE DRIVE I.D.
025410  000000                  6$:     .WORD   0                       ;ADDRESS OF DRIVE I.D. FIELD HERE
025412  104401  001165                  TYPE    ,$CRLF                  ;CR-LF
025416  005303                  7$:     DEC     R3                      ;DECREMENT THE COUNTER
025420  001405                          BEQ     8$                      ;BR IF AT END
025422  062737  000304  025410          ADD     #$RPEC2+2,6$            ;INCREMENT THE MESSAGE FIELD ADDRESS
025430  005204                          INC     R4                      ;INCREMENT DRIVE ADDRESS
025432  000746                          BR      5$                      ;CONTINUE
025434  104401  001165          8$:     TYPE    ,$CRLF                  ;CR-LF
025440  000207                  9$:     RTS     PC

                                ;'W' COMMAND (ROUTINE TO WRITE A DATA PACK)

025442  012737  177777  001216  DATAPK: MOV     #-1,PACK                ;SET THE 'W' COMMAND INDICATOR
025450  000137  024510                  JMP     ASSIGN                  ;ASSIGN REQUESTED DRIVE


```

```
                                        ;'R' COMMAND (ROUTINE TO READ A DATA PACK)

025454  012737  000001  001216  REDAPK: MOV     #1,PACK         ;SET THE 'READ' INDICATOR
025462  000137  024510                  JMP     ASSIGN          ;ASSIGN THE REQUESTED DRIVE
```

```
                                            ;ROUTINE TO CLEAR THE DPB FOR THE ASSIGNED DRIVE
                                            ;CALL:
                                            ;       MOV     #DPB,R0         ;DPB ADDRESS
                                            ;       JSR     PC,CLRDPB
                                            ;       RETURN

 025466                                     CLRDPB:
 025466  010346                                     MOV     R3,-(SP)        ;;PUSH R3 ON STACK
 025470  010446                                     MOV     R4,-(SP)        ;;PUSH R4 ON STACK
 025472  010546                                     MOV     R5,-(SP)        ;;PUSH R5 ON STACK
 025474  016046  000262                             MOV     $RPDT(R0),-(SP) ;SAVE DRIVE TYPE              (REVD)
 025500  010004                                     MOV     R0,R4           ;GET THE DPB ADDRESS
 025502  062704  000002                             ADD     #2,R4           ;ADDRESS OF FIRST LOCN TO BE CLEARED
 025506  012703  000005                             MOV     #5,R3           ;NUMBER OF LOCNS TO BE CLEARED
 025512  005024                             1$:     CLR     (R4)+           ;CLEAR THE LOCATION
 025514  005303                                     DEC     R3              ;DECREMENT THE COUNTER
 025516  001375                                     BNE     1$              ;BR IF NOT FINISHED
 025520  062704  000002                             ADD     #2,R4           ;MOVE THE ADDRESS PAST THE 'REG' ADDR
 025524  012703  000070                             MOV     #$NEXT-$REG,R3  ;NUMBER OF LOCNS TO BE CLEARED
 025530  005024                             2$:     CLR     (R4)+           ;CLEAR
 025532  162703  000002                             SUB     #2,R3           ;DECREMENT THE LOCN COUNTER
 025536  001374                                     BNE     2$              ;BR IF NOT FINISHED
 025540  062704  000014                             ADD     #12.,R4         ;MOVE PAST ADDRESS LIMITS
 025544  012703  000162                             MOV     #$RPEC2-MINSEC,R3  ;NUMBER OF LOCNS TO BE CLEARED
 025550  005024                             3$:     CLR     (R4)+           ;CLEAR A LOCATION
 025552  162703  000002                             SUB     #2,R3           ;DECREMENT THE COUNTER
 025556  001374                                     BNE     3$              ;BR IF NOT DONE
 025560  113760  001434  000024                     MOVB    BEGCOD,$CODE(R0)  ;INITIAL COMMAND CODE
 025566  013701  001434                             MOV     BEGCOD,R1       ;GET THE ACTUAL OP CODE
 025572  116160  001760  000002                     MOVB    COMTBL(R1),$COMND(R0)  ;OPERATION CODE
 025600  113760  001432  000030                     MOVB    BEGPAT,$PATTC(R0)  ;PATTERN CODE
 025606  106360  000030                             ASLB    $PATTC(R0)      ;CONVERT CODE TO A TABLE INDEX
 025612  013760  001436  000020                     MOV     BEGSIZ,$WRDL(R0)  ;BEGINNING RECORD SIZE
 025620  013760  001436  000004                     MOV     BEGSIZ,$WRDM(R0)  ;VALUE FOR DATA TRANSFER
 025626  005460  000004                             NEG     $WRDM(R0)       ;MAKE IT INTO 2'S COMPLEMENT
 025632  012760  000400  000022                     MOV     #256.,$SSEC(R0)  ;INITIAL VALUE OF SECTOR SIZE
 025640  132760  000001  000024                     BITB    #1,$CODE(R0)    ;HEADER ORDER ?
 025646  001403                                     BEQ     4$              ;BR IF NOT
 025650  062760  000004  000022                     ADD     #4,$SSEC(R0)    ;ADD HEADER SIZE TO SECTOR SIZE
 025656  012660  000262                     4$:     MOV     (SP)+,$RPDT(R0) ;RESTORE DRIVE TYPE           (REVD)
 025662  012605                                     MOV     (SP)+,R5        ;;POP STACK INTO R5
 025664  012604                                     MOV     (SP)+,R4        ;;POP STACK INTO R4
 025666  012603                                     MOV     (SP)+,R3        ;;POP STACK INTO R3
 025670  000207                                     RTS     PC              ;RETURN


                                            ;ROUTINE TO GET ADDRESS LIMITS FROM THE OPERATOR

 025672  010346                             DRVPRM: MOV     R3,-(SP)        ;SAVE R3
 025674  010446                                     MOV     R4,-(SP)        ;SAVE R4
 025676  005737  000042                             TST     42              ;RUNNING UNDER MONITOR CONTROL
 025702  001035                                     BNE     3$              ;BR IF YES
 025704  104401  054415                             TYPE    ,ENTLMT         ;'ENTER ADDRESSES'
 025710  006204                                     ASR     R4              ;CONVERT INDEX TO DRIVE NUMBER
 025712  010446                                     MOV     R4,-(SP)        ;;SAVE R4 FOR TYPEOUT
                                                                            ;;TYPE DRIVE NUMBER
 025714  104403                                     TYPOS                   ;;GO TYPE--OCTAL ASCII
 025716     002                                     .BYTE   2               ;;TYPE 2 DIGIT(S)
```

```
025717     000                                      .BYTE   0                    ;;SUPPRESS LEADING ZEROS
025720     104401  055074                           TYPE    ,SLASH               ;'/'
025724     012737  053711  025770                   MOV     #RP04B,2$            ;ADDRESS OF 'RP04' MESSAGE
025732     132764  000001  034364                   BITB    #BIT00,DRVTYP(R4)    ;RP04 ?
025740     001012                                   BNE     1$                   ;BR IF YES
025742     012737  053716  025770                   MOV     #RP05,2$             ;ADDRESS OF 'RP05' MESSAGE
025750     132764  000002  034364                   BITB    #BIT01,DRVTYP(R4)    ;RP05 ?
025756     001003                                   BNE     1$                   ;BR IF YES
025760     012737  053723  025770                   MOV     #RP06,2$             ;ADDRESS OF 'RP06' MESSAGE
025766     104401                          1$:      TYPE                         ;TYPE THE MESSAGE WHICH FOLLOWS
025770     000000                          2$:      .WORD   0                    ;MESSAGE ADDRESS
025772     104401  001165                           TYPE    ,$CRLF               ;CR-LF
025776     012737  000632  001350          3$:      MOV     #410.,CYLIMT         ;ASSUME AN RP04/5
026004     132764  000004  034364                   BITB    #BIT02,DRVTYP(R4)    ;SEE IF RP06
026012     001403                                   BEQ     4$                   ;BR IF NOT
026014     012737  001456  001350                   MOV     #814.,CYLIMT         ;CHANGE LIMIT TO 814
026022     062760  177777  000122          4$:      ADD     #-1,$FIRST(R0)       ;SEE IF FIRST TIME STARTED
026030     103417                                   BCS     5$                   ;BR IF NOT
026032     013760  001350  000106                   MOV     CYLIMT,MAXCYL(R0)    ;LOAD MAXIMUM CYLINDER
026040     005060  000110                           CLR     MINCYL(R0)           ;CLEAR MINIMUM CYLINDER
026044     013760  001346  000112                   MOV     TRKLMT,MAXTRK(R0)    ;LOAD MAXIMUM TRACK
026052     005060  000114                           CLR     MINTRK(R0)           ;CLEAR MINIMUM TRACK
026056     013760  001344  000116                   MOV     SECLMT,MAXSEC(R0)    ;LOAD MAXIMUM SECTOR
026064     005060  000120                           CLR     MINSEC(R0)           ;CLEAR MINIMUM SECTOR
026070     006304                          5$:      ASL     R4                   ;SETUP TO ADDRESS WORDS
026072     016403  055254                           MOV     TABLE(R4),R3         ;PARAMETER TABLE ADDRESS
026076     013763  001350  000002                   MOV     CYLIMT,2(R3)         ;LOAD CYLINDER LIMIT FOR LAST CYLINDER
026104     013763  001350  000010                   MOV     CYLIMT,10(R3)        ;LOAD CYLINDER LIMIT FOR STARTING CYLINDER
026112     005737  000042                           TST     42                   ;UNDER MONITOR CONTROL ?
026116     001002                                   BNE     6$                   ;BR IF YES
026120     004737  026604                           JSR     PC,PARENT            ;GET THE DRIVE'S PARAMETERS
026124     116060  000120  000010          6$:      MOVB    MINSEC(R0),$SEC(R0)  ;INITIAL SECTOR VALUE
026132     116060  000114  000011                   MOVB    MINTRK(R0),$TRK(R0)  ;INITIAL TRACK VALUE
026140     016060  000110  000012                   MOV     MINCYL(R0),$CYL(R0)  ;INITIAL CYLINDER VALUE
026146     012604                                   MOV     (SP)+,R4             ;RESTORE R4
026150     012603                                   MOV     (SP)+,R3             ;RESTORE R3
026152     000207                                   RTS     PC                   ;RETURN

                                           ;ROUTINE TO GET THE DRIVE I.D. FROM THE OPERATOR

026154     010546                          GETID:   MOV     R5,-(SP)             ;SAVE R5
026156     005737  000042                           TST     42                   ;UNDER MONITOR CONTROL ?
026162     001036                                   BNE     2$                   ;BR IF NOT
026164     005037  001262                  1$:      CLR     CFLAG                ;CLEAR THE 'CONTROL C' FLAG
026170     104401  054370                           TYPE    ,ENTDRV              ;'ENTER DRV I.D.: '
026174     005046                                   CLR     -(SP)                ;CLEAR THE STACK
026176     111016                                   MOVB    (R0),(SP)            ;PUT THE DRIVE NUMBER ON THE STACK
026200     104403                                   TYPOS                        ;TYPE THE DRIVE NUMBER
026202     002                                      .BYTE   2                    ;TYPE 2 DIGITS
026203     000                                      .BYTE   0                    ;SUPRESS LEADING ZEROS
026204     104401  001165                           TYPE    ,$CRLF               ;CR-LF
026210     104411                                   RDLIN                        ;READ THE ENTRY
026212     012605                                   MOV     (SP)+,R5             ;GET THE ENTRY ADDRESS
026214     005737  001262                           TST     CFLAG                ;'CONTROL C' ENTERED ?
026220     001361                                   BNE     1$                   ;BR IF IT WAS
026222     121527  000056                           CMPB    (R5),#'.             ;PERIOD ENTERED ?
026226     001414                                   BEQ     2$                   ;BR IF YES
```

```
026230  112560  000224                  MOVB    (R5)+,$DRVID(R0)   ;STORE THE I.D.
026234  112560  000225                  MOVB    (R5)+,$DRVID+1(R0)  ;STORE THE I.D.
026240  112560  000226                  MOVB    (R5)+,$DRVID+2(R0)  ;STORE THE I.D.
026244  112560  000227                  MOVB    (R5)+,$DRVID+3(R0)  ;STORE THE I.D.
026250  112560  000230                  MOVB    (R5)+,$DRVID+4(R0)  ;STORE THE I.D.
026254  112560  000231                  MOVB    (R5)+,$DRVID+5(R0)  ;STORE THE I.D.
026260  012605                  2$:     MOV     (SP)+,R5            ;RESTORE R5
026262  000207                          RTS     PC                  ;RETURN

                                ;ROUTINE TO GET THE ADDRESSES OF ANY BAD SECTORS (UP TO A MAX OF 16)

026264                          GETADR:
026264  010146                          MOV     R1,-(SP)            ;;PUSH R1 ON STACK
026266  010246                          MOV     R2,-(SP)            ;;PUSH R2 ON STACK
026270  010346                          MOV     R3,-(SP)            ;;PUSH R3 ON STACK
026272  010446                          MOV     R4,-(SP)            ;;PUSH R4 ON STACK
026274  005737  000042                  TST     42                  ;UNDER MONITOR CONTROL ?
026300  001012                          BNE     15$                 ;BR IF YES
026302  005037  001262          14$:    CLR     CFLAG               ;CLEAR 'CONTROL C' FLAG
026306  104401  054455                  TYPE    ,ENTADR             ;ENTER SECTOR ADDRESSES
026312  005046                          CLR     -(SP)               ;CLEAR THE STACK
026314  111016                          MOVB    (R0),(SP)           ;PUT THE DRIVE NUMBER ON THE STACK
026316  104403                          TYPOS                       ;TYPE THE DRIVE NUMBER
026320     002                          .BYTE   2                   ;TYPE 2 DIGITS
026321     000                          .BYTE   0                   ;SUPRESS LEADING ZEROS
026322  104401  001165                  TYPE    ,$CRLF              ;CR-LF
026326  012703  000040          15$:    MOV     #32.,R3             ;NUMBER OF LOCATIONS IN THE TABLE TO PRESET
026332  012704  000124                  MOV     #$BDSEC,R4          ;TABLE INCREMENT
026336  060004                          ADD     R0,R4               ;BLOCK STARTING ADDRESS
026340  012724  177777          1$:     MOV     #-1,(R4)+           ;SET LOCATION TO 1'S
026344  005303                          DEC     R3                  ;DECREMENT TABLE SIZE COUNT
026346  001374                          BNE     1$                  ;BR IF NOT FINISHED WITH TABLE
026350  005737  000042                  TST     42                  ;UNDER MONITOR CO TROL ?
026354  001106                          BNE     13$                 ;BR IF YES
026356  162704  000100                  SUB     #64.,R4             ;SET POINTER TO BEGINNING OF TABLE
026362  012703  000020                  MOV     #16.,R3             ;NUMBER OF ADDRESSES IN TABLE
026366  104411                  2$:     RDLIN                       ;GET ADDRESS FROM OPERATOR
026370  012601                          MOV     (SP)+,R1            ;TEXT POINTER
026372  005737  001262                  TST     CFLAG               ;'CONTROL C' ENTERED ?
026376  001341                          BNE     14$                 ;BR IF IT WAS
026400  013702  001350                  MOV     CYLIMT,R2           ;UPPER LIMIT OF INPUT
026404  004537  030144                  JSR     R5,CK.DIG           ;CHECK THE DIGIT(S)
026410  026552                          12$                         ;CARRIAGE RETURN ONLY ENTERED
026412  026572                          13$                         ;PERIOD ONLY ENTERED
026414  026552                          12$                         ;ILLEGAL INPUT
026416  026430                          4$                          ;TERMINATED WITH A CARRIAGE RETURN
026420  026434                          5$                          ;TERMINATED WITH A ","
026422  026424                          3$                          ;TERMINATED WITH A "."
026424  010214                  3$:     MOV     R2,(R4)             ;CYLINDER ADDRESS
026426  000461                          BR      13$                 ;EXIT, PERIOD ENTERED
026430  010214                  4$:     MOV     R2,(R4)             ;CYLINDER ADDRESS
026432  000442                          BR      11$                 ;FINISHED WITH THIS ADDRESS, 'CR' ENTERED
026434  010214                  5$:     MOV     R2,(R4)             ;CYLINDER ADDRESS
026436  013702  001346                  MOV     TRKLMT,R2           ;UPPER LIMIT OF INPUT
026442  004537  030144                  JSR     R5,CK.DIG           ;CHECK THE DIGIT(S)
026446  026552                          12$                         ;CARRIAGE RETURN ONLY ENTERED
026450  026572                          13$                         ;PERIOD ONLY ENTERED
```

```
026452  026552                          12$                             ;ILLEGAL INPUT
026454  026470                          7$                              ;TERMINATED WITH A CARRIAGE RETURN
026456  026476                          8$                              ;TERMINATED WITH A ","
026460  026462                          6$                              ;TERMINATED WITH A "."
026462  110264  000003          6$:     MOVB    R2,3(R4)                ;TRACK ADDRESS
026466  000441                          BR      13$                     ;EXIT, ENTRY TERMINATED BY PERIOD
026470  110264  000003          7$:     MOVB    R2,3(R4)                ;TRACK ADDRESS
026474  000421                          BR      11$                     ;FINISHED WITH THIS ADDRESS, 'CR' ENTERED
026476  110264  000003          8$:     MOVB    R2,3(R4)                ;TRACK ADDRES;
026502  013702  001344                  MOV     SECLMT,R2               ;UPPER LIMIT OF INPUT
026506  004537  030144                  JSR     R5,CK.DIG               ;CHECK THE DIGIT(S)
026512  026552                          12$                             ;CARRIAGE RETURN ONLY ENTERED
026514  026572                          13$                             ;PERIOD ONLY ENTERED
026516  026552                          12$                             ;ILLEGAL INPUT
026520  026534                          10$                             ;TERMINATED WITH A CARRIAGE RETURN
026522  026552                          12$                             ;TERMINATED WITH A ","
026524  026526                          9$                              ;TERMINATED WITH A "."
026526  110264  000002          9$:     MOVB    R2,2(R4)                ;SECTOR ADDRESS
026532  000417                          BR      13$                     ;EXIT, ENTRY TERMINATED BY PERIOD
026534  110264  000002          10$:    MOVB    R2,2(R4)                ;SECTOR ADDRESS
026540  005303                  11$:    DEC     R3                      ;MORE ENTRIES ?
026542  001413                          BEQ     13$                     ;BR IF NOT
026544  062704  000004                  ADD     #4,R4                   ;INCREMENT THE TABLE POINTER
026550  000706                          BR      2$                      ;CONTINUE
026552  012714  177777          12$:    MOV     #-1,(R4)                ;CLEAR PRESENT TABLE ENTRY
026556  012764  177777  000002          MOV     #-1,2(R4)               ;CLEAR PRESENT TABLE ENTRY
026564  104401  054600                  TYPE    ,BADENT                 ;'INVALID ENTRY'
026570  000676                          BR      2$                      ;TRY AGAIN
026572                          13$:
026572  012604                          MOV     (SP)+,R4                ;;POP STACK INTO R4
026574  012603                          MOV     (SP)+,R3                ;;POP STACK INTO R3
026576  012602                          MOV     (SP)+,R2                ;;POP STACK INTO R2
026600  012601                          MOV     (SP)+,R1                ;;POP STACK INTO R1
026602  000207                          RTS     PC                      ;RETURN

                                ;PARAMETER ENTRY ROUTINE
                                ;CALL
                                ;       MOV     #ADR,R3                 ;PARAMETER TABLE ADDRESS
                                ;       JSR     PC,PARENT               ;GET THE PARAMETERS

026604  010346                  PARENT: MOV     R3,-(SP)                ;SAVE THE PARAMETER TABLE ADDRESS
026606  005037  001262                  CLR     CFLAG                   ;CLEAR THE 'CONTROL C' FLAG
026612  012337  026622          1$:     MOV     (R3)+,3$                ;ADDRESS OF PARAMETER NAME
026616  001442                          BEQ     9$                      ;BR IF AT END OF TABLE
026620  104401                          TYPE                            ;TYPE THE PARAMETER NAME
026622  000000                  3$:     .WORD   0                       ;ADDRESS OF PARAMETER NAME TEXT
026624  012302                          MOV     (R3)+,R2                ;MAXIMUM PARAMETER VALUE
026626  012305                          MOV     (R3)+,R5                ;ADDRESS OF PARAMETER
026630  011546                          MOV     (R5),-(SP)              ;CURRENT VALUE OF PARAMETER
026632  104405                          TYPDS                           ;TYPE THE CURRENT VALUE OF THE PARAMETER
026634  104401  055074                  TYPE    ,SLASH                  ;' / '
026640  104411                          RDLIN                           ;READ THE KEYBOARD
026642  012601                          MOV     (SP)+,R1                ;INPUT ASCII STRING ADDRESS
026644  005737  001262                  TST     CFLAG                   ;'CONTROL C' ENTERED ?
026650  001021                          BNE     8$                      ;BR IF IT WAS
026652  004537  030144                  JSR     R5,CK.DIG               ;CHECK THE DIGIT(S)
026656  026612                          1$                              ;CARRIAGE RETURN ONLY ENTERED
```

```
    026660  026724                          9$                      ;PERIOD ONLY ENTERED
    026662  026676                          6$                      ;ILLEGAL INPUT
    026664  026672                          5$                      ;TERMINATED WITH A CARRIAGE RETURN
    026666  026676                          6$                      ;TERMINATED WITH A ","
    026670  026710                          7$                      ;TERMINATED WITH A "."
    026672  010215                  5$:     MOV     R2,(R5)         ;MOVE NEW VALUE TO PARAMETER LOCATION
    026674  000746                          BR      1$              ;GET MORE PARAMETERS
    026676  104401  054600          6$:     TYPE    ,BADENT         ;'BAD ENTRY'
    026702  162703  000006                  SUB     #6,R3           ;DECREMENT THE TABLE POINTER
    026706  000741                          BR      1$              ;TRY AGAIN
    026710  010215                  7$:     MOV     R2,(R5)         ;NEW VALUE
    026712  000404                          BR      9$              ;EXIT
    026714  005037  001262          8$:     CLR     CFLAG           ;CLEAR THE 'CONTROL C' FLAG
    026720  011603                          MOV     (SP),R3         ;RELOAD THE PARAMETER TABLE ADDRESS
    026722  000733                          BR      1$              ;TRY AGAIN
    026724  005726                  9$:     TST     (SP)+           ;CORRECT THE STACK POINTER
    026726  000207                          RTS     PC              ;RETURN

                                        ;TYPEOUT ASSIGN/DEASSIGN ERROR MESSAGE
                                        ;CALL
                                        ;       MOV     #MESADR,ASNMSG  ;ERROR MESSAGE ADDRESS
                                        ;       JSR     PC,ASNERR
                                        ;       RETURN

    026730  104401  054323          ASNERR: TYPE    ,QUES           ;QUESTION MARK
    026734  104401  053506                  TYPE    ,UNTMSG         ;TYPE 'DRIVE'
    026740  010446                          MOV     R4,-(SP)        ;;SAVE R4 FOR TYPEOUT
                                                                    ;;TYPE DRIVE NUMBER
    026742  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
    026744     002                          .BYTE   2               ;;TYPE 2 DIGIT(S)
    026745     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
    026746  104401                          TYPE                    ;TYPE SPECIFIC MESSAGE
    026750  000000                  ASNMSG: .WORD   0               ;MESSAGE ADDRESS
    026752  104401  001165                  TYPE    ,$CRLF
    026756  000207                          RTS     PC

                                        ;DEASSIGN DRIVE IF A FATAL ERROR OCCURS
                                        ;CALL
                                        ;       JSR     PC,DROP
                                        ;       RETURN

    026760  005004                  DROP:   CLR     R4              ;CLEAR R4 FOR DRIVE NUMBER
    026762  111004                          MOVB    (R0),R4         ;MOVE DRIVE NUMBER TO R4
    026764  146437  034470  001462          BICB    ATABIT(R4),ASNLST  ;REMOVE DRIVE FROM ASSIGNED LIST
    026772  006304                          ASL     R4              ;MAKE DRIVE NUMBER INTO A TABLE INDEX
    026774  010064  001464                  MOV     R0,DUNIT(R4)    ;PUT DRIVE IN DROP LIST
    027000  104401  001165                  TYPE    ,$CRLF
    027004  104401  001165                  TYPE    ,$CRLF
    027010  104401  054104                  TYPE    ,DROPNG         ;TYPE 'DROPPING DRIVE'
    027014  104401  054213                  TYPE    ,DRNUM          ;'DRIVE #'
    027020  006204                          ASR     R4              ;DRIVE NUMBER
    027022  010446                          MOV     R4,-(SP)        ;;SAVE R4 FOR TYPEOUT
                                                                    ;;TYPE DRIVE NUMBER
    027024  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
    027026     002                          .BYTE   2               ;;TYPE 2 DIGIT(S)
    027027     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
    027030  104401  001165                  TYPE    ,$CRLF
```

```
027034  000207                              RTS     PC

                                    ;ROUTINE TO DEASSIGN DRIVE IF ERRORS BECOMES EXCESSIVE

027036  032777  000020  152074  ABNRML: BIT     #SW04,@SWR              ;SEE IF SWITCH 4 SET
027044  001006                          BNE     1$                      ;BR IF IT'S SET
027046  023760  001406  000056          CMP     MAXER,$TOTAL(R0)  ;CHECK TOTAL ERROR VALUE
027054  103002                          BHIS    1$                      ;BR IF ERRORS DONOT EXCEED MAX
027056  000137  026760                  JMP     DROP                    ;DEASSING THE DRIVE
027062  000207                  1$:     RTS     PC                      ;RETURN
```

```
                                                 .SBTTL  END OF PASS
                                                 ;ROUTINE TO CHECK FOR END OF PASS AND END OF TEST

027064  005737  001430                   EOP:     TST     ENDET                   ;END OF PASS DETERMINED BY SEEKS OR WORDS ?
027070  001412                                    BEQ     EOP1                    ;BR IF SEEKS
027072  026037  000054  001374                    CMP     $READ+2(R0),ENDCON+2    ;CHECK MSW OF WORDS READ COUNT
027100  101017                                    BHI     EOP2                    ;BR IF MSW GREATER THAN LIMIT
027102  103405                                    BLO     EOP1                    ;BR IF MSW LESS THAN LIMIT
027104  026037  000052  001372                    CMP     $READ(R0),ENDCON        ;CHECK LSW AGAINST LIMIT
027112  103012                                    BHIS    EOP2                    ;BR IF EQUAL OR GREATER
027114  000510                                    BR      EOPX                    ;EXIT

027116  026037  000044  001400           EOP1:    CMP     $POSIT+2(R0),ENDSEK+2   ;CHECK MSW OF SEEK COUNT
027124  101005                                    BHI     EOP2                    ;BR IF MSW GREATER THAN LIMIT
027126  103503                                    BLO     EOPX                    ;EXIT IF MSW LESS THAN LIMIT
027130  026037  000042  001376                    CMP     $POSIT(R0),ENDSEK       ;CHECK LSW OF SEEK COUNT
027136  103477                                    BLO     EOPX                    ;EXIT IF LSW LESS THAN LIMIT
027140  104401  001165                   EOP2:    TYPE    ,$CRLF                  ;CR-LF
027144  104401  054140                            TYPE    ,ENDPAS                 ;END OF PASS FOR THE DRIVE
027150  016046  000070                            MOV     $PASSC(R0),-(SP)        ;PUT PASS COUNT ON THE STACK
027154  104405                                    TYPDS                           ;CONVERT PASS COUNT TO DECIMAL AND TYPE IT
027156  111037  001246                            MOVB    (R0),UNIT               ;STORE THE DRIVE NUMBER
027162  032777  000020  151750                    BIT     #SW04,@SWR              ;SWITCH 4 SET ?
027170  001017                                    BNE     1$                      ;BR IF SET
027172  026037  000070  001402                    CMP     $PASSC(R0),PASCNT       ;SEE IF AT END OF TEST
027200  103413                                    BLO     1$                      ;BR IF NOT
027202  104401  054154                            TYPE    ,ENDTST                 ;TYPE 'END OF TEST'
027206  104401  054213                            TYPE    ,DRNUM                  ;'DRIVE #'
027212  013746  001246                            MOV     UNIT,-(SP)              ;;SAVE UNIT FOR TYPEOUT
                                                                                  ;;TYPE DRIVE NUMBER
027216  104403                                    TYPOS                           ;;GO TYPE--OCTAL ASCII
027220     002                                    .BYTE   2                       ;;TYPE 2 DIGIT(S)
027221     000                                    .BYTE   0                       ;;SUPPRESS LEADING ZEROS
027222  104401  001165                            TYPE    ,$CRLF                  ;CR-LF
027226  000431                                    BR      3$                      ;DEASSIGN THE DRIVE
027230  104401  054213                   1$:      TYPE    ,DRNUM                  ;'DRIVE #'
027234  013746  001246                            MOV     UNIT,-(SP)              ;;SAVE UNIT FOR TYPEOUT
                                                                                  ;;TYPE DRIVE NUMBER
027240  104403                                    TYPOS                           ;;GO TYPE--OCTAL ASCII
027242     002                                    .BYTE   2                       ;;TYPE 2 DIGIT(S)
027243     000                                    .BYTE   0                       ;;SUPPRESS LEADING ZEROS
027244  104401  001165                            TYPE    ,$CRLF                  ;CR-LF
027250  004737  023200                            JSR     PC,TYPEST               ;TYPE THE DRIVE'S STATISTICS
027254  010346                                    MOV     R3,-(SP)                ;SAVE R3
027256  010446                                    MOV     R4,-(SP)                ;SAVE R4
027260  010004                                    MOV     R0,R4                   ;DRIVE'S BLOCK ADDRESS
027262  062704  000036                            ADD     #$OPERC,R4              ;ADD THE STARTING ADDR OF SECTIONS TO CLEAR
027266  012703  000010                            MOV     #8.,R3                  ;NUMBER OF LOCNS TO BE CLEARED
                                                                                  ;(ERROR COUNTERS NOT CLEARED)
027272  005024                           2$:      CLR     (R4)+                   ;CLEAR THE LOCN
027274  005303                                    DEC     R3                      ;DECREMENT THE LOCATION COUNTER
027276  001375                                    BNE     2$                      ;BR IF MORE TO GO
027300  012604                                    MOV     (SP)+,R4                ;RESTORE R4
027302  012603                                    MOV     (SP)+,R3                ;RESTORE R3
027304  005260  000070                            INC     $PASSC(R0)              ;INCREMENT THE PASS COUNT
027310  000412                                    BR      EOPX                    ;EXIT
027312  104401  001165                   3$:      TYPE    ,$CRLF
```

```
     50 027316  005004                           CLR     R4                      ;CLEAR R4 FOR DRIVE NUMBER
     51 027320  111004                           MOVB    (R0),R4                 ;MOVE DRIVE NUMBER
     52 027322  146437  034470  001462           BICB    ATABIT(R4),ASNLST  ;DELETE DRIVE FROM ASSIGNED LIST
     53 027330  006304                           ASL     R4                      ;MAKE DRIVE NUMBER INTO TABLE INDEX
     54 027332  010064  001464                   MOV     R0,DUNIT(R4)            ;PUT BLOCK ADDRESS INTO DROP LIST
     55 027336  000207                   EOPX:   RTS     PC                      ;RETURN
```

```
                                ;ROUTINE TO GET THE REMAINDER OF THE RANDOM NUMBER
                                ;CALL
                                ;        MOV     NUMBER,R5       ;DIVISOR INTO R5
                                ;        JSR     PC,GETREM
                                ;        RETURN                  ;REMAINDER IS IN R5

027340  013746  033634          GETREM: MOV     $LONUM,-(SP)    ;STORE RANDOM NUMBER ON THE STACK FOR DIVIDE
027344  013746  033632                  MOV     $HINUM,-(SP)    ;UPPER PART
027350  010546                          MOV     R5,-(SP)        ;PUT THE DIVISOR ONTO THE STACK
027352  004737  027366                  JSR     PC,LINKDV       ;DIVIDE THE RANDOM NUMBERS
027356  012605                          MOV     (SP)+,R5        ;PUT THE REMAINDER INTO R5
027360  005726                          TST     (SP)+           ;ADJUST THE STACK POINTER
027362  000240                          NOP                     ;FOR DEBUGGING HALT
027364  000207                          RTS     PC

                                ;LINK ROUTINE TO THE DIVISION UTILITY SUBROUTINE
                                ;        THIS ROUTINE ALLOWS THE 'SYSMAC' DIVIDE ROUTINE
                                ;        CALLING SEQUENCE TO BE USED

027366  104412                  LINKDV: SAVREG                  ;STORE R0 - R5
027370  016605  000026                  MOV     26(SP),R5       ;DIVISOR
027374  005004                          CLR     R4              ;OTHER DIVISOR WORD
027376  016602  000030                  MOV     30(SP),R2       ;UPPER DIVIDEND WORD
027402  016603  000032                  MOV     32(SP),R3       ;LOWER DIVIDEND WORD
027406  005000                          CLR     R0              ;CLEAR OTHER DIVIDEND REGISTERS
027410  005001                          CLR     R1
027412  004737  027434                  JSR     PC,M.DPID       ;GO TO THE DIVIDE ROUTINE
027416  010166  000030                  MOV     R1,30(SP)       ;REMAINDER ON THE STACK
027422  010366  000032                  MOV     R3,32(SP)       ;QUOTIENT ON THE STACK
027426  104413                          RESREG                  ;RESTORE R0 - R5
027430  012616                          MOV     (SP)+,(SP)      ;MOVE RETURN UP THE STACK
027432  000207                          RTS     PC

                                ;        DIVISION UTILITY SUBROUTINE
                                ;        R0-R1-R2-R3=DIVIDEND
                                ;        R4-R5=DIVISOR
                                ;        R0-R1=REMAINDER AFTER DIVISION
                                ;        R2-R3=QUOTIENT AFTER DIVISION
                                ;        ENTER WITH JSR  PC,M.DPID
                                ;
027434  012746  000040          M.DPID: MOV     #40,-(SP)       ;COUNTER FOR DIVISION CYCLES
027440  010446                          MOV     R4,-(SP)        ;HIGH ORDER
027442  010546                          MOV     R5,-(SP)        ;LOW ORDER DIVISOR TO THE STACK
027444  005466  000002                  NEG     2(SP)           ;FORM NEGATIVE
027450  005416                          NEG     @SP             ;VERSION OF THE DIVISOR
027452  005666  000002                  SBC     2(SP)
027456  061601                          ADD     @SP,R1
027460  005500                          ADC     R0              ;PERFORM THE INITIAL SUBTRACTION
027462  066600  000002                  ADD     2(SP),R0
027466  103445                          BCS     M.DP50          ;IF CARRY THEN OVERFLOW HAS OCCURRED
027470  005046                          CLR     -(SP)           ;THIS IS A LONGER LASTING CARRY BIT
027472  006103                  M.DP40: ROL     R3
027474  006102                          ROL     R2
027476  006101                          ROL     R1
027500  006100                          ROL     R0
027502  005716                          TST     @SP             ;TEST "CARRY" INDICATOR
027504  001410                          BEQ     M.DP41          ;IF NO "CARRY" THEN ADD ELSE SUBTRACT
```

```
027506  005016                                  CLR     @SP                     ;CLEAR UP FOR NEXT TIME
027510  066601  000002                          ADD     2(SP),R1
027514  005500                                  ADC     R0                      ;ADD -(DIVISOR)
027516  005516                                  ADC     @SP     ;  I            ;SET "CARRY"
027520  066600  000004                          ADD     4(SP),R0;<-
027524  000404                                  BR      M.DP42
027526  060501                          M.DP41: ADD     R5,R1
027530  005500                                  ADC     R0                      ;ADD +(DIVISOR)
027532  005516                                  ADC     @SP     ;  I            ;SET "CARRY"
027534  060400                                  ADD     R4,R0   ;<-
027536  005516                          M.DP42: ADC     @SP                     ;SET "CARRY"
027540  005716                                  TST     @SP                     ;TEST THE UPDATE INDICATOR
027542  001401                                  BEQ     .+4     ;->             ;IF ZERO FORGET IT
027544  005203                                  INC     R3      ;  I            ;NO CARRY POSSIBLE HERE
027546  005366  000006                          DEC     6(SP)   ;<-             ;DECREMENT COUNTER
027552  003347                                  BGT     M.DP40                  ;BRANCH IF MORE TC DO
027554  006003                                  ROR     R3
027556  103404                                  BCS     M.DP44
027560  060501                                  ADD     R5,R1
027562  005500                                  ADC     R0
027564  060400                                  ADD     R4,R0
027566  000241                                  CLC
027570  006103                          M.DP44: ROL     R3
027572  062706  000010                          ADD     #10,SP                  ;ADJUST STACK BY 4 WORDS
027576  000242                                  CLV
027600  000207                                  RTS     PC
027602  062706  000006                  M.DP50: ADD     #6,SP
027606  000262                                  SEV
027610  000207                                  RTS     PC


                                        ;ROUTINE TO REPLACE LEADING ZEROS IN A NUMERIC STRING WITH SPACES
                                        ;CALL
                                        ;       MOV     #ADR,-(SP)              ;ADDRESS OF NUMBER (IN ASCII)
                                        ;       JSR     R5,REPLZ
                                        ;       .WORD   N                       ;'N' IS NUMBER OF DIGITS TO BE TYPED

027612  010046                          REPLZ:  MOV     R0,-(SP)                ;SAVE R0
027614  012746  000012                          MOV     #10.,-(SP)              ;MAXIMUM NUMBER OF DIGITS TO BE TYPED
027620  162516                                  SUB     (R5)+,(SP)              ;SUBTACT DIGITS TO FORM INDEX
027622  016600  000006                          MOV     6(SP),R0                ;ADDRESS OF NUMBER TO R0
027626  122710  000060                  1$:     CMPB    #'0,(R0)                ;BYTE EQUAL TO ASCII '0' ?
027632  001004                                  BNE     2$                      ;BR IF NOT
027634  112710  000040                          MOVB    #40,(R0)                ;REPLACE THE ZERO WITH A SPACE
027640  005200                                  INC     R0                      ;INCREMENT THE BYTE ADDRESS
027642  000771                                  BR      1$                      ;GO BACK AND LOOK FOR MORE LEADING ZEROS
027644  105710                          2$:     TSTB    (R0)                    ;SEE IF ZERO BYTE TERMINATOR
027646  001003                                  BNE     3$                      ;BR IF NOT
027650  005300                                  DEC     R0                      ;BACKUP STRING POINTER
027652  112710  000060                          MOVB    #'0,(R0)                ;PUT A ZERO BACK IN
027656  016637  000006  027672          3$:     MOV     6(SP),4$                ;PUT ADDRESS IN LOCATION FOR TYPEOUT
027664  062637  027672                          ADD     (SP)+,4$                ;BEGINNING OF SIGNIFICANT DIGITS
027670  104401                                  TYPE                            ;TYPE THE NUMBER
027672  000000                          4$:     .WORD   0                       ;ADDRESS OF NUMBER
027674  012600                                  MOV     (SP)+,R0                ;RESTORE R0
027676  012616                                  MOV     (SP)+,(SP)              ;MOVE RETURN ADDRESS
027700  000205                                  RTS     R5                      ;RETURN
```

```

                                        ,TYPE NUMERICAL ASCIZ STRING SUPRESS LEADING ZEROS

                                        ;CALL
                                        ;       MOV     #NUMADR,-(SP)   ;FIRST ADDRESS OF ASCIZ STRING
                                        ;       JSR     PC,$SUPRS

027702  010046                          $SUPRS: MOV     R0,-(SP)        ;SAVE R0
027704  016600  000004                          MOV     4(SP),R0        ;PICKUP THE POINTER
027710  105710                          1$:     TSTB    (R0)            ;TERMINATOR ?
027712  001403                                  BEQ     2$              ;BR IF YES
027714  122720  000060                          CMPB    #'0,(R0)+       ;IS THIS AN ASCII '0' ?
027720  001773                                  BEQ     1$              ;BR IF YES
027722  005300                          2$:     DEC     R0              ;BACKUP BY '1'
027724  010037  027732                          MOV     R0,3$           ;SAVE FOR TYPING
027730  104414                                  DISPLY                  ;GO PRINT
027732  000000                          3$:     .WORD   0               ;ASCIZ POINTER GOES HERE
027734  012600                                  MOV     (SP)+,R0        ;RESTORE R0
027736  012616                                  MOV     (SP)+,(SP)      ;RESTORE THE STACK
027740  000207                                  RTS     PC              ;RETURN

                                        ;ROUTINE TO TYPE AT PRIORITY 4

027742  013746  177776                  TYPRI4: MOV     @#PS,-(SP)      ;SAVE THE PRESENT STATUS
027746  012737  000200  177776                  MOV     #200,@#PS       ;CHANGE THE PRIORITY TO 4
027754  012537  027764                          MOV     (R5)+,1$        ;MESSAGE ADDRESS
027760  004737  032570                          JSR     PC,$TYPE        ;TYPE THE MESSAGE
027764  000000                          1$:     .WORD   0               ;MESSAGE ADDRESS GOES HERE
027766  000205                                  RTS     R5              ;RETURN

                                        ;ROUTINE TO TYPE ERRORS
                                        ;CALL
                                        ;       DISPLY                  ;MUST DEFINED IN 'TRAP' TABLE
                                        ;       MESADR                  ;ADDRESS OF MESSAGE
                                        ;       RETURN

027770  032777  020000  151142          $DSPLY: BIT     #BIT13,@SWR     ;INHIBIT ERROR TYPEOUT ?
027776  001004                                  BNE     1$              ;BR IF YES
030000  005037  177776                          CLR     @#PS            ;SET PRIORITY TO ZERO
030004  000137  032570                          JMP     $TYPE           ;TYPE THE MESSAGE
030010  062716  000002                  1$:     ADD     #2,(SP)         ;INCREMENT THE RETURN
030014  000002                                  RTI                     ;RETURN

                                        ;THIS ROUTINE IS USED TO CHECK IF AN
                                        ;ASCII CHARACTER IS A DIGIT BETWEEN 0 AND 7.
                                        ;CALL
                                        ;       MOV     #ADR,R1         ;ADDRESS OF ASCII CHARACTER
                                        ;       JSR     R5,CK.OCT       ;CHECK THE CHARACTER
                                        ;       RETURN1                 ;CHARACTER IS NOT BETWEEN 0-7
                                        ;       RETURN2                 ;CHARACTER IS IN R2 AS A
                                        ;                               ;OCTAL DIGIT

030016  121127  000060                  CK.OCT: CMPB    (R1),#'0        ;LESS THAN ZERO?
030022  103407                                  BLO     1$              ;YES -- BRANCH
030024  121127  000067                          CMPB    (R1),#'7        ;GREATER THAN SEVEN?
030030  101004                                  BHI     1$              ;YES -- BRANCH
030032  111102                                  MOVB    (R1),R2         ;GET THE CHARACTER
```

```
030034  042702  177770                    BIC     #^C7,R2            ;STRIP AWAY THE ASCII
030040  005725                            TST     (R5)+              ;ADJUST FOR RETURN
030042  000205                    1$:     RTS     R5                 ;RETURN

                                  ;THIS ROUTINE IS USED TO CHECK AN ASCII CHARACTER
                                  ;AND DETERMINE IF IT IS A DIGIT BETWEEN 0 AND 9.
                                  ;CALL
                                  ;       MOV     #ADR,R1            ;ADDRESS OF ASCII CHARACTER
                                  ;       JSR     R5,CK.DEC          ;CHECK THE CHARACTER
                                  ;       RETURN1                    ;NOT BETWEEN 0 AND 9
                                  ;       RETURN2                    ;BETWEEN 0 AND 9
                                  ;                                  ;R2 = DIGIT

030044  121127  000060            CK.DEC: CMPB    (R1),#'0           ;LESS THAN ZERO?
030050  103407                            BLO     1$                 ;YES -- BRANCH
030052  121127  000071                    CMPB    (R1),#'9           ;GREATER THAN NINE?
030056  101004                            BHI     1$                 ;YES -- BRANCH
030060  111102                            MOVB    (R1),R2            ;GET THE CHARACTER
030062  042702  000060                    BIC     #'0,R2             ;STRIP AWAY THE ASCII
030066  005725                            TST     (R5)+              ;ADJUST FOR RETURN
030070  000205                    1$:     RTS     R5                 ;RETURN

                                  ;THIS ROUTINE WILL CHECK AN ASCII CHARACTER TO
                                  ;DETERMINE WHAT IT IS.
                                  ;CALL
                                  ;       MOV     #ADR,R1            ;ADDRESS OF ASCII CHARACTER
                                  ;       JSR     R5,CK.CHR          ;CHECK CHARACTER
                                  ;       RETURN  ADR1               ;UNKNOWN CHARACTER
                                  ;       RETURN  ADR2               ;CARRIAGE RETURN * (R1)=ADR+1
                                  ;       RETURN  ADR3               ;COMMA * (R1)=ADR+1
                                  ;       RETURN  ADR4               ;PERIOD * (R1)=ADR+1
                                  ;       RETURN  ADR5               ;DIGIT BETWEEN 0 AND 7.
                                  ;       RETURN  ADR6               ;DIGIT BETWEEN 8 AND 9.
                                  ;                                  ;R2 = DIGIT * (R1)=ADR+1

030072  105711                    CK.CHR: TSTB    (R1)               ;''CARRIAGE RETURN''?
030074  001417                            BEQ     3$                 ;YES -- BRANCH
030076  121127  000054                    CMPB    (R1),#',           ;''COMMA''?
030102  001413                            BEQ     2$                 ;YES -- BRANCH
030104  121127  000056                    CMPB    (R1),#'.           ;''PERIOD''?
030110  001407                            BEQ     1$                 ;YES -- BRANCH
030112  004537  030044                    JSR     R5,CK.DEC          ;''DIGIT''?
030116  000410                            BR      4$                 ;NO -- BRANCH
030120  004537  030016                    JSR     R5,CK.OCT          ;OCTAL ?
030124  005725                            TST     (R5)+              ;DIGIT BETWEEN 8-9
030126  005725                            TST     (R5)+              ;DIGIT BETWEEN 0-7
030130  005725                    1$:     TST     (R5)+              ;PERIOD
030132  005725                    2$:     TST     (R5)+              ;COMMA
030134  005725                    3$:     TST     (R5)+              ;CARRIAGE RETURN
030136  005201                            INC     R1                 ;MOVE POINTER TO NEXT CHARACTER
030140  011505                    4$:     MOV     (R5),R5            ;UNKNOWN CHARACTER
030142  000205                            RTS     R5                 ;RETURN

                                  ;THIS ROUTINE CHECKS AN ASCII STRING FOR LEGAL
                                  ;CHARACTERS AND FORMS A DECIMAL VALUE BINARY NUMBER IN R2.
                                  ;CALL
                                  ;       MOV     #ADR,R1            ;ADDRESS OF ASCIZ STRING
```

```
                                 ;        MOV     #NUM,R2          ;MAX. MAGNITUDE OF INPUT NUMBER
                                 ;        JSR     R5,CK.DIG        ;CHECK DIGITS
                                 ;        RETURN  ADR1             ;"CR" ONLY ENTERED -- R2=0
                                 ;        RETURN  ADR2             ;"PERIOD" ONLY ENTERED -- R2=0
                                 ;        RETURN  ADR3             ;ILLEGAL CHARACTER OR INPUT TOO LARGE -- R2=?
                                 ;        RETURN  ADR4             ;"CR" -- R2 = NUMBER
                                 ;        RETURN  ADR5             ;"COMMA" -- R2 = NUMBER
                                 ;        RETURN  ADR6             ;"PERIOD" -- R2 = NUMBER

 030144  010446          CK.DIG: MOV      R4,-(SP)         ;SAVE R4
 030146  010346                  MOV      R3,-(SP)         ;SAVE R3
 030150  010246                  MOV      R2,-(SP)         ;SAVE THE MAX. SIZE ON THE STACK
 030152  005002                  CLR      R2               ;START WITH 0
 030154  005003                  CLR      R3
 030156  005004                  CLR      R4
 030160  004537  030072          JSR      R5,CK.CHR        ;CHECK ONE CHARACTER
 030164  030260                  6$                        ;ILLEGAL CHARACTER
 030166  030266                  9$                        ;CARRIAGE RETURN
 030170  030260                  6$                        ;" "
 030172  030262                  7$                        ;","
 030174  030200                  1$                        ;DIGIT 0-7
 030176  030200                  1$                        ;DIGIT 8-9
 030200  062705  000004  1$:     ADD      #4,R5            ;STEP RETURN POINTER PAST "CR" & "PERIOD" RETURNS
 030204  006303          2$:     ASL      R3               ;INPUT NUMBER *2
 030206  010346                  MOV      R3,-(SP)         ;SAVE *2
 030210  006303                  ASL      R3               ;*4
 030212  006303                  ASL      R3               ;*8
 030214  062603                  ADD      (SP)+,R3         ;(*2)+(*8) = *10
 030216  060203                  ADD      R2,R3            ;UPDATE THE INPUT NUMBER
 030220  004537  030072          JSR      R5,CK.CHR        ;CHECK ONE CHARACTER
 030224  030264                  8$                        ;ILLEGAL CHARACTER
 030226  030250                  5$                        ;CARRIAGE RETURN
 030230  030246                  4$                        ;" "
 030232  030240                  3$                        ;","
 030234  030204                  2$                        ;DIGIT 0-7
 030236  030204                  2$                        ;DIGIT 8-9
 030240  105711          3$:     TSTB     (R1)             ;DOES A "CR" FOLLOW THE "PERIOD"
 030242  001010                  BNE      8$               ;BR IF NOT
 030244  005724                  TST      (R4)+            ;INCREMENT THE RETURN
 030246  005724          4$:     TST      (R4)+            ;INCREMENT THE RETURN
 030250  005724          5$:     TST      (R4)+            ;INCREMENT THE RETURN
 030252  020316                  CMP      R3,(SP)          ;CHECK THE MAGNITUDE OF THE NUMBER
 030254  101004                  BHI      9$               ;BR IF ENTERED NUMBER TOO LARGE
 030256  000402                  BR       8$               ;BYPASS INCREMENT
 030260  005725          6$:     TST      (R5)+            ;INCREMENT RETURN PAST INVALID RETURN
 030262  005725          7$:     TST      (R5)+            ;INCREMENT RETURN
 030264  060405          8$:     ADD      R4,R5            ;SETUP RETURN POINTER
 030266  010302          9$:     MOV      R3,R2            ;ENTERED VALUE
 030270  005726                  TST      (SP)+            ;CLEAN MAX. SIZE OFF OF STACK
 030272  012603                  MOV      (SP)+,R3         ;RESTORE R3
 030274  012604                  MOV      (SP)+,R4         ;RESTORE R4
 030276  011505                  MOV      (R5),R5          ;GET RETURN ADDRESS
 030300  000205                  RTS      R5               ;RETURN

                         ;THIS ROUTINE WILL CONVERT A 16-BIT UNSIGNED BINARY NUMBER TO AN
                         ;UNSIGNED DECIMAL ASCIZ NUMBER.
                         ;CALL
```

```
                             ;       MOV     NUMBER,-(SP)    ;PUT THE NUMBER ON THE STACK
                             ;       JSR     PC,$SB2D        ;CALL
                             ;       RETURN                  ;ADDRESS OF THE 1ST ASCIZ CHAR IS ON THE STACK
                             ;
                             ;NOTE:  THE PROGRAM REQUIRES THIS FORM OF '$SB2D', NOT THE VERSION ON
                             ;       THE SYSMAC LIBRARY, REV C AND LATER

030302 016637 000002 030326  $SB2D:  MOV     2(SP),1$        ;SAVE THE BINARY NUMBER
030310 012746 030326                 MOV     #1$,-(SP)       ;SET THE POINTER
030314 004737 033732                 JSR     PC,$DB2D        ;CALL THE DOUBLE LENGTH CONVERT
030320 012666 000002                 MOV     (SP)+,2(SP)     ;PICKUP THE POINTER
030324 000207                        RTS     PC              ;RETURN
030326 000000 000000         1$:     .WORD   0,0

                             ;THIS ROUTINE WILL CONVERT A 16-BIT UNSIGNED BINARY NUMBER TO AN
                             ;UNSIGNED OCTAL ASCIZ NUMBER.
                             ;CALL
                             ;       MOV     NUMBER,-(SP)    ;PUT THE NUMBER ON THE STACK
                             ;       JSR     PC,$SB2O        ;CALL
                             ;       RETURN                  ;ADDRESS OF THE 1ST ASCIZ CHAR IS ON THE STACK
                             ;
                             ;NOTE:  THE PROGRAM REQUIRES THIS FORM OF '$SB2O', NOT THE VERSION ON
                             ;       THE SYSMAC LIBRARY, REV C AND LATER

030332 016637 000002 030356  $SB2O:  MOV     2(SP),1$        ;SAVE THE BINARY NUMBER
030340 012746 030356                 MOV     #1$,-(SP)       ;SET THE POINTER
030344 004737 034126                 JSR     PC,$DB2O        ;CALL THE DOUBLE LENGTH CONVERT
030350 012666 000002                 MOV     (SP)+,2(SP)     ;PICKUP THE POINTER
030354 000207                        RTS     PC              ;RETURN
030356 000000 000000         1$:     .WORD   0,0
```

```
                                        .SBTTL  TTY INPUT ROUTINE

                                        ;;*****************************************************************
                                        .ENABL  LSB
030362  000000                          $TKCNT: .WORD   0                       ;;NUMBER OF ITEMS IN QUEUE
030364  000000                          $TKQIN: .WORD   0                       ;;INPUT POINTER
030366  000000                          $TKQOUT: .WORD  0                       ;;OUTPUT POINTER
030370                                  $TKQSRT: .BLKB  7                       ;;TTY KEYBOARD QUEUE
        030377                          $TKQEND=.
                                        .EVEN

                                        ;*TK INITIALIZE ROUTINE
                                        ;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
                                        ;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
                                        ;
                                        ;*CALL:
                                        ;*       JSR     PC,$TKINT
                                        ;*       RETURN
                                        ;
030400  005037  030362                  $TKINT: CLR     $TKCNT          ;;CLEAR COUNT OF ITEMS IN QUEUE
030404  012737  030370  030364                  MOV     #$TKQSRT,$TKQIN ;;MOVE THE STARTING ADDRESS OF THE
030412  013737  030364  030366                  MOV     $TKQIN,$TKQOUT  ;;QUEUE INTO THE INPUT & OUTPUT POINTERS.
030420  012737  030450  000060                  MOV     #$TKSRV,@#TKVEC ;;INITIALIZE THE KEYBOARD VECTOR
030426  012737  000200  000062                  MOV     #200,@#TKVEC+2  ;;"BR" LEVEL 4
030434  005777  150506                          TST     @$TKB           ;;CLEAR DONE FLAG
030440  012777  000100  150476                  MOV     #100,@$TKS      ;;ENABLE TTY KEYBOARD INTERRUPT
030446  000207                                  RTS     PC              ;;RETURN TO CALLER

                                        ;*TK SERVICE ROUTINE
                                        ;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
                                        ;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
                                        ;*IT IN THE QUEUE.
                                        ;*IF THE CHARACTER IS A "CONTROL-C" (^C) $TKINT IS CALLED AND
                                        ;*UPON RETURN EXIT IS MADE TO THE "CONTROL-C" RESTART ADDRESS (CTRAP)
                                        ;
030450  117746  150472                  $TKSRV: MOVB    @$TKB,-(SP)     ;;PICKUP THE CHARACTER
030454  042716  177600                          BIC     #^C177,(SP)     ;;STRIP THE JUNK
030460  021627  000021                          CMP     (SP),#$XON      ;;IS IT A RANDOM XON?
030464  001002                                  BNE     30$             ;;BRANCH IF NO
030466  005726                                  TST     (SP)+           ;;CLEAN RANDOM XON OFF STACK
030470  000002                                  RTI                     ;;RETURN
030472                                  30$:
030472  021627  000003                          CMP     (SP),#3         ;;IS IT A CONTROL C?
030476  001007                                  BNE     1$              ;;BRANCH IF NO
030500  104401  031605                          TYPE    ,$CNTLC         ;;TYPE A CONTROL-C (^C)
030504  004737  030400                          JSR     PC,$TKINT       ;;INIT THE KEYBOARD
030510  005726                                  TST     (SP)+           ;;CLEAN UP STACK
030512  000137  031646                          JMP     CTRAP           ;;CONTROL C RESTART
030516  021627  000007                  1$:     CMP     (SP),#7         ;;IS IT A CONTROL G?
030522  001004                                  BNE     2$              ;;BRANCH IF NO
030524  022737  000176  001140                  CMP     #SWREG,SWR      ;;IS SOFT-SWR SELECTED?
030532  001500                                  BEQ     6$              ;;GO TO SWR CHANGE

030534                                  2$:
030534  022737  000007  030362                  CMP     #7,$TKCNT       ;;IS THE QUEUE FULL?
030542  001004                                  BNE     3$              ;;BRANCH IF NO
030544  104401  001160                          TYPE    ,$BELL          ;;RING THE TTY BELL
```

```
030550  005726                          TST     (SP)+           ;;CLEAN CHARACTER OFF OF STACK
030552  000451                          BR      5$              ;;EXIT
030554  021627  000023          3$:     CMP     (SP),#23        ;;IS IT A CONTROL-S?
030560  001021                          BNE     32$             ;;BRANCH IF NO
030562  005077  150356                  CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
030566  005726                          TST     (SP)+           ;;CLEAN CHAR OFF STACK
030570  105777  150350          31$:    TSTB    @$TKS           ;;WAIT FOR A CHAR
030574  100375                          BPL     31$             ;;LOOP UNTIL ITS THERE
030576  117746  150344                  MOVB    @$TKB,-(SP)     ;;GET THE CHARACTER
030602  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
030606  022627  000021                  CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
030612  001366                          BNE     31$             ;;BRANCH IF NO
030614  012777  000100  150322          MOV     #100,@$TKS      ;;REENABLE TTY KEYBOARD INTERRUPTS
030622  000002                          RTI                     ;;RETURN
030624  005237  030362          32$:    INC     $TKCNT          ;;COUNT THIS CHARACTER
030630  021627  000140                  CMP     (SP),#140       ;;IS IT UPPER CASE?
030634  002405                          BLT     4$              ;;BRANCH IF YES
030636  021627  000175                  CMP     (SP),#175       ;;IS IT A SPECIAL CHAR?
030642  003002                          BGT     4$              ;;BRANCH IF YES
030644  042716  000040                  BIC     #40,(SP)        ;;MAKE IT UPPER CASE
030650  112677  177510          4$:     MOVB    (SP)+,@$TKQIN   ;;AND PUT IT IN QUEUE
030654  005237  030364                  INC     $TKQIN          ;;UPDATE THE POINTER
030660  023727  030364  030377          CMP     $TKQIN,#$TKQEND ;;GO OFF THE END?
030666  001003                          BNE     5$              ;;BRANCH IF NO
030670  012737  030370  030364          MOV     #$TKQSRT,$TKQIN ;;RESET THE POINTER
030676  000002                  5$:     RTI                     ;;RETURN

                                ;;*************************************************************
                                ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
                                ;*ROUTINE IS ENTERED  FROM THE TRAP HANDLER, AND WILL
                                ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP
                                ;*CALL WHEN OPERATING IN TTY INTERRUPT MODE.
030700  022737  000176  001140  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED
030706  001124                          BNE     15$             ;;EXIT IF NOT
030710  105777  150230                  TSTB    @$TKS           ;;IS A CHAR WAITING?
030714  100121                          BPL     15$             ;;IF NOT, EXIT
030716  117746  150224                  MOVB    @$TKB,-(SP)     ;;YES
030722  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT  7-BIT ASCII
030726  021627  000007                  CMP     (SP),#7         ;;IS IT A CONTROL-G?
030732  001300                          BNE     2$              ;;IF NOT, PUT IT IN THE TTY QUEUE
                                                                ;;AND EXIT

                                ;;*************************************************************
                                ;*CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
                                ;*ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
                                ;*CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.
030734  123727  001134  000001  6$:     CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
030742  001674                          BEQ     2$              ;;BRANCH IF YES
030744  005726                          TST     (SP)+           ;;CLEAR CONTROL-G OFF STACK
030746  004737  030400                  JSR     PC,$TKINT       ;;FLUSH THE TTY INPUT QUEUE
030752  005077  150166                  CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
030756  112737  000001  001135          MOVB    #1,$INTAG       ;;SET INTERRUPT MODE INDICATOR

030764  104401  031617                  TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
030770  104401  031624          $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
030774  013746  000176                  MOV     SWREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
031000  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
```

```
031002  104401  031635                     TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
031006  005046                    19$:     CLR     -(SP)           ;;CLEAR COUNTER
031010  005046                             CLR     -(SP)           ;;THE NEW SWR
031012  105777  150126            7$:      TSTB    @$TKS           ;;CHAR THERE?
031016  100375                             BPL     7$              ;;IF NOT TRY AGAIN

031020  117746  150122                     MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
031024  042716  177600                     BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII

031030  021627  000003                     CMP     (SP),#3         ;;IS IT A CONTROL-C?
031034  001015                             BNE     9$              ;;BRANCH IF NOT
031036  104401  031605                     TYPE    ,$CNTLC         ;;YES, ECHO CONTROL-C (^C)
031042  062706  000006                     ADD     #6,SP           ;;CLEAN UP STACK
031046  123727  001135  000001             CMPB    $INTAG,#1       ;;REENABLE TTY KEYBOARD INTERRUPTS?
031054  001003                             BNE     8$              ;;BRANCH IF NO
031056  012777  000100  150060             MOV     #100,@$TKS      ;;ALLOW TTY KEYBOARD INTERRUPTS
031064  000137  031646            8$:      JMP     CTRAP           ;;CONTROL-C RESTART


031070  021627  000025            9$:      CMP     (SP),#25        ;;IS IT A CONTROL-U?
031074  001005                             BNE     10$             ;;BRANCH IF NOT
031076  104401  031612                     TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (^U)
031102  062706  000006            20$:     ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
031106  000737                             BR      19$             ;;LET'S TRY IT AGAIN


031110  021627  000015            10$:     CMP     (SP),#15        ;;IS IT A <CR>?
031114  001022                             BNE     16$             ;;BRANCH IF NO
031116  005766  000004                     TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
031122  001403                             BEQ     11$             ;;BRANCH IF YES
031124  016677  000002  150006             MOV     2(SP),@SWR      ;;SAVE NEW SWR
031132  062706  000006            11$:     ADD     #6,SP           ;;CLEAN UP STACK
031136  104401  001165            14$:     TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
031142  123727  001135  000001             CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
031150  001003                             BNE     15$             ;;BRANCH IF NOT
031152  012777  000100  147764             MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
031160  000002                    15$:     RTI                     ;;RETURN
031162  004737  032740            16$:     JSR     PC,$TYPEC       ;;ECHO CHAR
031166  021627  000060                     CMP     (SP),#60        ;;CHAR < 0?
031172  002420                             BLT     18$             ;;BRANCH IF YES
031174  021627  000067                     CMP     (SP),#67        ;;CHAR > 7?
031200  003015                             BGT     18$             ;;BRANCH IF YES
031202  042726  000060                     BIC     #60,(SP)+       ;;STRIP-OFF ASCII
031206  005766  000002                     TST     2(SP)           ;;IS THIS THE FIRST CHAR
031212  001403                             BEQ     17$             ;;BRANCH IF YES
031214  006316                             ASL     (SP)            ;;NO, SHIFT PRESENT
031216  006316                             ASL     (SP)            ;;    CHAR OVER TO MAKE
031220  006316                             ASL     (SP)            ;;    ROOM FOR NEW ONE.
031222  005266  000002            17$:     INC     2(SP)           ;;KEEP COUNT OF CHAR
031226  056616  177776                     BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
031232  000667                             BR      7$              ;;GET THE NEXT ONE
031234  104401  001164            18$:     TYPE    ,$QUES          ;;TYPE ?<CR><LF>
031240  000720                             BR      20$             ;;SIMULATE CONTROL-U
                                  .DSABL   LSB

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

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

031242  011646                          $RDCHR: MOV     (SP),-(SP)              ;;PUSH DOWN THE PC AND
031244  016666  000004  000002                  MOV     4(SP),2(SP)             ;;THE PS
031252  005066  000004                          CLR     4(SP)                   ;;GET READY FOR A CHARACTER
031256  005046                                  CLR     -(SP)                   ;;PUT NEW PS ON STACK
031260  012746  031266                          MOV     #64$,-(SP)              ;;PUT NEW PC ON  STACK
031264  000002                                  RTI                             ;;POP NEW PC AND PS
031266                                  64$:
031266  005737  030362                  1$:     TST     $TKCNT                  ;;WAIT ON A CHARACTER
031272  001775                                  BEQ     1$
031274  005337  030362                          DEC     $TKCNT                  ;;DECREMENT THE COUNTER
031300  117766  177062  000004                  MOVB    @$TKQOUT,4(SP)          ;;GET ONE CHARACTER
031306  005237  030366                          INC     $TKQOUT                 ;;UPDATE THE POINTER
031312  023727  030366  030377                  CMP     $TKQOUT,#$TKQEND        ;;DID IT GO OFF OF THE END?
031320  001003                                  BNE     2$                      ;;BRANCH IF NO
031322  012737  030370  030366                  MOV     #$TKQSRT,$TKQOUT        ;;RESET THE POINTER
031330  000002                          2$:     RTI                             ;;RETURN
                                        ;;*******************************************************************
                                        ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
                                        ;*CALL:
                                        ;*       RDLIN                           ;;INPUT A STRING FROM THE TTY
                                        ;*       RETURN HERE                     ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
                                        ;*                                       ;;TERMINATOR WILL BE A BYTE OF ALL 0'S

031332  010346                          $RDLIN: MOV     R3,-(SP)                ;;SAVE R3
031334  005046                                  CLR     -(SP)                   ;;CLEAR THE RUBOUT KEY
031336  012703  031566                  1$:     MOV     #$TTYIN,R3              ;;GET ADDRESS
031342  022703  031605                  2$:     CMP     #$TTYIN+15.,R3          ;;BUFFER FULL?
031346  101456                                  BLOS    4$                      ;;BR IF YES
031350  104410                                  RDCHR                           ;;GO READ ONE CHARACTER FROM THE TTY
031352  112613                                  MOVB    (SP)+,(R3)              ;;GET CHARACTER
031354  122713  000177                  10$:    CMPB    #177,(R3)               ;;IS IT A RUBOUT
031360  001022                                  BNE     5$                      ;;BR IF NO
031362  005716                                  TST     (SP)                    ;;IS THIS THE FIRST RUBOUT?
031364  001007                                  BNE     6$                      ;;BR IF NO
031366  112737  000134  031564                  MOVB    #'\,9$                  ;;TYPE A BACK SLASH
031374  104401  031564                          TYPE    ,9$
031400  012716  177777                          MOV     #-1,(SP)                ;;SET THE RUBOUT KEY
031404  005303                          6$:     DEC     R3                      ;;BACKUP BY ONE
031406  020327  031566                          CMP     R3,#$TTYIN              ;;STACK EMPTY?
031412  103434                                  BLO     4$                      ;;BR IF YES
031414  111337  031564                          MOVB    (R3),9$                 ;;SETUP TO TYPEOUT THE DELETED CHAR.
031420  104401  031564                          TYPE    ,9$                     ;;GO TYPE
031424  000746                                  BR      2$                      ;;GO READ ANOTHER CHAR.
031426  005716                          5$:     TST     (SP)                    ;;RUBOUT KEY SET?
031430  001406                                  BEQ     7$                      ;;BR IF NO
031432  112737  000134  031564                  MOVB    #'\,9$                  ;;TYPE A BACK SLASH
031440  104401  031564                          TYPE    ,9$
031444  005016                                  CLR     (SP)                    ;;CLEAR THE RUBOUT KEY
031446  122713  000025                  7$:     CMPB    #25,(R3)                ;;IS CHARACTER A CTRL U?
031452  001003                                  BNE     8$                      ;;BR IF NO
```

```
031454  104401  031612                          TYPE    ,$CNTLU         ;;TYPE A CONTROL 'U'
031460  000726                                  BR      1$              ;;GO START OVER
031462  122713  000022                  8$:     CMPB    #22,(R3)        ;;IS CHARACTER A '^R'?
031466  001011                                  BNE     3$              ;;BRANCH IF NO
031470  105013                                  CLRB    (R3)            ;;CLEAR THE CHARACTER
031472  104401  001165                          TYPE    ,$CRLF          ;;TYPE A "CR" & "LF"
031476  104401  031566                          TYPE    ,$TTYIN         ;;TYPE THE INPUT STRING
031502  000717                                  BR      2$              ;;GO PICKUP ANOTHER CHACTER
031504  104401  001164                  4$:     TYPE    ,$QUES          ;;TYPE A '?'
031510  000712                                  BR      1$              ;;CLEAR THE BUFFER AND LOOP
031512  111337  031564                  3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
031516  104401  031564                          TYPE    ,9$
031522  122723  000015                          CMPB    #15,(R3)+       ;;CHECK FOR RETURN
031526  001305                                  BNE     2$              ;;LOOP IF NOT RETURN
031530  105063  177777                          CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
031534  104401  001166                          TYPE    ,$LF            ;;TYPE A LINE FEED
031540  005726                                  TST     (SP)+           ;;CLEAN RUBOUT KEY FROM THE STACK
031542  012603                                  MOV     (SP)+,R3        ;;RESTORE R3
031544  011646                                  MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
031546  016666  000004  000002                  MOV     4(SP),2(SP)     ;;        FIRST ASCII CHARACTER ON IT
031554  012766  031566  000004                  MOV     #$TTYIN,4(SP)
031562  000002                                  RTI                     ;;RETURN
031564     000                          9$:     .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
031565     000                                  .BYTE   0               ;;TERMINATOR
031566                                  $TTYIN: .BLKB   15.             ;;RESERVE 15. BYTES FOR TTY INPUT
031605     136     103     015          $CNTLC: .ASCIZ  /^C/<15><12>    ;;CONTROL 'C'
031612     136     125     015          $CNTLU: .ASCIZ  /^U/<15><12>    ;;CONTROL 'U'
031617     136     107     015          $CNTLG: .ASCIZ  /^G/<15><12>    ;;CONTROL 'G'
031624     015     012     123          $MSWR:  .ASCIZ  <15><12>/SWR = /
031635     040     040     116          $MNEW:  .ASCIZ  /  NEW = /

                                        ;THIS ROUTINE WILL PROCESS THE (^C) CHARACTER

031646  012737  000001  001262          CTRAP:  MOV     #1,CFLAG                ;SET THE 'CONTROL C' FLAG
031654  005237  030362                          INC     $TKCNT                  ;COUNT THIS CHARACTER
031660  112777  000015  176476                  MOVB    #15,@$TKQIN             ;PUT 'RETURN' CHARACTER IN QUEUE
031666  005237  030364                          INC     $TKQIN                  ;UPDATE THE POINTER
031672  023727  030364  030377                  CMP     $TKQIN,#$TKQEND ;GO OFF THE END ?
031700  001003                                  BNE     1$                      ;BR IF YES
031702  012737  030370  030364                  MOV     #$TKQSRT,$TKQIN ;RESET THE POINTER
031710  000002                          1$:     RTI                             ;RETURN
```

```
                                           .SBTTL  ERROR HANDLER ROUTINE

                                           ;;*****************************************************************
                                           ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
                                           ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
                                           ;*AND GO TO $ERRTYP ON ERROR
                                           ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
                                           ;*SW15=1         HALT ON ERROR
                                           ;*SW13=1         INHIBIT ERROR TYPEOUTS
                                           ;*SW10=1         BELL ON ERROR
                                           ;*CALL
                                           ;*       ERROR   N        ;;ERROR=EMT AND N=ERROR ITEM NUMBER

031712  105037  032246                     $ERROR: CLRB    IBSAVE          ;;CLEAR THE ITEM BYTE SAVE LOCATION
031716  104407                                     CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
031720  010337  001244                             MOV     R3,ATTN         ;SAVE THE ATTENTICN REGISTER CONTENTS
031724  010137  001242                             MOV     R1,DRIVE        ;DRIVE NUMBER
031730  032777  020000  147202                     BIT     #SW13,@SWR      ;INHIBIT PRINTOUTS ?
031736  001002                                     BNE     .+6             ;BR IF YES
031740  004737  023752                             JSR     PC,$TIME        ;TYPE THE TIME
031744  105237  001103                     7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
031750  001775                                     BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
031752  013777  001102  147162                     MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
031760  032777  002000  147152                     BIT     #BIT10,@SWR     ;;BELL ON ERROR?
031766  001402                                     BEQ     1$              ;;NO - SKIP
031770  104401  001160                             TYPE    ,$BELL          ;;RING BELL
031774  005237  001112                     1$:     INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
032000  011637  001116                             MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
032004  162737  000002  001116                     SUB     #2,$ERRPC
032012  117737  147100  001114                     MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
032020  032777  001000  147112                     BIT     #BIT09,@SWR     ;;SEE IF LOOP ON ERROR IS SET
032026  001060                                     BNE     1004$           ;;BRANCH AROUND ROUTINE IF SO
032030  122737  000177  001114                     CMPB    #177,$ITEMB     ;;SEE IF THIS IS THE POWER FAIL CALL
032036  001454                                     BEQ     1004$           ;;BRANCH AROUND ROUTINE IF IT IS
032040  105737  032246                             TSTB    IBSAVE          ;;SEE IF THIS IS THE 2ND ERROR CALL IN THIS ROUTINE
032044  001047                                     BNE     1003$           ;;BRANCH IF SO
032046  022737  177777  032244                     CMP     #-1,CPSAVE      ;;SEE IF CPSAVE HAS CPU ERR REG TIMEOUT INDICATION
032054  001445                                     BEQ     1004$           ;;BRANCH IF SO
032056  013746  000004                             MOV     ERRVEC,-(SP)    ;;SAVE CONTENTS OF ERROR VECTOR
032062  012737  032100  000004                     MOV     #1000$,ERRVEC   ;;SETUP 'TRAP' RETURN ADDRESS
032070  013737  177766  032244                     MOV     177766,CPSAVE   ;;MOVE CPU ERROR REGISTER TO CPSAVE FOR TEST
032076  000406                                     BR      1001$
032100  012737  177777  032244             1000$:  MOV     #-1,CPSAVE      ;;SET CPU ERROR REGISTER TIMEOUT INDICATOR
032106  012716  032114                             MOV     #1001$,(SP)     ;;SETUP RETURN ADDRESS
032112  000002                                     RTI
032114  012637  000004                     1001$:  MOV     (SP)+,ERRVEC    ;;RESTORE CONTENTS OF ERROR VECTOR

032120  022737  177777  032244             1002$:  CMP     #-1,CPSAVE      ;;SEE IF CPSAVE HAS CPU ERR REG TIMEOUT INDICATION
032126  001420                                     BEQ     1004$           ;;BRANCH IF SO
032130  032737  000001  032244                     BIT     #BIT00,CPSAVE   ;;SEE IF POWER MONITOR BIT IS SET IN CPU ERR REG
032136  001414                                     BEQ     1004$           ;;BRANCH IF OK
032140  042737  000001  177766                     BIC     #BIT00,177766   ;;CLEAR THE BIT FOUND SET
032146  113737  001114  032246                     MOVB    $ITEMB,IBSAVE   ;;MAKE IBSAVE NON-ZERO FOR DUAL ERROR CALL
032154  112737  000177  001114                     MOVB    #177,$ITEMB     ;;SET $ITEMB TO SPECIAL POWER FAIL POINTER
032162  000402                                     BR      1004$           ;;BRANCH OVER IBSAVE CLEARING

032164  105037  032246                     1003$:  CLRB    IBSAVE          ;;CLEAR IBSAVE SO 2ND TIME THROUGH EXITS
```

```
032170                             1004$:
032170  032777  020000  146742             BIT     #BIT13,@SWR        ;;SKIP TYPEOUT IF SET
032176  001004                             BNE     20$                ;;SKIP TYPEOUTS
032200  004737  032250                     JSR     PC,$ERRTYP         ;;GO TO USER ERROR ROUTINE
032204  104401  001165                     TYPE    ,$CRLF
032210                             20$:
032210  105737  032246             2$:     TSTB    IBSAVE             ;;SEE IF IBSAVE IS LOADED
032214  001005                             BNE     3$                 ;;BRANCH IF NOT - NO HALT ON PWR MON BIT ERROR
032216  005777  146716                     TST     @SWR               ;;HALT ON ERROR
032222  100002                             BPL     3$                 ;;SKIP IF CONTINUE
032224  000000                             HALT                       ;;HALT ON ERROR!
032226  104407                             CKSWR                      ;;TEST FOR CHANGE IN SOFT-SWR
032230                             3$:
032230  023737  000042  000046             CMP     @#42,@#46          ;ARE WE IN ACT-11 AUTO MODE?
032236  001001                             BNE     .+4                ;BRANCH IF NOT
032240  000000                             HALT                       ;HALT ON ERROR IF ACT AUTO MODE
032242  000002                             RTI                        ;RETURN
032244  000000                     CPSAVE: .WORD   0                  ;;LOCATION TO SAVE CPU ERROR REG CONTENTS
032246  000000                     IBSAVE: .WORD   0                  ;;LOCATION TO SAVE ITEM BYTE
```

```
                                        .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE

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

        032250                          $ERRTYP:
        032250  104401  001165                  TYPE    ,$CRLF                  ;;"CARRIAGE RETURN" & "LINE FEED"
        032254  010046                          MOV     R0,-(SP)                ;;SAVE R0
        032256  005000                          CLR     R0                      ;;PICKUP THE ITEM INDEX
        032260  153700  001114                  BISB    @#$ITEMB,R0
        032264  001004                          BNE     1$                      ;;IF ITEM NUMBER IS ZERO, JUST
                                                                                ;;TYPE THE PC OF THE ERROR
        032266  013746  001116                  MOV     $ERRPC,-(SP)            ;;SAVE $ERRPC FOR TYPEOUT
                                                                                ;;ERROR ADDRESS
        032272  104402                          TYPOC                           ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
        032274  000437                          BR      6$                      ;;GET OUT
        032276  122700  000177          1$:     CMPB    #177,R0                 ;;SEE IF THIS ERROR CALL IS SPECIAL POWER FAIL CALL
        032302  001006                          BNE     1000$                   ;;BRANCH IF NOT
        032304  113737  001102  032566          MOVB    $TSTNM,PFTSTN           ;;GET TEST NUMBER
        032312  012700  032426                  MOV     #PFECH,R0               ;;MOVE POWER FAIL ERROR CALL TABLE TO R0
        032316  000406                          BR      1001$                   ;;BRANCH TO CALL ERROR
        032320  005300                  1000$:  DEC     R0                      ;;ADJUST THE INDEX SO THAT IT WILL
        032322  006300                          ASL     R0                      ;;       WORK FOR THE ERROR TABLE
        032324  006300                          ASL     R0
        032326  006300                          ASL     R0
        032330  062700  004026                  ADD     #$ERRTB,R0              ;;FORM TABLE POINTER
        032334  012037  032344          1001$:  MOV     (R0)+,2$                ;;PICKUP "ERROR MESSAGE" POINTER
        032340  001404                          BEQ     3$                      ;;SKIP TYPEOUT IF NO POINTER
        032342  104401                          TYPE                            ;;TYPE THE "ERROR MESSAGE"
        032344  000000                  2$:     .WORD   0                       ;;"ERROR MESSAGE" POINTER GOES HERE
        032346  104401  001165                  TYPE    ,$CRLF                  ;;"CARRIAGE RETURN" & "LINE FEED"
        032352  012037  032362          3$:     MOV     (R0)+,4$                ;;PICKUP "DATA HEADER" POINTER
        032356  001404                          BEQ     5$                      ;;SKIP TYPEOUT IF 0
        032360  104401                          TYPE                            ;;TYPE THE "DATA HEADER"
        032362  000000                  4$:     .WORD   0                       ;;"DATA HEADER" POINTER GOES HERE
        032364  104401  001165                  TYPE    ,$CRLF                  ;;"CARRIAGE RETURN" & "LINE FEED"
        032370  011000                  5$:     MOV     (R0),R0                 ;;PICKUP "DATA TABLE" POINTER
        032372  001004                          BNE     7$                      ;;GO TYPE THE DATA
        032374  012600                  6$:     MOV     (SP)+,R0                ;;RESTORE R0
        032376  104401  001165                  TYPE    ,$CRLF                  ;;"CARRIAGE RETURN" & "LINE FEED"
        032402  000207                          RTS     PC                      ;;RETURN
        032404                          7$:
        032404  013046                          MOV     @(R0)+,-(SP)            ;;SAVE @(R0)+ FOR TYPEOUT
        032406  104402                          TYPOC                           ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
        032410  005710                          TST     (R0)                    ;;IS THERE ANOTHER NUMBER?
        032412  001770                          BEQ     6$                      ;;BR IF NO
        032414  104401  032422                  TYPE    ,8$                     ;;TYPE TWO(2) SPACES
        032420  000771                          BR      7$                      ;;LOOP
        032422     040     040     000  8$:     .ASCIZ  /  /                    ;;TWO(2) SPACES
                                                .EVEN
        032426  032436  032520  032552  PFECH:  PFECH1,PFECH2,PFECH3,PFECH4 ;;WORDS DEFINING TABLES BELOW
        032436     120     117     127  PFECH1: .ASCIZ  ?POWER MONITOR BIT IN CPU ERROR REGISTER FOUND SET?
        032520     124     105     123  PFECH2: .ASCIZ  ?TESTNO  ERR PC  CPUERREG?
                                                .EVEN
        032552  032566  001116  032244  PFECH3: .WORD   PFTSTN,$ERRPC,CPSAVE,0
```

```
032562     000      000      000  PFECH4: .BYTE   0,0,0,0
032566  000000                    PFTSTN: .WORD   0                     ;;CONTAINS TEST NUMBER FOR PF BIT ERROR
```

```
                                        .SBTTL  TYPE ROUTINE

                                        ;;*******************************************************************
                                        ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
                                        ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
                                        ;*NOTE1:         $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
                                        ;*NOTE2:         $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
                                        ;*NOTE3:         $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
                                        ;*
                                        ;*CALL:
                                        ;*1) USING A TRAP INSTRUCTION
                                        ;*       TYPE    ,MESADR                 ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
                                        ;*OR
                                        ;*       TYPE
                                        ;*       MESADR
                                        ;*

032570  105737  001157                  $TYPE:  TSTB    $TPFLG                  ;;IS THERE A TERMINAL?
032574  100002                                  BPL     1$                      ;;BR IF YES
032576  000000                                  HALT                            ;;HALT HERE IF NO TERMINAL
032600  000407                                  BR      3$                      ;;LEAVE
032602  010046                          1$:     MOV     R0,-(SP)                ;;SAVE R0
032604  017600  000002                          MOV     @2(SP),R0               ;;GET ADDRESS OF ASCIZ STRING
032610  112046                          2$:     MOVB    (R0)+,-(SP)             ;;PUSH CHARACTER TO BE TYPED ONTO STACK
032612  001005                                  BNE     4$                      ;;BR IF IT ISN'T THE TERMINATOR
032614  005726                                  TST     (SP)+                   ;;IF TERMINATOR POP IT OFF THE STACK
032616  012600                          60$:    MOV     (SP)+,R0                ;;RESTORE R0
032620  062716  000002                  3$:     ADD     #2,(SP)                 ;;ADJUST RETURN PC
032624  000002                                  RTI                             ;;RETURN
032626  122716  000011                  4$:     CMPB    #HT,(SP)                ;;BRANCH IF <HT>
032632  001430                                  BEQ     8$
032634  122716  000200                          CMPB    #CRLF,(SP)              ;;BRANCH IF NOT <CRLF>
032640  001006                                  BNE     5$
032642  005726                                  TST     (SP)+                   ;;POP  <CR><LF> EQUIV
032644  104401                                  TYPE                            ;;TYPE A CR AND LF
032646  001165                                  $CRLF
032650  105037  033056                          CLRB    $CHARCNT                ;;CLEAR CHARACTER COUNT
032654  000755                                  BR      2$                      ;;GET NEXT CHARACTER
032656  004737  032740                  5$:     JSR     PC,$TYPEC               ;;GO TYPE THIS CHARACTER
032662  123726  001156                  6$:     CMPB    $FILLC,(SP)+            ;;IS IT TIME FOR FILLER CHARS.?
032666  001350                                  BNE     2$                      ;;IF NO GO GET NEXT CHAR.
032670  013746  001154                          MOV     $NULL,-(SP)             ;;GET # OF FILLER CHARS. NEEDED
                                                                                ;;AND THE NULL CHAR.
032674  105366  000001                  7$:     DECB    1(SP)                   ;;DOES A NULL NEED TO BE TYPED?
032700  002770                                  BLT     6$                      ;;BR IF NO--GO POP THE NULL OFF OF STACK
032702  004737  032740                          JSR     PC,$TYPEC               ;;GO TYPE A NULL
032706  105337  033056                          DECB    $CHARCNT                ;;DO NOT COUNT AS A COUNT
032712  000770                                  BR      7$                      ;;LOOP

                                        ;HORIZONTAL TAB PROCESSOR

032714  112716  000040                  8$:     MOVB    #' ,(SP)                ;;REPLACE TAB WITH SPACE
032720  004737  032740                  9$:     JSR     PC,$TYPEC               ;;TYPE A SPACE
032724  132737  000007  033056                  BITB    #7,$CHARCNT             ;;BRANCH IF NOT AT
032732  001372                                  BNE     9$                      ;;TAB STOP
032734  005726                                  TST     (SP)+                   ;;POP SPACE OFF STACK
032736  000724                                  BR      2$                      ;;GET NEXT CHARACTER
```

```
      032740                            $TYPEC:
      032740  105777  146200                    TSTB    @$TKS              ;;CHAR IN KYBD BUFFER?
      032744  100022                            BPL     10$                ;;BR IF NOT
      032746  017746  146174                    MOV     @$TKB,-(SP)        ;;GET CHAR
      032752  042716  177600                    BIC     #177600,(SP)       ;;STRIP EXTRANEOUS BITS
      032756  122716  000023                    CMPB    #$XOFF,(SP)        ;;WAS CHAR XOFF
      032762  001012                            BNE     102$               ;;BR IF NOT
      032764                            101$:
      032764  105777  146154                    TSTB    @$TKS              ;;WAIT FOR CHAR
      032770  100375                            BPL     101$
      032772  117716  146150                    MOVB    @$TKB,(SP)         ;;GET CHAR
      032776  042716  177600                    BIC     #177600,(SP)       ;;STRIP IT
      033002  122716  000021                    CMPB    #$XON,(SP)         ;;WAS IT XON?
      033006  001366                            BNE     101$               ;;BR IF NOT
      033010                            102$:
      033010  005726                            TST     (SP)+              ;;FIX STACK
      033012                            10$:
      033012  105777  146132                    TSTB    @$TPS              ;;WAIT UNTIL PRINTER IS READY
      033016  100375                            BPL     10$
      033020  116677  000002  146124            MOVB    2(SP),@$TPB        ;;LOAD CHAR TO BE TYPED INTO DATA REG.
      033026  122766  000015  000002            CMPB    #CR,2(SP)          ;;IS CHARACTER A CARRIAGE RETURN?
      033034  001003                            BNE     1$                 ;;BRANCH IF NO
      033036  105037  033056                    CLRB    $CHARCNT           ;;YES--CLEAR CHARACTER COUNT
      033042  000406                            BR      $TYPEX             ;;EXIT
      033044  122766  000012  000002    1$:     CMPB    #LF,2(SP)          ;;IS CHARACTER A LINE FEED?
      033052  001402                            BEQ     $TYPEX             ;;BRANCH IF YES
      033054  105227                            INCB    (PC)+              ;;COUNT THE CHARACTER
      033056  000000                    $CHARCNT:.WORD  0                  ;;CHARACTER COUNT STORAGE
      033060  000207                    $TYPEX: RTS     PC
```

```
                                        .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE

                                        ;;*******************************************************************
                                        ;;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
                                        ;;*OCTAL (ASCII) NUMBER AND TYPE IT.
                                        ;;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
                                        ;;*CALL:
                                        ;;*      MOV     NUM,-(SP)          ;;NUMBER TO BE TYPED
                                        ;;*      TYPOS                      ;;CALL FOR TYPEOUT
                                        ;;*      .BYTE   N                  ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
                                        ;;*      .BYTE   M                  ;;M=1 OR 0
                                        ;;*                                            ;;1=TYPE LEADING ZEROS
                                        ;;*                                            ;;0=SUPPRESS LEADING ZEROS
                                        ;;*
                                        ;;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
                                        ;;*$TYPOS OR $TYPOC
                                        ;;*CALL:
                                        ;;*      MOV     NUM,-(SP)          ;;NUMBER TO BE TYPED
                                        ;;*      TYPON                      ;;CALL FOR TYPEOUT
                                        ;;*
                                        ;;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
                                        ;;*CALL:
                                        ;;*      MOV     NUM,-(SP)          ;;NUMBER TO BE TYPED
                                        ;;*      TYPOC                      ;;CALL FOR TYPEOUT

033062  017646  000000                  $TYPOS: MOV     @(SP),-(SP)         ;;PICKUP THE MODE
033066  116637  000001  033305                  MOVB    1(SP),$0FILL        ;;LOAD ZERO FILL SWITCH
033074  112637  033307                          MOVB    (SP)+,$OMODE+1      ;;NUMBER OF DIGITS TO TYPE
033100  062716  000002                          ADD     #2,(SP)             ;;ADJUST RETURN ADDRESS
033104  000406                                  BR      $TYPON
033106  112737  000001  033305          $TYPOC: MOVB    #1,$0FILL           ;;SET THE ZERO FILL SWITCH
033114  112737  000006  033307                  MOVB    #6,$OMODE+1         ;;SET FOR SIX(6) DIGITS
033122  112737  000005  033304          $TYPON: MOVB    #5,$OCNT            ;;SET THE ITERATION COUNT
033130  010346                                  MOV     R3,-(SP)            ;;SAVE R3
033132  010446                                  MOV     R4,-(SP)            ;;SAVE R4
033134  010546                                  MOV     R5,-(SP)            ;;SAVE R5
033136  113704  033307                          MOVB    $OMODE+1,R4         ;;GET THE NUMBER OF DIGITS TO TYPE
033142  005404                                  NEG     R4
033144  062704  000006                          ADD     #6,R4               ;;SUBTRACT IT FOR MAX. ALLOWED
033150  110437  033306                          MOVB    R4,$OMODE           ;;SAVE IT FOR USE
033154  113704  033305                          MOVB    $0FILL,R4           ;;GET THE ZERO FILL SWITCH
033160  016605  000012                          MOV     12(SP),R5           ;;PICKUP THE INPUT NUMBER
033164  005003                                  CLR     R3                  ;;CLEAR THE OUTPUT WORD
033166  006105                          1$:     ROL     R5                  ;;ROTATE MSB INTO "C"
033170  000404                                  BR      3$                  ;;GO DO MSB
033172  006105                          2$:     ROL     R5                  ;;FORM THIS DIGIT
033174  006105                                  ROL     R5
033176  006105                                  ROL     R5
033200  010503                                  MOV     R5,R3
033202  006103                          3$:     ROL     R3                  ;;GET LSB OF THIS DIGIT
033204  105337  033306                          DECB    $OMODE              ;;TYPE THIS DIGIT?
033210  100016                                  BPL     7$                  ;;BR IF NO
033212  042703  177770                          BIC     #177770,R3          ;;GET RID OF JUNK
033216  001002                                  BNE     4$                  ;;TEST FOR 0
033220  005704                                  TST     R4                  ;;SUPPRESS THIS 0?
033222  001403                                  BEQ     5$                  ;;BR IF YES
033224  005204                          4$:     INC     R4                  ;;DON'T SUPPRESS ANYMORE 0'S
```

```
        033226  052703  000060                    BIS     #'0,R3            ;;MAKE THIS DIGIT ASCII
        033232  052703  000040          5$:       BIS     #' ,R3            ;;MAKE ASCII IF NOT ALREADY
        033236  110337  033302                    MOVB    R3,8$             ;;SAVE FOR TYPING
        033242  104401  033302                    TYPE    ,8$               ;;GO TYPE THIS DIGIT
        033246  105337  033304          7$:       DECB    $OCNT             ;;COUNT BY 1
        033252  003347                            BGT     2$                ;;BR IF MORE TO DO
        033254  002402                            BLT     6$                ;;BR IF DONE
        033256  005204                            INC     R4                ;;INSURE LAST DIGIT ISN'T A BLANK
        033260  000744                            BR      2$                ;;GO DO THE LAST DIGIT
        033262  012605                  6$:       MOV     (SP)+,R5          ;;RESTORE R5
        033264  012604                            MOV     (SP)+,R4          ;;RESTORE R4
        033266  012603                            MOV     (SP)+,R3          ;;RESTORE R3
        033270  016666  000002  000004            MOV     2(SP),4(SP)       ;;SET THE STACK FOR RETURNING
        033276  012616                            MOV     (SP)+,(SP)
        033300  000002                            RTI                       ;;RETURN
        033302     000                  8$:       .BYTE   0                 ;;STORAGE FOR ASCII DIGIT
        033303     000                            .BYTE   0                 ;;TERMINATOR FOR TYPE ROUTINE
        033304     000                  $OCNT:    .BYTE   0                 ;;OCTAL DIGIT COUNTER
        033305     000                  $OFILL:   .BYTE   0                 ;;ZERO FILL SWITCH
        033306  000000                  $OMODE:   .WORD   0                 ;;NUMBER OF DIGITS TO TYPE
```

```
                                          .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

                                          ;;*******************************************************************
                                          ;;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
                                          ;;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
                                          ;;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
                                          ;;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
                                          ;;*REPLACED WITH SPACES.
                                          ;;*CALL:
                                          ;;*       MOV     NUM,-(SP)               ;;PUT THE BINARY NUMBER ON THE STACK
                                          ;;*       TYPDS                           ;;GO TO THE ROUTINE

      033310                              $TYPDS:
      033310  010046                              MOV     R0,-(SP)                ;;PUSH R0 ON STACK
      033312  010146                              MOV     R1,-(SP)                ;;PUSH R1 ON STACK
      033314  010246                              MOV     R2,-(SP)                ;;PUSH R2 ON STACK
      033316  010346                              MOV     R3,-(SP)                ;;PUSH R3 ON STACK
      033320  010546                              MOV     R5,-(SP)                ;;PUSH R5 ON STACK
      033322  012746  020200                      MOV     #20200,-(SP)            ;;SET BLANK SWITCH AND SIGN
      033326  016605  000020                      MOV     20(SP),R5               ;;GET THE INPUT NUMBER
      033332  100004                              BPL     1$                      ;;BR IF INPUT IS POS.
      033334  005405                              NEG     R5                      ;;MAKE THE BINARY NUMBER POS.
      033336  112766  000055  000001              MOVB    #'-,1(SP)               ;;MAKE THE ASCII NUMBER NEG.
      033344  005000                      1$:     CLR     R0                      ;;ZERO THE CONSTANTS INDEX
      033346  012703  033524                      MOV     #$DBLK,R3               ;;SETUP THE OUTPUT POINTER
      033352  112723  000040                      MOVB    #' ,(R3)+               ;;SET THE FIRST CHARACTER TO A BLANK
      033356  005002                      2$:     CLR     R2                      ;;CLEAR THE BCD NUMBER
      033360  016001  033514                      MOV     $DTBL(R0),R1            ;;GET THE CONSTANT
      033364  160105                      3$:     SUB     R1,R5                   ;;FORM THIS BCD DIGIT
      033366  002402                              BLT     4$                      ;;BR IF DONE
      033370  005202                              INC     R2                      ;;INCREASE THE BCD DIGIT BY 1
      033372  000774                              BR      3$
      033374  060105                      4$:     ADD     R1,R5                   ;;ADD BACK THE CONSTANT
      033376  005702                              TST     R2                      ;;CHECK IF BCD DIGIT=0
      033400  001002                              BNE     5$                      ;;FALL THROUGH IF 0
      033402  105716                              TSTB    (SP)                    ;;STILL DOING LEADING 0'S?
      033404  100407                              BMI     7$                      ;;BR IF YES
      033406  106316                      5$:     ASLB    (SP)                    ;;MSD?
      033410  103003                              BCC     6$                      ;;BR IF NO
      033412  116663  000001  177777              MOVB    1(SP),-1(R3)            ;;YES--SET THE SIGN
      033420  052702  000060              6$:     BIS     #'0,R2                  ;;MAKE THE BCD DIGIT ASCII
      033424  052702  000040              7$:     BIS     #' ,R2                  ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
      033430  110223                              MOVB    R2,(R3)+                ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
      033432  005720                              TST     (R0)+                   ;;JUST INCREMENTING
      033434  020027  000010                      CMP     R0,#10                  ;;CHECK THE TABLE INDEX
      033440  002746                              BLT     2$                      ;;GO DO THE NEXT DIGIT
      033442  003002                              BGT     8$                      ;;GO TO EXIT
      033444  010502                              MOV     R5,R2                   ;;GET THE LSD
      033446  000764                              BR      6$                      ;;GO CHANGE TO ASCII
      033450  105726                      8$:     TSTB    (SP)+                   ;;WAS THE LSD THE FIRST NON-ZERO?
      033452  100003                              BPL     9$                      ;;BR IF NO
      033454  116663  177777  177776              MOVB    -1(SP),-2(R3)           ;;YES--SET THE SIGN FOR TYPING
      033462  105013                      9$:     CLRB    (R3)                    ;;SET THE TERMINATOR
      033464  012605                              MOV     (SP)+,R5                ;;POP STACK INTO R5
      033466  012603                              MOV     (SP)+,R3                ;;POP STACK INTO R3
      033470  012602                              MOV     (SP)+,R2                ;;POP STACK INTO R2
      033472  012601                              MOV     (SP)+,R1                ;;POP STACK INTO R1
```

```
033474  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
033476  104401  033524                  TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
033502  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
033510  012616                          MOV     (SP)+,(SP)
033512  000002                          RTI                     ;;RETURN TO USER
033514  023420                  $DTBL:  10000.
033516  001750                          1000.
033520  000144                          100.
033522  000012                          10.
033524                          $DBLK:  .BLKW   4
```

```
                                .SBTTL  RANDOM NUMBER GENERATOR ROUTINE

                                ;;*******************************************************************
                                ;*THIS ROUTINE IS A DOUBLE PRECISION PSEUDO RANDOM NUMBER GENERATOR
                                ;*WITH A RANGE OF 0 TO 2(+33)-1.
                                ;*CALL:
                                ;*      JSR     PC,$RAND                ;;CALL THE ROUTINE
                                ;*      RETURN                          ;;RETURN HERE THE RANDOM
                                ;*                                      ;;NUMBER WILL BE IN
                                ;*                                      ;;$HINUM,$LONUM

033534                          $RAND:
033534  010046                          MOV     R0,-(SP)                ;;PUSH R0 ON STACK
033536  010146                          MOV     R1,-(SP)                ;;PUSH R1 ON STACK
033540  010246                          MOV     R2,-(SP)                ;;PUSH R2 ON STACK
033542  013700  033634                  MOV     $LONUM,R0               ;;SET R0 WITH LOW
033546  013701  033632                  MOV     $HINUM,R1               ;;SET R1 WITH HIGH
033552  012702  177771                  MOV     #-7,R2                  ;;SET SHIFT COUNT
033556  006300                  1$:     ASL     R0                      ;;SHIFT R0 LEFT AND
033560  006101                          ROL     R1                      ;;ROTATE CARRY INTO R1 AND
033562  005202                          INC     R2                      ;;CHECK FOR DONE
033564  001374                          BNE     1$                      ;;CONTINUE SHIFT LOOP
033566  063700  033634                  ADD     $LONUM,R0               ;;ADD NUMBER TO MAKE X 129
033572  005501                          ADC     R1                      ;;PROPOGATE CARRY
033574  063701  033632                  ADD     $HINUM,R1               ;;ADD NUMBER TO MAKE X 129
033600  062700  001057                  ADD     #1057,R0                ;;ADD LOW CONSTANT
033604  005501                          ADC     R1                      ;;PROPOGATE CARRY
033606  062701  047401                  ADD     #47401,R1               ;;ADD HIGH CONSTANT
033612  010037  033634                  MOV     R0,$LONUM               ;;SAVE R0
033616  010137  033632                  MOV     R1,$HINUM               ;;SAVE R1
033622  012602                          MOV     (SP)+,R2                ;;POP STACK INTO R2
033624  012601                          MOV     (SP)+,R1                ;;POP STACK INTO R1
033626  012600                          MOV     (SP)+,R0                ;;POP STACK INTO R0
033630  000207                          RTS     PC                      ;;RETURN
033632  176543                  $HINUM: .WORD   176543
033634  123456                  $LONUM: .WORD   123456
```

```
                                        .SBTTL  SAVE AND RESTORE R0-R5 ROUTINES

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

        033636                          $SAVREG:
        033636  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
        033640  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
        033642  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
        033644  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
        033646  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
        033650  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
        033652  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PS OF MAIN FLOW
        033656  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PC OF MAIN FLOW
        033662  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PS OF CALL
        033666  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PC OF CALL
        033672  000002                          RTI

                                        ;*RESTORE R0-R5
                                        ;*CALL:
                                        ;*       RESREG
        033674                          $RESREG:
        033674  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF CALL
        033700  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF CALL
        033704  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
        033710  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
        033714  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
        033716  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
        033720  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
        033722  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
        033724  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
        033726  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
        033730  000002                          RTI
```

```
                                .SBTTL  DOUBLE LENGTH BINARY TO DECIMAL ASCII CONVERT ROUTINE

                                ;;******************************************************************
                                ;*THIS ROUTINE WILL CONVERT A 32-BIT BINARY NUMBER TO AN UNSIGNED
                                ;*DECIMAL (ASCII) NUMBER. THE SIGN OF THE BINARY NUMBER MUST BE
                                ;*POSITIVE.
                                ;*CALL
                                ;*      MOV     #PNTR,-(SP)       ;;POINTER TO LOW WORD OF BINARY NUMBER
                                ;*      JSR     PC,@#$DB2D
                                ;*      RETURN                    ;;THE FIRST ADDRESS OF ASCIZ
                                                                  ;;IS ON THE STACK


033732  104412                  $DB2D:  SAVREG                    ;;SAVE REGISTERS
033734  016602  000002                  MOV     2(SP),R2          ;;PICKUP THE DATA POINTER
033740  012700  034112                  MOV     #$DECVL,R0        ;;GET ADDRESS OF "$DECVL" STRING
033744  010066  000002                  MOV     R0,2(SP)          ;;PUT ADDRESS OF ASCIZ STRING ON STACK
033750  012201                          MOV     (R2)+,R1          ;;PICKUP THE BINARY NUMBER
033752  012202                          MOV     (R2)+,R2
033754  012737  000012  034030          MOV     #10.,4$           ;;SET UP TO DO 10 CONVERSIONS
033762  012704  034042                  MOV     #$TNPWR,R4        ;;ADDRESS OF TEN POWER
033766  012705  034044                  MOV     #$TNPWR+2,R5
033772  005003                  1$:     CLR     R3                ;;CLEAR PARTIAL
033774  161401                  2$:     SUB     (R4),R1           ;;SUBTRACT TEN POWER
033776  005602                          SBC     R2
034000  161502                          SUB     (R5),R2
034002  002402                          BLT     3$                ;;BR IF TEN POWER TO LARGE
034004  005203                          INC     R3                ;;ADD 1 TO PARTIAL
034006  000772                          BR      2$                ;;LOOP
034010  062401                  3$:     ADD     (R4)+,R1          ;;RESTORE SUBTRACTED VALUE
034012  005502                          ADC     R2
034014  062402                          ADD     (R4)+,R2
034016  022525                          CMP     (R5)+,(R5)+       ;;MOVE TO NEXT TEN POWER
034020  052703  000060                  BIS     #'0,R3            ;;CHANGE PARTIAL TO ASCII
034024  110320                          MOVB    R3,(R0)+          ;;SAVE IT
034026  005327                          DEC     (PC)+             ;;DONE?
034030  000000                  4$:     .WORD   0
034032  001357                          BNE     1$                ;;BR IF NO
034034  105020                          CLRB    (R0)+             ;;TERMINATOR
034036  104413                          RESREG                    ;;RESTORE REGISTERS
034040  000207                          RTS     PC                ;;RETURN
034042  145000                  $TNPWR: 145000                    ;;1.0E09
034044  035632                          35632
034046  160400                          160400                    ;;1.0E08
034050  002765                          2765
034052  113200                          113200                    ;;1.0E07
034054  000230                          230
034056  041100                          041100                    ;;1.0E06
034060  000017                          17
034062  103240                          103240                    ;;1.0E05
034064  000001                          1
034066  023420                          23420                     ;;1.0E04
034070  000000                          0
034072  001750                          1750                      ;;1.0E03
034074  000000                          0
034076  000144                          144                       ;;1.0E02
034100  000000                          0
```

```
        034102  000012                  12                          ;;1.0E01
        034104  000000                  0
        034106  000001                  1                           ;;1.0E00
        034110  000000                  0
        034112                  $DECVL: .BLKB   12.                 ;;RESERVE STORAGE FOR ASCIZ STRING
```

```
                                .SBTTL  DOUBLE LENGTH BINARY TO OCTAL ASCII CONVERT ROUTINE

                                ;;*******************************************************************
                                ;*THIS ROUTINE WILL CONVERT A 32-BIT UNSIGNED BINARY NUMBER TO AN
                                ;*UNSIGNED OCTAL ASCIZ NUMBER.
                                ;*CALL
                                ;*      MOV     #PNTR,-(SP)     ;;POINTER TO LOW WORD OF BINARY NUMBER
                                ;*      JSR     PC,@#$DB20      ;;CALL THE ROUTINE
                                ;*      RETURN                  ;;THE ADDRESS OF THE FIRST ASCIZ CHAR. IS ON THE STACK

034126  104412                  $DB20:  SAVREG                  ;;SAVE ALL REGISTERS
034130  016601  000002                  MOV     2(SP),R1        ;;PICKUP THE POINTER TO LOW WORD
034134  012705  034245                  MOV     #$OCTVL+13.,R5  ;;POINTER TO DATA TABLE
034140  012704  000014                  MOV     #12.,R4         ;;DO ELEVEN CHARACTERS
034144  012703  177770                  MOV     #^C7,R3         ;;MASK
034150  012100                          MOV     (R1)+,R0        ;;LOWER WORD
034152  012101                          MOV     (R1)+,R1        ;;HIGH WORD
034154  005002                          CLR     R2              ;;TERMINATOR
034156  110245                  1$:     MOVB    R2,-(R5)        ;;PUT CHARACTER IN DATA TABLE
034160  010002                          MOV     R0,R2           ;;GET THIS DIGIT
034162  005304                          DEC     R4              ;;COUNT THIS CHARACTER
034164  003007                          BGT     3$              ;;BR IF NOT THE LAST DIGIT
034166  001405                          BEQ     2$              ;;BR IF IT IS THE LAST DIGIT
034170  005205                          INC     R5              ;;ALL DIGITS DONE-ADJUST POINTER FOR FIRST
034172  010566  000002                  MOV     R5,2(SP)        ;;ASCIZ CHAR. & PUT IT ON THE STACK
034176  104413                          RESREG                  ;;RESTORE ALL REGISTERS
034200  000207                          RTS     PC              ;;RETURN TO USER
034202  006203                  2$:     ASR     R3              ;;POSITION THE MASK FOR THE LAST DIGIT
034204  006001                  3$:     ROR     R1              ;;POSITION THE BINARY NUMBER FOR
034206  006000                          ROR     R0              ;;      THE NEXT OCTAL DIGIT
034210  006001                          ROR     R1
034212  006000                          ROR     R0
034214  006001                          ROR     R1
034216  006000                          ROR     R0
034220  040302                          BIC     R3,R2           ;;MASK OUT ALL JUNK
034222  062702  000060                  ADD     #'0,R2          ;;MAKE THIS CHAR. ASCII
034226  000753                          BR      1$              ;;GO PUT IT IN THE DATA TABLE
034230                          $OCTVL: .BLKB   14.             ;;RESERVE DATA TABLE
```

```
                                        .SBTTL  TRAP DECODER

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

   034246  010046                       $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
   034250  016600  000002                       MOV     2(SP),R0        ;;GET TRAP ADDRESS
   034254  005740                               TST     -(R0)           ;;BACKUP BY 2
   034256  111000                               MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
   034260  006300                               ASL     R0              ;;POSITION FOR INDEXING
   034262  016000  034302                       MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
   034266  000200                               RTS     R0              ;;GO TO ROUTINE

                                        ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO

   034270  011646                       $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
   034272  016666  000004  000002               MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
   034300  000002                               RTI                     ;;RESTORE THE PSW

                                        .SBTTL  TRAP TABLE

                                        ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
                                        ;*BY THE "TRAP" INSTRUCTION.

                                        ;       ROUTINE
                                        ;       -------
   034302  034270                       $TRPAD: .WORD   $TRAP2
   034304  032570                               $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
   034306  033106                               $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
   034310  033062                               $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
   034312  033122                               $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
   034314  033310                               $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)

   034316  030770                               $GTSWR  ;;CALL=GTSWR    TRAP+6(104406)  GET SOFT-SWR SETTING

   034320  030700                               $CKSWR  ;;CALL=CKSWR    TRAP+7(104407)  TEST FOR CHANGE IN SOFT-SWR
   034322  031242                               $RDCHR  ;;CALL=RDCHR    TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
   034324  031332                               $RDLIN  ;;CALL=RDLIN    TRAP+11(104411) TTY TYPEIN STRING ROUTINE
   034326  033636                               $SAVREG ;;CALL=SAVREG   TRAP+12(104412) SAVE R0-R5 ROUTINE
   034330  033674                               $RESREG ;;CALL=RESREG   TRAP+13(104413) RESTORE R0-R5 ROUTINE
 2 034332  027770                               $DSPLY  ;;CALL=DISPLY   TRAP+14(104414) ROUTINE TO TYPE ERROR MESSAGES
 3         000032                       $TERM=.-$TRPAD
 4
```

```

                                        .SBTTL  SINGLE/DUAL PORT RH11/RP04/5/6 DRIVER (REV 1.0)

                                        ;COPYRIGHT (C) 1976,1979
                                        ;DIGITAL EQUIPMENT CORP.
                                        ;MAYNARD, MA 01754

                                        ;STORAGE FOR RPDS1, RPER1, RPER2, AND RPER3 ON AN ERROR "2"
                                                ;RPERRS    = RPDS1
                                                ;RPERRS+2 = RPER1
                                                ;RPERRS+4 = RPER2
                                                ;RPERRS+6 = RPER3

 034334   000000  000000  000000  RPERRS: .WORD   0,0,0,0

                                        ;TABLE OF DRIVE ACTIVE INDICATORS (DRVACT=8 BYTES)
                                                ;DRVACT=0 IF DRIVE IS IDLE
                                                ;DRVACT>0 IF DRIVE IS ACTIVE WITH A COMMAND
                                                ;DRVACT<0 IF DRIVE IS ACTIVE WITH AN ERROR RECOVERY OPERATION

 034344      000                        DRVACT: .BYTE   0               ;DRIVE 0
 034345      000                                .BYTE   0               ;DRIVE 1
 034346      000                                .BYTE   0               ;DRIVE 2
 034347      000                                .BYTE   0               ;DRIVE 3
 034350      000                                .BYTE   0               ;DRIVE 4
 034351      000                                .BYTE   0               ;DRIVE 5
 034352      000                                .BYTE   0               ;DRIVE 6
 034353      000                                .BYTE   0               ;DRIVE 7

                                        ;TABLE OF DRIVE STATUS INDICATORS (DRVSTA=8 BYTES)
                                                ;DRVSTA=0 IF DRIVE IS OFFLINE OR NONEXSITENT
                                                ;DRVSTA>0 IF DRIVE IS ONLINE
                                                ;DRVSTA<0 IF DRIVE IS UNSAFE

 034354      000                        DRVSTA: .BYTE   0               ;DRIVE 0
 034355      000                                .BYTE   0               ;DRIVE 1
 034356      000                                .BYTE   0               ;DRIVE 2
 034357      000                                .BYTE   0               ;DRIVE 3
 034360      000                                .BYTE   0               ;DRIVE 4
 034361      000                                .BYTE   0               ;DRIVE 5
 034362      000                                .BYTE   0               ;DRIVE 6
 034363      000                                .BYTE   0               ;DRIVE 7

                                        ;TABLE OF DRIVE TYPES (DRVTYP=8 BYTES)
                                                ;DRVTYP=0 IF DRIVE IS NONEXISTENT (DRVSTA=0, ALSO)
                                                ;DRVTYP=1 IF DRIVE IS RP04
                                                ;DRVTYP=2 IF DRIVE IS RP05
                                                ;DRVTYP=4 IF DRIVE IS RP06
                                                ;DRVTYP=-1 IF NOT RP04/5/6

 034364      000                        DRVTYP: .BYTE   0               ;DRIVE 0
 034365      000                                .BYTE   0               ;DRIVE 1
 034366      000                                .BYTE   0               ;DRIVE 2
 034367      000                                .BYTE   0               ;DRIVE 3
 034370      000                                .BYTE   0               ;DRIVE 4
 034371      000                                .BYTE   0               ;DRIVE 5
 034372      000                                .BYTE   0               ;DRIVE 6
```

```
034373       000                    .BYTE   0               ;DRIVE 7

                            ;TABLE OF DUAL PORT INITIALIZATION INDICATORS
                                    ;DPINT=0 IF INITIALIZATION IS NOT ACTIVE ON THE DRIVE
                                    ;DPINT<0 IF INITIALIZATION IS IN PROGRESS

034374       000            DPINT:  .BYTE   0               ;DRIVE 0
034375       000                    .BYTE   0               ;DRIVE 1
034376       000                    .BYTE   0               ;DRIVE 2
034377       000                    .BYTE   0               ;DRIVE 3
034400       000                    .BYTE   0               ;DRIVE 4
034401       000                    .BYTE   0               ;DRIVE 5
034402       000                    .BYTE   0               ;DRIVE 6
034403       000                    .BYTE   0               ;DRIVE 7

                            ;TABLE OF PENDING DUAL PORT REQUESTS
                                    ;DPRQS=0 IF THAT A DUAL PORT REQUEST IS NOT PENDING FOR THAT DRIVE
                                    ;DPRQS<0 IF THAT A DUAL PORT REQUEST IS PENDING FOR THAT DRIVE

034404       000            DPRQS:  .BYTE   0               ;DRIVE 0
034405       000                    .BYTE   0               ;DRIVE 1
034406       000                    .BYTE   0               ;DRIVE 2
034407       000                    .BYTE   0               ;DRIVE 3
034410       000                    .BYTE   0               ;DRIVE 4
034411       000                    .BYTE   0               ;DRIVE 5
034412       000                    .BYTE   0               ;DRIVE 6
034413       000                    .BYTE   0               ;DRIVE 7

                            ;TRANSFER WAIT FLAG (TRNSWT=1 WORD)
                                    ;THIS IS A ONE WORD QUEUE. IT WILL CONTAIN THE ADDRESS OF
                                    ;''DPB'' OF THE I/O OPERATION.

034414    000000            TRNSWT: .WORD   0

                            ;SEARCH WAIT KEYS (SRCHWT=1 WORD)
                                    ;THIS IS A ONE WORD QUEUE THAT WILL CONTAIN A KEY FOR EACH OF
                                    ;THE DRIVES THAT ARE PERFORMING A SEARCH COMMAND FOR THE I/O
                                    ;REQUEST THAT IS AT THE TOP OF THEIR REQUEST QUEUE.
                                    ;EACH DRIVE IS ASSIGNED ONE BIT, STARTING AT BIT00 FOR DRIVE 0.

034416    000000            SRCHWT: .WORD   0


                            ;RP04/5/6 DRIVER ACTIVE FLAG (ACTDRV=1 BYTE)
                                    ;ACTDRV=0 IF DRIVER IS INACTIVE
                                    ;ACTDRV>0 IF DRIVER IS ACTIVE

034420       000            ACTDRV: .BYTE   0

                            ;SOFTWARE TIMER ROUTINE ACTIVE FLAG (ACTSTR=1 BYTE)
                                    ;ACTSTR=0 IF SOFTWARE TIMER ROUTINE IS INACTIVE
                                    ;ACTSTR>0 IF SOFTWARE TIMER ROUTINE IS ACTIVE

034421       000            ACTSTR: .BYTE   0

                            ;UNLOAD FLAG (ULDFLG=8 BYTES)
                                    ;ULDFLG=0 IF NO UNLOAD COMMAND
```

```
                                        ;ULDFLG>0 IF UNLOAD COMMAND IN PROGRESS
                                        ;ULDFLG<0 IF UNLOAD COMMAND IN WAIT QUEUE

034422        000               ULDFLG: .BYTE   0               ;DRIVE 0
034423        000                       .BYTE   0               ;DRIVE 1
034424        000                       .BYTE   0               ;DRIVE 2
034425        000                       .BYTE   0               ;DRIVE 3
034426        000                       .BYTE   0               ;DRIVE 4
034427        000                       .BYTE   0               ;DRIVE 5
034430        000                       .BYTE   0               ;DRIVE 6
034431        000                       .BYTE   0               ;DRIVE 7

                                ;LOOK AHEAD COUNT (LACNT=8 BYTES)
                                        ;LACNT WILL INDICATE THE NUMBER OF LOOK AHEADS PERFORMED

034432        000               LACNT:  .BYTE   0               ;DRIVE 0
034433        000                       .BYTE   0               ;DRIVE 1
034434        000                       .BYTE   0               ;DRIVE 2
034435        000                       .BYTE   0               ;DRIVE 3
034436        000                       .BYTE   0               ;DRIVE 4
034437        000                       .BYTE   0               ;DRIVE 5
034440        000                       .BYTE   0               ;DRIVE 6
034441        000                       .BYTE   0               ;DRIVE 7

                                ;SAVE REGISTERS FLAG (SAVEFG =1 WORD)
                                        ;SAVEFG <0 IF SAVE THE RH11/RP04/5/6 REGISTERS WHEN THE
                                        ;OPERATION IS COMPLETED AS PER (DPB+14).
                                        ;SAVEFG=0 IF SAVE THE RH11/RP04/5/6 REGISTERS, AS PER
                                        ;(DPB+14), AFTER AN ERROR.

034442  000000                  SAVEFG: .WORD   0

                                ;SEEK FLAG (SEEKFG=1 WORD)
                                        ;SEEKFG=0 IF WHEN THE DISK ADDRESS ISN'T IN THE WINDOW
                                        ;FOR A DATA TRANSFER START A SEARCH COMMAND
                                        ;SEEKFG<0 IF DATA TRANSFER WILL DO IMPLIED SEEKS,
                                        ;DISREGARD THE WINDOW

034444  000000                  SEEKFG: .WORD   0

                                ;TIMEOUT TABLE (TIMER=8 WORDS)
                                        ;THIS TABLE CONTAINS THE TIME ALLOWED FOR AN OPERATION

034446  177777                  TIMER:  .WORD   -1      ;DRIVE 0
034450  177777                          .WORD   -1      ;DRIVE 1
034452  177777                          .WORD   -1      ;DRIVE 2
034454  177777                          .WORD   -1      ;DRIVE 3
034456  177777                          .WORD   -1      ;DRIVE 4
034460  177777                          .WORD   -1      ;DRIVE 5
034462  177777                          .WORD   -1      ;DRIVE 6
034464  177777                          .WORD   -1      ;DRIVE 7

                                ;DATA TRANSFER UNDERWAY INDICATOR (DTUW=1 WORD)
                                        ;DTUW<0 IF NO DATA TRANSFER UNDERWAY
                                        ;DTUW=+N (WHERE N=0 TO 7) IMPLIES DATA TRANSFER UNDERWAY ON DRIVE N

034466  177777                  DTUW:   .WORD   -1
```

```

                                ;ATTENTION BITS TABLE (ATABIT=8 BYTES)
                                        ;THIS TABLE CONTAINS THE CORRESPONDING BIT TO EACH DRIVES
                                        ;ATTENTION BIT

034470     001                  ATABIT: .BYTE   1      ;DRIVE 0
034471     002                          .BYTE   2      ;DRIVE 1
034472     004                          .BYTE   4      ;DRIVE 2
034473     010                          .BYTE   10     ;DRIVE 3
034474     020                          .BYTE   20     ;DRIVE 4
034475     040                          .BYTE   40     ;DRIVE 5
034476     100                          .BYTE   100    ;DRIVE 6
034477     200                          .BYTE   200    ;DRIVE 7

                                ;RP04/5/6 TO RH11 'MASSBUS CONTROL BUS PARITY ERRORS'' (MCPE) ALLOWED BEFORE
                                ;CALLING IT FATAL (MCPEMX=1 WORD)

034500  000003                  MCPEMX: .WORD   3

                                ;STORAGE FOR RPADR (THE FIRST ADDRESS (776700) OF THE RH11/RP04/5/6),
                                ;RPVEC (THE VECTOR ADDRESS (254)), AND RPVEC+2 (THE BR LEVEL (5)).

034502  176700                  RPADR:  .WORD   176700
034504  000254  000240          RPVEC:  .WORD   254,5*32.

                                ;MAXIMUM NUMBER OF LOOK AHEADS ALLOWED IS 4 (MXLACT=1 WORD)

034510  000004                  MXLACT: .WORD   4
                                ;MAXIMUM DELTA DELAY IS 8 SECTORS (MXDLTA=1 WORD)

034512  001000                  MXDLTA: .WORD   8.*64.
                                ;MINIMUM DELTA DELAY IS 2 SECTORS (MNDLTA=1 WORD)

034514  000200                  MNDLTA: .WORD   2*64.
                                ;MAXIMUM SEARCH FOR I/O WINDOW IS 5 SECTORS (MXWNDW=1 WORD)

034516  000005                  MXWNDW: .WORD   5

                                ;DEFINITIONS OF THE RH11/RP04/5/6 ADDRESS INDEXES

        000000                  RPCS1=0                        ;CONTROL AND STATUS REGISTER #1 (DRIVE REG. 00)
        000002                  RPWC=2                         ;WORD COUNT REGISTER (NOT A DRIVE REG)
        000004                  RPBA=4                         ;UNIBUS ADDRESS REGISTER (NOT A DRIVE REG)
        000006                  RPDA=6                         ;DESIRED SECTOR/TRACK ADDRESS REGISTER (DRIVE REG. 05)
        000010                  RPCS2=10                       ;CONTROL AND STATUS REGISTER #2 (NOT A DRIVE REG)
        000012                  RPDS1=12                       ;DRIVE STATUS REGISTER (DRIVE REG 01)
        000014                  RPER1=14                       ;ERROR REGISTER #1 (DRIVE REG. 02)
        000016                  RPAS=16                        ;ATTENTION SUMMARY PSEUDO REGISTER (DRIVE REG. 04)
        000020                  RPLA=20                        ;LOOK AHEAD REGISTER (DRIVE REG. 07)
        000022                  RPDB=22                        ;DATA BUFFER REGISTER (NOT A DRIVE REG.)
        000024                  RPMR=24                        ;MAINTAINABILITY REGISTER (DRIVE REG. 03)
        000026                  RPDT=26                        ;DRIVE TYPE REGISTER (DRIVE REG. 06)
        000030                  RPSN=30                        ;SERIAL NUMBER REGISTER (DRIVE REG. 10)
        000032                  RPOF=32                        ;OFFSET REGISTER (DRIVE REG. 11)
        000034                  RPCA=34                        ;DESIRED CYLINDER ADDRESS REGISTER (DRIVE REG. 12)
        000036                  RPCC=36                        ;CURRENT CYLINDER ADDRESS REGISTER (DRIVE REG. 13)
        000040                  RPER2=40                       ;ERROR REGISTER #2 (DRIVE REG. 14)
        000042                          RPER3=42                                ;ERROR REGISTER #3 (DRIVE REG. 15)
        000044                          RPEC1=44                                ;ECC POSITION REGISTER (DRIVE REG. 16)
        000046                          RPEC2=46                                ;ECC PATTERN REGISTER (DRIVE REG.  17)

                                        ;RH11/RP04/5/6 DRIVER INITIALIZATION CODE
                                                ;THIS ROUTINE WILL DETERMINE WHICH RP04/5/6 DRIVES ARE
                                                ;AVAILABLE FOR TESTING AND SET THE DRVSTA INDICATOR
                                                ;TO THE PROPER STATE FOR EACH DRIVE.
                                                ;NOTE: THIS ROUTINE CALLS DRVINT
                                        ;
                                        ;CALL
                                        ;
                                        ;       JSR     PC,RPINIT
                                        ;       RETURN
                                        ;
                                        ;NOTE: THE 'P' OR 'L' CLOCK MUST BE STARTED
                                        ;
034520  104412                          RPINIT: SAVREG                          ;SAVE R0 - R5
034522  013746  177776                          MOV     @#PS,-(SP)              ;SAVE THE PRESENT PROCESSOR STATUS
034526  012737  000240  177776                  MOV     #<5*32.>,@#PS           ;CHANGE THE PRIORITY TO 5
034534  004737  042652                          JSR     PC,CLRQUE               ;CLEAR ALL REQUEST QUEUES
034540  012701  034334                          MOV     #RPERRS,R1              ;FIRST ADDRESS TO BE CLEARED
034544  012702  034444                          MOV     #SEEKFG,R2              ;LAST ADDRESS TO BE CLEARED
034550  005021                          1$:     CLR     (R1)+                   ;CLEAR
034552  020102                                  CMP     R1,R2                   ;ARE WE DONE?
034554  101775                                  BLOS    1$                      ;BRANCH IF NO
034556  012702  034466                          MOV     #DTUW,R2                ;LAST ADDRESS
034562  012721  177777                  2$:     MOV     #-1,(R1)+               ;INITIALIZE
034566  020102                                  CMP     R1,R2                   ;DONE?
034570  101774                                  BLOS    2$                      ;LOOP IF NO
034572  005037  034354                          CLR     DRVSTA                  ;SET ALL DRIVES TO OFFLINE
034576  005037  034356                          CLR     DRVSTA+2
034602  005037  034360                          CLR     DRVSTA+4
034606  005037  034362                          CLR     DRVSTA+6
034612  013703  034504                          MOV     RPVEC,R3                ;SETUP THE RH11/RP04/5/6 VECTOR
034616  012723  037434                          MOV     #ISR,(R3)+
034622  013713  034506                          MOV     RPVEC+2,(R3)
034626  013704  034502                          MOV     RPADR,R4                ;FIRST ADDRESS OF RH11/RP04
034632  012764  000040  000010                  MOV     #BIT05,RPCS2(R4)          ;MASSBUS INIT
034640  005001                                  CLR     R1                      ;START WITH DRIVE 0
034642  004037  034732                  3$:     JSR     R0,DRVINT               ;INIT THE DRIVE
034646  000401                                  BR      4$                      ;'DVA' NOT SET OR PARITY ERROR
034650  000402                                  BR      5$                      ;NORMAL RETURN
034652  105061  034354                  4$:     CLRB    DRVSTA(R1)              ;SET DRIVE STATUS TO OFFLINE
034656  005201                          5$:     INC     R1                      ;GO TO NEXT DRIVE
034660  042701  177770                          BIC     #^C7,R1                 ;MASK OUT UNUSED BITS
034664  001366                                  BNE     3$                      ;BR IF MORE DRIVES TO GO
034666  012701  000007                          MOV     #7,R1                   ;START WITH DRIVE 7
034672  005037  177776                          CLR     @#PS                    ;CLEAR THE PROCESSOR STATUS
034676  105761  034374                  6$:     TSTB    DPINT(R1)               ;WAITING FOR DRIVE TO SWITCH PORTS ?
034702  001405                                  BEQ     8$                      ;BR NOT WAITING
034704  004737  042306                          JSR     PC,SET.IE               ;SET INTERRUPT
034710  105761  034374                  7$:     TSTB    DPINT(R1)               ;DRIVE SWITCHED PORTS ?
034714  001375                                  BNE     7$                      ;BR IF NOT
034716  005301                          8$:     DEC     R1                      ;GO TO THE NEXT DRIVE
034720  100366                                  BPL     6$                      ;CHECK NEXT DRIVE
034722  012637  177776                          MOV     (SP)+,@#PS              ;RESTORE THE PROCESSOR STATUS
```

```
034726  104413                            RESREG                      ;RESTORE R0 - R5
034730  000207                            RTS     PC                  ;BYE-BYE

                                ;DRIVE INITILIZATION ROUTINE
                                          ;THIS ROUTINE DETERMINES IF A DRIVE EXIST AND IF IT IS
                                          ;AN RP04/5/6. IF IT IS, A ''READ-IN PRESET'' IS ISSUED AND FMT22
                                          ;IS SET TO A ''1''. THEN MOL, DPR, DRY, AND VV ARE CHECKED TO
                                          ;INSURE THEY ARE ALL ON A ''1''. AND DEPENDING ON THEIR STATE,
                                          ;DRVSTA IS SET TO THE PROPER CONDITION.
                                ;CALL
                                ;         MOV     #DRVNUM,R1          ;DRIVE NUMBER TO R1
                                ;         MOV     RPADR,R4            ;UNIBUS ADDRESS OF RH11/RP04/5/6 (RPCS1)
                                ;         JSR     R0,DRVINT           ;CALLED BY A JSR
                                ;         RETURN1                     ;ERROR OCCURRED (PARITY)
                                ;         RETURN2                     ;NORMAL RETURN
                                ;

034752  010546                  DRVINT:   MOV     R5,-(SP)            ;SAVE R5
034734  105061  034354                    CLRB    DRVSTA(R1)          ;START DRIVE STATUS AS OFFLINE
034740  105061  034364                    CLRB    DRVTYP(R1)          ;CLEAR THE DRIVE TYPE INDICATOR
034744  105061  034422                    CLRB    ULDFLG(R1)          ;CLEAR THE UNLOAD FLAG
034750  010164  000010                    MOV     R1,RPCS2(R4)        ;SELECT A DRIVE
034754  112764  000111  000000            MOVB    #111,RPCS1(R4)      ;DO A DRIVE CLEAR COMMAND (& SEIZE DRIVE)
034762  032764  010000  000010            BIT     #BIT12,RPCS2(R4)    ;NONEXISTENT DRIVE?
034770  001403                            BEQ     1$                  ;NO---BRANCH
034772  004737  042306                    JSR     PC,SET.IE           ;GO SET ''IE'' WITHOUT A ''TRE''
034776  000543                            BR      6$                  ;LEAVE THIS ROUTINE
035000  105061  034354          1$:       CLRB    DRVSTA(R1)          ;SET DRIVE STATUS TO OFFLINE
035004  032764  004000  000000            BIT     #BIT11,RPCS1(R4)      ;SEE IF DRIVE AVAILABLE
035012  001537                            BEQ     7$                  ;BR IF DRIVE NOT AVAILABLE
035014  004037  041626                    JSR     R0,RD.RP            ;READ THE DRIVE TYPE REG.
035020  000026                            RPDT
035022  035332                            8$                          ;ERROR RETURN ADDRESS
035024  012605                            MOV     (SP)+,R5            ;PUT DRIVE TYPE IN R5
035026  112761  000001  034364            MOVB    #1,DRVTYP(R1)       ;SET RP04 INDICATOR
035034  022705  020020                    CMP     #20020,R5           ;IS IT A SINGLE PORT RP04?
035040  001431                            BEQ     2$                  ;BRANCH IF YES
035042  022705  024020                    CMP     #24020,R5           ;IS IT A DUAL PORT RP04?
035046  001426                            BEQ     2$                  ;BR IF YES
035050  112761  000002  034364            MOVB    #2,DRVTYP(R1)       ;SET RP05 INDICATOR
035056  022705  020021                    CMP     #20021,R5           ;SINGLE PORT RP05 ?
035062  001420                            BEQ     2$                  ;BR IF YES
035064  022705  024021                    CMP     #24021,R5           ;DUAL PORT RP05 ?
035070  001415                            BEQ     2$                  ;BR IF YES
035072  112761  000004  034364            MOVB    #4,DRVTYP(R1)       ;SET RP06 INDICATOR
035100  022705  020022                    CMP     #20022,R5           ;SINGLE PORT RP06 ?
035104  001407                            BEQ     2$                  ;BR IF YES
035106  022705  024022                    CMP     #24022,R5           ;DUAL PORT RP06 ?
035112  001404                            BEQ     2$                  ;BR IF YES
035114  112761  177777  034364            MOVB    #-1,DRVTYP(R1)      ;SET INDICATOR TO 'OTHER'
035122  000471                            BR      6$                  ;EXIT
035124  005737  035336          2$:       TST     TSTPGM              ;INHIBIT PROGRAMMABLE DRIVE?
035130  001010                            BNE     9$                  ;BRANCH IF NO
035132  032764  001000  000012            BIT     #BIT09, RPDS1(R4);IS DRIVE PROGRAMMABLE?
035140  001404                            BEQ     9$                  ;BRANCH IF NO
035142  152761  000010  034364            BISB    #BIT03, DRVTYP(R1);SET INDICATOR
035150  000456                            BR      6$                  ;EXIT
```

```
035152  010346                  9$:     MOV     R3,-(SP)                ;SAVE R3
035154  006301                          ASL     R1                      ;CREATE WORD INDEX
035156  016103  001740                  MOV     BLKADR(R1),R3           ;GET DPB START ADDRESS
035162  010563  000262                  MOV     R5,$RPDT(R3)            ;STORE DRIVE'S TYPE
035166  006201                          ASR     R1                      ;RESTORE DRIVE NUMBER
035170  012603                          MOV     (SP)+,R3                ;RESTORE R3
035172  012746  000121                  MOV     #121,-(SP)              ;DO A 'READ-IN-PRESET''
035176  004037  042002                  JSR     R0,WRT.RP
035202  000000                          RPCS1
035204  035332                          8$
035206  012746  010000                  MOV     #BIT12,-(SP)            ;SET FMT22=1
035212  004037  042002                  JSR     R0,WRT.RP
035216  000032                          RPOF
035220  035332                          8$
035222  004037  041626                  JSR     R0,RD.RP                ;READ RPDS1
035226  000012                          RPDS1
035230  035332                          8$
035232  012605                          MOV     (SP)+,R5                ;AND SAVE IT IN R5
035234  100015                          BPL     4$                      ;BRANCH IF ATA=0
035236  116164  034470  000016          MOVB    ATABIT(R1),RPAS(R4)  ;CLEAR ATTENTION BIT
035244  004037  041626                  JSR     R0,RD.RP                ;FIND OUT WHY ATA=1
035250  000014                          RPER1
035252  035332                          8$
035254  006126                          ROL     (SP)+                   ;IS IT UNSAFE?
035256  100004                          BPL     4$                      ;BR IF NOT
035260  112761  177777  034354          MOVB    #-1,DRVSTA(R1)          ;SET UNSAFE INDICATOR
035266  000407                          BR      6$                      ;EXIT
035270  005105                  4$:     COM     R5                      ;CHECK MOL, DPR, DRY, AND VV
035272  042705  167077                  BIC     #^C<BIT12!BIT08!BIT07!BIT06>,R5
035276  001003                          BNE     6$                      ;BRANCH IF MOL, DPR, DRY, OR VV IS CLEAR
035300  112761  000001  034354          MOVB    #1,DRVSTA(R1)           ;SET DRIVE STATUS TO ONLINE
035306  005720                  6$:     TST     (R0)+                   ;STEP OVER THE ERROR RETURN
035310  000410                          BR      8$                      ;EXIT
035312  006301                  7$:     ASL     R1                      ;CHANGE INDEX TO ADDRESS WORDS
035314  012761  003720  034446          MOV     #2000.,TIMER(R1)   ;START 2 SEC TIMER
035322  006201                          ASR     R1                      ;RESTORE R1
035324  105161  034374                  COMB    DPINT(R1)               ;SET PORT INITIALIZE INIDICATOR
035330  005720                          TST     (R0)+
035332  012605                  8$:     MOV     (SP)+,R5                ;RESTORE R5
035334  000200                          RTS     R0                      ;EXIT

                                ;TEST PROGRAMMABLE DRIVE FLAG (TSTPGM=1 WORD)
                                        ;THE FLAG WILL BE SET BY THE PROGRAM UNDER
                                        ;MANUFACTURING CONDITIONS (ACT,APT) AND
                                        ;CLEARED UNDER FIELD CONDITIONS(XXDP CHAIN,
                                        ;STANDALONE) WITH STARTING ADDRESS 200.
                                        ;THE FLAG WILL BE SET UNDER ALL CONDITIONS
                                        ;WITH STARTING ADDRESS 220.

035336  000000                  TSTPGM: .WORD   0         ;=1 TEST PROGRAMMABLE DRIVE

                                ;REQUEST PRE-PROCESSOR-HANDLES SUBSYSTEM REQUEST
                                ;
                                ;CALL
                                ;
                                ;       JSR     R0,@#RP04               ;CALL THE RP04/5/6 DRIVER
                                ;       PNTADR                          ;ADDRESS OF POINTER OF DRIVES PARAMETER BLOCK
```

```
                                        ;       RETURN1                         ;RETURN HERE IF QUEUE IS FULL
                                        ;       RETURN2                         ;RETURN HERE IF REQUEST IS IN QUEUE OR THERE
                                                                                ;IS AN ERROR CONDITION

035340  013746  177776          RP04:   MOV     @#PS,-(SP)              ;SAVE THE CALLING STATUS
035344  013737  034506  177776          MOV     RPVEC+2,@#PS            ;DON'T ALLOW ANY RP04/5/6 INTERRUPTS
035352  112737  000001  034420          MOVB    #1,ACTDRV               ;SET ''ACTIVE DRIVER'' FLAG
035360  104412                          SAVREG                          ;SAVE R0 - R5
035362  011002                          MOV     (R0),R2                 ;PICKUP THE DRIVE PARAMETER BLOCK POINTER
035364  005062  000016                  CLR     16(R2)                  ;CLEAR THE STATUS/ERROR INDICATOR
035370  111201                          MOVB    (R2),R1                 ;PICKUP THE DRIVE NUMBER
035372  013704  034502                  MOV     RPADR,R4                ;UNIBUS ADDRESS OF RPCS1
035376  105761  034354                  TSTB    DRVSTA(R1)              ;CHECK DRIVES STATUS
035402  003014                          BGT     1$                      ;BRANCH IF ONLINE
035404  105761  034422                  TSTB    ULDFLG(R1)              ;UNLOAD COMMAND IN QUEUE?
035410  001036                          BNE     3$                      ;BRANCH IF YES
035412  105761  034374                  TSTB    DPINT(R1)               ;TRYING TO INIT THE DRIVE
035416  001042                          BNE     5$                      ;BR IF YES
035420  004037  034732                  JSR     R0,DRVINT               ;GO INIT. THE DRIVE
035424  000434                          BR      4$                      ;ERROR RETURN
035426  105761  034354                  TSTB    DRVSTA(R1)              ;IS DRIVE STATUS ONLINE?
035432  003445                          BLE     6$                      ;BR IF NOT
035434  105761  034404          1$:     TSTB    DPRQS(R1)               ;OUTSTANDING PORT REQUEST FOR THE DRIVE ?
035440  001031                          BNE     5$                      ;BR IF YES
035442  010164  000010                  MOV     R1,RPCS2(R4)            ;SELECT THE DRIVE
035446  004037  042750                  JSR     R0,DRVQUE               ;PUT THIS REQUEST IN QUEUE
035452  000460                          BR      9$                      ;QUEUE IS FULL
035454  122762  000103  000002          CMPB    #103,2(R2)              ;IS THIS REQ. FOR AN UNLOAD?
035462  001003                          BNE     2$                      ;BR IF NO
035464  112761  177777  034422          MOVB    #-1,ULDFLG(R1)          ;SET THE ''UNLOAD IN QUEUE'' FLAG
035472  105761  034344          2$:     TSTB    DRVACT(R1)              ;IS THIS DRIVE ACTIVE?
035476  001043                          BNE     8$                      ;BR IF YES
035500  004737  035632                  JSR     PC,OPT                  ;CALL THE OPTIMIZER
035504  000440                          BR      8$
035506  012762  120000  000016  3$:     MOV     #BIT15!BIT13,16(R2) ;SET THE ''UNLOAD IN QUEUE'' ERROR FLAG
035514  000434                          BR      8$                      ;EXIT
035516  004737  036742          4$:     JSR     PC,CI7                  ;GO HANDLE THE PARITY ERROR
035522  000431                          BR      8$
035524  004037  042750          5$:     JSR     R0,DRVQUE               ;PUT REQUEST IN QUEUE
035530  000431                          BR      9$                      ;QUEUE IS FULL
035532  032714  000100                  BIT     #BIT06,(R4)             ;IS 'IE' SET ALREADY ?
035536  001023                          BNE     8$                      ;BR IF IT IS
035540  004737  042306                  JSR     PC,SET.IE               ;SET INTERRUPT
035544  000420                          BR      8$                      ;RETURN, REQUEST IN QUEUE
035546  105761  034354          6$:     TSTB    DRVSTA(R1)              ;SEE IF DRIVE OFFLINE OR UNSAFE
035552  002412                          BLT     7$                      ;BR IF UNSAFE
035554  012762  140000  000016          MOV     #BIT15!BIT14,16(R2)     ;SET OFFLINE ERROR INDICATOR
035562  105761  034364                  TSTB    DRVTYP(R1)              ;SEE IF OFFLINE OR NONEXISTENT
035566  001007                          BNE     8$                      ;BR IF OFFLINE
035570  012762  100002  000016          MOV     #BIT15!BIT01,16(R2)     ;REPORT DRIVE NONEXISTENT
035576  000403                          BR      8$                      ;GO TO EXIT
035600  012762  110000  000016  7$:     MOV     #BIT15!BIT12,16(R2)     ;DRIVE IS UNSAFE
035606  104413                  8$:     RESREG                          ;RESTORE R0 - R5
035610  005720                          TST     (R0)+                   ;SETUP FOR NORMAL RETURN
035612  000401                          BR      10$                     ;FINISH UP, THEN EXIT
035614  104413                  9$:     RESREG                          ;RESTORE R0 - R5
035616  005720                  10$:    TST     (R0)+                   ;CORRECT THE RETURN ADDRESS
```

```
035620  105037  034420                  CLRB    ACTDRV            ;CLEAR ''ACTIVE DRIVER'' FLAG
035624  012637  177776                  MOV     (SP)+,@#PS        ;RETURN ''PS'' TO USER LEVEL
035630  000200                          RTS     R0                ;RETURN TO CALLER

                                ;OPTIMIZER-CALLED FOR A PARTICULAR DRIVE
                                ;
                                ;CALL
                                ;       MOV     #DRVNUM,R1        ;DRIVE NUMBER TO R1
                                ;       JSR     PC,OPT            ;SETUP A COMMAND
                                ;
035632  104412                  OPT:    SAVREG                    ;SAVE R0 - R5
035634  013746  177776                  MOV     @#PS,-(SP)        ;SAVE PROC. STATUS
035640  146137  034470  034416          BICB    ATABIT(R1),SRCHWT         ;CLEAR ''SEARCH WAIT'' KEY
035646  004737  043024                  JSR     PC,GETREQ         ;GET ''DPB'' POINTER OF REQUEST
035652  005702                          TST     R2                ;IS THERE A REQUEST IN QUEUE?
035654  001505                          BEQ     7$                ;NO--BRANCH TO EXIT
035656  032764  004000  000000          BIT     #BIT11,RPCS1(R4)          ;IS DVA SET?
035664  001407                          BEQ     10$               ;BRANCH IF NOT
035666  032764  000100  000012          BIT     #BIT6,RPDS1(R4)   ;IS VV SET ?
035674  001003                          BNE     10$               ;BR IF IT IS
035676  004037  034732          9$:     JSR     R0,DRVINT         ;SEE IF DRIVE STILL ONLINE ?
035702  000470                          BR      6$                ;PARITY OR 'DVA' NOT SET
035704  105761  034354          10$:    TSTB    DRVSTA(R1)        ;IS DRIVE ONLINE?
035710  003014                          BGT     1$                ;YES--BRANCH
035712  004737  043046                  JSR     PC,POPQUE         ;NO--REMOVE REQUEST FROM QUEUE
035716  012762  140000  000016          MOV     #BIT15!BIT14,16(R2)   ;SET OFFLINE STATUS/ERROR INDICATOR
035724  105761  034354                  TSTB    DRVSTA(R1)        ;IS DRIVE UNSAFE ?
035730  100064                          BPL     8$                ;BR TO EXIT IF NOT
035732  012762  110000  000016          MOV     #BIT15!BIT12,16(R2)   ;SET UNSAFE STATUS/ERROR INDICATOR
035740  000460                          BR      8$                ;BRANCH TO EXIT
035742  012746  000111          1$:     MOV     #111,-(SP)        ;LOAD COMMAND ONTO THE STACK
035746  004037  042002                  JSR     R0,WRT.RP         ;LOAD THE REGISTER
035752  000000                          RPCS1                     ;REGISTER INCREMENT
035754  036064                          6$                        ;ERROR RETURN ADDRESS
035756  032714  004000                  BIT     #BIT11,(R4)       ;DRIVE AVAILABLE ?
035762  001427                          BEQ     5$                ;BR IF NOT
035764  122762  000150  000002          CMPB    #150,2(R2)        ;IS THE REQUEST FOR I/O?
035772  002403                          BLT     2$                ;YES--BRANCH
035774  004737  036326                  JSR     PC,CI4            ;CALL THE COMMAND INITIATOR
036000  000440                          BR      8$                ;BRANCH TO EXIT
036002  005737  034466          2$:     TST     DTUW              ;DATA TRANSFER UNDERWAY?
036006  002012                          BGE     4$                ;YES--GO START A SEARCH
036010  005737  034444                  TST     SEEKFG            ;DO IMPLIED SEEKS?
036014  100404                          BMI     3$                ;YES---BRANCH
036016  004037  037276                  JSR     R0,LA             ;NO--DO LOOK AHEAD
036022  000427                          BR      8$                ;RETURN HERE ON A PARITY ERROR
036024  000403                          BR      4$                ;GO START A SEARCH
036026  004737  036112          3$:     JSR     PC,CI1            ;START A DATA TRANSFER
036032  000423                          BR      8$
036034  004737  036220          4$:     JSR     PC,CI3            ;START A SEARCH
036040  000420                          BR      8$                ;GO TO THE EXIT
036042  112761  177777  034404  5$:     MOVB    #-1,DPRQS(R1)     ;SET PORT REQUEST INDICATOR
036050  010103                          MOV     R1,R3             ;SET UP TO ADDRESS WORDS
036052  006303                          ASL     R3                ;CONVERT TO WORD INDEX
036054  012763  023420  034446          MOV     #10000.,TIMER(R3)   ;START 10 SEC TIMER
036062  000402                          BR      7$                ;EXIT
036064  004737  036742          6$:     JSR     PC,CI7            ;PROCESS THE PARITY ERROR
```

```
036070  032714  000100          7$:     BIT     #BIT06,(R4)     ;SEE IF 'IE' ALREADY SET
036074  001002                          BNE     8$              ;BR IF SET
036076  004737  042306                  JSR     PC,SET.IE       ;SET "IE" WITHOUT A "TRE"
036102  012637  177776          8$:     MOV     (SP)+,@#PS      ;RESTORE PROC. STATUS
036106  104413                          RESREG                  ;RESTORE R0 - R5
036110  000207                          RTS     PC

                                ;COMMAND INITIATOR
                                ;
                                ;CALL
                                ;       MOV     #DRVNUM,R1      ;DRIVE NUMBER
                                ;       MOV     #DPB,R2         ;ADDRESS OF DPB
                                ;       JSR     PC,CI?          ;CI?= CI1,CI3, OR CI4
                                ;                               ;WHERE:
                                ;                               ;CI1=DATA TRANSFER
                                ;                               ;CI2=SEARCH REQUESTED BY DATA XFER
                                ;                               ;CI4=NOT DATA TRANSFER

036112  004737  043046          CI1:    JSR     PC,POPQUE       ;REMOVE REQUEST FROM ''DRIVES WAIT'' QUEUE
036116  010237  034414                  MOV     R2,TRNSWT       ;PUT REQ. IN TRANSFER WAIT QUEUE
036122  010203                          MOV     R2,R3           ;DPB ADDRESS TO R3
036124  013704  034502                  MOV     RPADR,R4        ;RPCS1 ADDRESS
036130  010164  000010                  MOV     R1,RPCS2(R4)    ;SELECT DRIVE
036134  062703  000004                  ADD     #4,R3           ;DESIRED WORD COUNT
036140  062704  000002                  ADD     #2,R4           ;RPWC ADDRESS
036144  012324                          MOV     (R3)+,(R4)+     ;LOAD WORD COUNT
036146  012324                          MOV     (R3)+,(R4)+     ;LOAD BUFFER ADDRESS
036150  012346                          MOV     (R3)+,-(SP)     ;LOAD SECTOR AND TRACK
036152  004037  042002                  JSR     R0,WRT.RP       ;CALL THE LOAD(WRITE) ROUTINE
036156  000006                          RPDA                    ;INDEX OF REGISTER TO LOAD
036160  036742                          CI7                     ;ERROR RETURN ADDRESS
036162  012346                          MOV     (R3)+,-(SP)     ;LOAD CYLINDER ADDRESS
036164  004037  042002                  JSR     R0,WRT.RP
036170  000034                          RPCA
036172  036742                          CI7
036174  016246  000002                  MOV     2(R2),-(SP)     ;LOAD ''COMMAND+GO'', ''A17&A16'', AND ''PSEL''
036200  004037  042002                  JSR     R0,WRT.RP
036204  000000                          RPCS1
036206  036742                          CI7
036210  010137  034466                  MOV     R1,DTUW         ;SET ''DATA TRANSFER UNDERWAY''
036214  000137  036704                  JMP     CI5
036220  013704  034502          CI3:    MOV     RPADR,R4        ;RPCS1 ADDRESS
036224  010164  000010                  MOV     R1,RPCS2(R4)    ;SELECT DRIVE
036230  016246  000012                  MOV     12(R2),-(SP)    ;DESIRED CYLINDER ADDRESS
036234  004037  042002                  JSR     R0,WRT.RP
036240  000034                          RPCA
036242  036742                          CI7
036244  116203  000010                  MOVB    10(R2),R3       ;PICKUP SECTOR ADDRESS
036250  163703  034516                  SUB     MXWNDW,R3       ;BACKUP BY MAX. SEARCH FOR I/O WINDOW
036254  002002                          BGE     1$
036256  062703  000026                  ADD     #22.,R3
036262  010346                  1$:     MOV     R3,-(SP)        ;COMBINE THE ADJUSTED SECTOR WITH
036264  116266  000011  000001          MOVB    11(R2),1(SP)    ;THE DESIRED TRACK
036272  004037  042002                  JSR     R0,WRT.RP       ;LOAD DESIRED TRACK & SECTOR
036276  000006                          RPDA
036300  036742                          CI7
036302  012746  000131                  MOV     #131,-(SP)      ;START A SEARCH
```

```
036306  004037  042002                  JSR     R0,WRT.RP
036312  000000                          RPCS1
036314  036742                          CI7
036316  156137  034470  034416          BISB    ATABIT(R1),SRCHWT       ;SET ''SEARCH WAIT'' KEY
036324  000567                          BR      CI5
036326  013704  034502          CI4:    MOV     RPADR,R4        ;RPCS1 ADDRESS
036332  010164  000010                  MOV     R1,RPCS2(R4)    ;SELECT DRIVE
036336  116203  000002                  MOVB    2(R2),R3        ;PICKUP THE REQUESTED COMMAND
036342  122703  000131                  CMPB    #131,R3         ;IS IT A SEARCH COMMAND?
036346  001007                          BNE     1$              ;BRANCH IF NO
036350  016246  000010                  MOV     10(R2),-(SP)    ;LOAD DESIRED TRACK & SECTOR
036354  004037  042002                  JSR     R0,WRT.RP
036360  000006                          RPDA
036362  036742                          CI7
036364  000403                          BR      2$              ;GO LOAD CYLINDER
036366  122703  000105          1$:     CMPB    #105,R3         ;IS IT A SEEK COMMAND
036372  001007                          BNE     3$              ;BRANCH IF NO
036374  016246  000012          2$:     MOV     12(R2),-(SP)    ;LOAD DESIRED CYLINDER
036400  004037  042002                  JSR     R0,WRT.RP
036404  000034                          RPCA
036406  036742                          CI7
036410  000546                          BR      CI6
036412  122703  000115          3$:     CMPB    #115,R3         ;IS IT AN ''OFFSET'' COMMAND?
036416  001013                          BNE     4$              ;BR IF NO
036420  004037  041626                  JSR     R0,RD.RP        ;MERGE THE OFFSET VALUE INTO RPOF
036424  000032                          RPOF                    ;BUT DON'T CHANGE THE UPPER
036426  036742                          CI7
036430  116216  000001                  MOVB    1(R2),(SP)      ;BYTE WHEN LOADING THE
036434  004037  042002                  JSR     R0,WRT.RP       ;REGISTER (RPOF)
036440  000032                          RPOF
036442  036742                          CI7
036444  000530                          BR      CI6             ;GO START THE COMMAND
036446  122703  000107          4$:     CMPB    #107,R3         ;IS IT A ''RECALIBRATE'' COMMAND?
036452  001525                          BEQ     CI6             ;BRANCH IF YES
036454  122703  000117                  CMPB    #117,R3         ;IS IT A RETURN TO CENTER?
036460  001522                          BEQ     CI6             ;BRANCH IF YES
036462  122703  000103                  CMPB    #103,R3         ;IS IT AN ''UNLOAD'' COMMAND?
036466  001016                          BNE     5$              ;BRANCH IF NO
036470  112761  000001  034344          MOVB    #1,DRVACT(R1)   ;SET THE DRIVE ACTIVE INDICATOR
036476  105061  034354                  CLRB    DRVSTA(R1)      ;PUT DRIVE STATUS TO OFFLINE
036502  112761  000001  034422          MOVB    #1,ULDFLG(R1)   ;SET ''UNLOAD IN PROGRESS'' FLAG
036510  010346                          MOV     R3,-(SP)        ;START THE ''UNLOAD'' COMMAND
036512  004037  042002                  JSR     R0,WRT.RP
036516  000000                          RPCS1
036520  036742                          CI7
036522  000207                          RTS     PC              ;RETURN TO USER
036524  122703  000143          5$:     CMPB    #143,R3         ;IS IT A ''SET FORMAT'' COMMAND?
036530  001014                          BNE     6$              ;BRANCH IF NO
036532  004037  041626                  JSR     R0,RD.RP        ;READ THE OFFSET REGISTER
036536  000032                          RPOF
036540  036742                          CI7
036542  116266  000001  000001          MOVB    1(R2),1(SP)     ;COMBINE ''FMT22'',''ECI'', AND ''HCI''
036550  004037  042002                  JSR     R0,WRT.RP       ;LOAD ''FMT22'', ''ECI'', AND/OR ''HCI''.
036554  000032                          RPOF
036556  036742                          CI7
036560  000436                          BR      12$
036562  122703  000141          6$:     CMPB    #141,R3         ;IS IT A ''GET REGISTER'' COMMAND?
```

```
036566  001023                           BNE     10$                 ;BRANCH IF NO
036570  016203  000006           7$:     MOV     6(R2),R3            ;POINTS TO 1ST ADDRESS OF WHERE
                                                                     ;TO PUT THE REGISTER(S)
036574  116237  000010  036612           MOVB    10(R2),9$           ;INIT. THE INDEX FOR THE FIRST REG.
036602  116205  000011                   MOVB    11(R2),R5           ;INDEX OF LAST REG. TO MOVE
036606  004037  041626           8$:     JSR     R0,RD.RP            ;READ RP04/5/6 REGISTER
036612  000000                   9$:     RPCS1                       ;INDEX OF REG. TO READ
036614  036742                           CI7
036616  012623                           MOV     (SP)+,(R3)+         ;GET THE CONTENTS OF RH11/RP04/5/6 REG.
036620  023705  036612                   CMP     9$,R5               ;LAST REG. BEEN READ?
036624  001414                           BEQ     12$                 ;GET OUT IF YES
036626  062737  000002  036612           ADD     #2,9$               ;INCREASE THE INDEX BY 2
036634  000764                           BR      8$                  ;LOOP--MORE TO READ
036636  122703  000145           10$:    CMPB    #145,R3             ;IS IT A "SELECT DRIVE" COMMAND?
036642  001405                           BEQ     12$                 ;BRANCH IF YES
036644  010346                   11$:    MOV     R3,-(SP)            ;LOAD THE COMMAND
036646  004037  042002                   JSR     R0,WRT.RP
036652  000000                           RPCS1
036654  036742                           CI7
036656  004737  043046           12$:    JSR     PC,POPQUE           ;REMOVE REQ. FROM QUEUE
036662  052762  000200  000016           BIS     #BIT07,16(R2)       ;SET THE "DONE" BIT
036670  005737  034442                   TST     SAVEFG              ;SAVE THE RH11/RP04/5/6 REGISTERS?
036674  100002                           BPL     13$                 ;BRANCH IF NO
036676  004737  042170                   JSR     PC,SVRH11           ;YES--GO SAVE THE REGISTERS
036702  000207                   13$:    RTS     PC                  ;RETURN TO USER
036704  006301                   CI5:    ASL     R1
036706  012761  001750  034446           MOV     #1000.,TIMER(R1)            ;SET A ONE SECOND TIMER
036714  006201                           ASR     R1
036716  112761  000001  034344           MOVB    #1,DRVACT(R1)       ;SET THE DRIVE ACTIVE
036724  000207                           RTS     PC                  ;RETURN TO THE USER
036726  010346                   CI6:    MOV     R3,-(SP)            ;LOAD THE COMMAND
036730  004037  042002                   JSR     R0,WRT.RP
036734  000000                           RPCS1
036736  036742                           CI7
036740  000761                           BR      CI5
036742  032764  010000  000010   CI7:    BIT     #BIT12,RPCS2(R4)    ;DRIVE NON-EXISTENT ?
036750  001034                           BNE     CI8                 ;BR IF YES
036752  005702                   1$:     TST     R2                  ;ANYTHING IN QUEUE ?
036754  001405                           BEQ     CI7B                ;BR IF NOT
036756  012762  104000  000016           MOV     #BIT15!BIT11,16(R2)     ;SET "PARITY" ERROR INDICATOR
036764  004737  042170                   JSR     PC,SVRH11           ;GO SAVE THE RH11/RP04/5/6 REGISTERS
036770  012746  000111           CI7B:   MOV     #111,-(SP)          ;DO A "DRIVE CLEAR"
036774  004037  042002                   JSR     R0,WRT.RP
037000  000000                           RPCS1
037002  037042                           CI8
037004  004737  042730                   JSR     PC,EMPTYQ           ;EMPTY THE QUEUE
037010  105061  034422                   CLRB    ULDFLG(R1)          ;CLEAR THE UNLOAD IN QUEUE FLAG
037014  105061  034344                   CLRB    DRVACT(R1)          ;DRIVE IS IDLE
037020  020137  034466                   CMP     R1,DTUW             ;IF THIS DRIVE HAD AN I/O REQUEST
037024  001005                           BNE     1$                  ;IN PROGRESS CLEAR ALL OF THE FLAGS
037026  005037  034414                   CLR     TRNSWT
037032  012737  177777  034466           MOV     #-1,DTUW
037040  000207                   1$:     RTS     PC
037042  104412                   CI8:    SAVREG                      ;SAVE R0 - R5
037044  032764  010000  000010           BIT     #BIT12,RPCS2(R4)    ;IS 'NED' SET ?
037052  001002                           BNE     1$                  ;BR IF YES
037054  005001                           CLR     R1
```

```
037056  005003                            CLR     R3
037060  105761  034344            1$:     TSTB    DRVACT(R1)          ;DRIVE ACTIVE?
037064  001443                            BEQ     5$                  ;BRANCH IF NO
037066  013702  034414                    MOV     TRNSWT,R2           ;GET THE "TRANSFER WAIT" QUEUE
037072  020137  034466                    CMP     R1,DTUW             ;DID THIS DRIVE HAVE AN I/O IN PROGRESS?
037076  001402                            BEQ     2$                  ;BRANCH IF YES
037100  004737  043024                    JSR     PC,GETREQ           ;GET THE DPB POINTER
037104  005702                    2$:     TST     R2                  ;QUEUE ENTRY FOR DRIVE ?
037106  001415                            BEQ     4$                  ;BR IF NOT
037110  032764  010000  000010            BIT     #BIT12,RPCS2(R4)    ;'NED' SET ?
037116  001404                            BEQ     3$                  ;BR IF NOT
037120  012762  100002  000016            MOV     #BIT15!BIT01,16(R2)  ;SET 'DRIVE NON-EXISTENT' INDICATOR
037126  000405                            BR      4$                  ;CONTINUE
037130  012762  102000  000016    3$:     MOV     #BIT15!BIT10,16(R2) ;SET "NON-CLEARABLE PARITY" ERROR INDICATOR
037136  004737  042170                    JSR     PC,SVRH11           ;SAVE RH11/RP04/5/6 REGISTERS
037142  012763  177777  034446    4$:     MOV     #-1,TIMER(R3)       ;STOP THE TIMER
037150  105061  034344                    CLRB    DRVACT(R1)          ;SET "DRIVE ACTIVE" TO IDLE
037154  020137  034466                    CMP     R1,DTUW             ;IS THIS DRIVE SETUP FOR A TRANSFER
037160  001005                            BNE     5$                  ;BR IF NOT
037162  012737  177777  034466            MOV     #-1,DTUW            ;RESET THE INDICATOR
037170  005037  034414                    CLR     TRNSWT              ;CLEAR THE TRANSFER QUEUE
037174  105061  034422            5$:     CLRB    ULDFLG(R1)          ;CLEAR UNLOAD FLAG
037200  032764  010000  000010            BIT     #BIT12,RPCS2(R4)    ;'NED' SET ?
037206  001021                            BNE     6$                  ;BR IF YES
037210  005201                            INC     R1                  ;MOVE TO THE NEXT DRIVE
037212  062703  000002                    ADD     #2,R3
037216  042701  177770                    BIC     #^C7,R1
037222  001316                            BNE     1$                  ;BRANCH IF MORE DRIVES
037224  012737  177777  034466            MOV     #-1,DTUW            ;NO DATA TRANSFERS UNDERWAY
037232  005037  034414                    CLR     TRNSWT              ;CLEAR THE 'TRANSFER WAIT' QUEUE
037236  004737  042652                    JSR     PC,CLRQUE           ;CLEAR ALL OF THE REQUEST QUEUES
037242  012764  000040  000010            MOV     #BIT05,RPCS2(R4)    ;DO A MASSBUS INIT.
037250  000406                            BR      7$                  ;CONTINUE
037252  004737  042730            6$:     JSR     PC,EMPTYQ           ;CLEAR THE DRIVE'S QUEUE
037256  105061  034354                    CLRB    DRVSTA(R1)          ;SET DRIVE TO OFFLINE
037262  105061  034364                    CLRB    DRVTYP(R1)          ;CLEAR THE DRIVE TYPE INDICATOR
037266  004737  042306            7$:     JSR     PC,SET.IE           ;SET "IE" WITHOUT "TRE"
037272  104413                            RESREG                      ;RESTORE R0 - R5
037274  000207                            RTS     PC                  ;RETURN

                                  ;LOOK AHEAD ROUTINE
                                  ;
                                  ;CALL
                                  ;       MOV     #DRVNUM,R1          ;DRIVE NUMBER
                                  ;       MOV     #DPB,R2             ;POINT TO DPB
                                  ;       JSR     R0,LA               ;GO CHECK THE WINDOW
                                  ;       RETURN1                     ;ERROR RETURN
                                  ;       RETURN2                     ;START A SEARCH
                                  ;       RETURN3                     ;START A DATA TRANSFER

037276  013704  034502            LA:     MOV     RPADR,R4            ;GET RPCS1'S ADDRESS
037302  010164  000010                    MOV     R1,RPCS2(R4)        ;SELECT DRIVE
037306  004037  041626                    JSR     R0,RD.RP            ;READ CURRENT CYLINDER
037312  000036                            RPCC
037314  037426                            4$                          ;ERROR RETURN ADDRESS
037316  022662  000012                    CMP     (SP)+,12(R2)        ;IS CURRENT CYLINDER=DESIRED
                                                                      ;CYLINDER?
```

```
 748 037322  001037                          BNE     3$               ;EXIT IF NO
 749 037324  105261  034432                  INCB    LACNT(R1)        ;INCREMENT THE LOOK AHEAD COUNT
 750 037330  126137  034432  034510          CMPB    LACNT(R1),MXLACT ;EXCEED MAX?
 751 037336  003026                          BGT     2$               ;BRANCH IF YES
 752 037340  116203  000010                  MOVB    10(R2),R3        ;GET DESIRED SECTOR ADDRESS AND
 753 037344  000303                          SWAB    R3               ;MULT. BY 64--ALIGN WITH
 754 037346  006203                          ASR     R3               ;LOOK AHEAD REGISTER
 755 037350  006203                          ASR     R3
 756 037352  012737  000340  177776          MOV     #340,@#PS        ;PRIORITY LEVEL '7''
 757 037360  004037  041626                  JSR     R0,RD.RP         ;READ LOOK AHEAD REGISTER
 758 037364  000020                          RPLA
 759 037366  037426                          4$
 760 037370  162603                          SUB     (SP)+,R3         ;CALCULATE THE DELTA
 761 037372  002002                          BGE     1$
 762 037374  062703  002600                  ADD     #<22.*64.>,R3    ;MAKE THE DELTA POSITIVE
 763 037400  023703  034512          1$:     CMP     MXDLTA,R3        ;CHECK THE DELTA TO SEE
 764 037404  002406                          BLT     3$               ;IF IT IS WITHIN THE
 765 037406  023703  034514                  CMP     MNDLTA,R3        ;WINDOW---IF YES, ZERO
 766 037412  002003                          BGE     3$               ;THE LOOK AHEAD COUNT
 767 037414  105061  034432          2$:     CLRB    LACNT(R1)        ;AND TAKE THE I/O EXIT
 768 037420  005720                          TST     (R0)+
 769 037422  005720                  3$:     TST     (R0)+            ;ADJUST THE RETURN ADDRESS
 770 037424  000402                          BR      5$               ;EXIT
 771 037426  004737  036742          4$:     JSR     PC,CI7           ;PROCESS THE ERROR
 772 037432  000200                  5$:     RTS     R0               ;RETURN
 773
 774                                 ;INTERRUPT SERVICE ROUTINE
 775
 776 037434  112737  000001  034420  ISR:    MOVB    #1,ACTDRV        ;SET ''ACTIVE DRIVER'' FLAG
 777 037442  104412                          SAVREG                   ;SAVE R0 - R5
 778 037444  013704  034502                  MOV     RPADR,R4         ;ADDRESS OF RHSCS1
 779 037450  013701  034466                  MOV     DTUW,R1          ;GET ''DATA TRANSFER UNDERWAY'' INDICATOR
 780 037454  002403                          BLT     1$               ;BRANCH IF NO DATA TRANSFER UNDERWAY
 781 037456  004737  037500                  JSR     PC,TD            ;CALL TRANSFER DONE
 782 037462  000402                          BR      2$               ;EXIT
 783 037464  004737  037762          1$:     JSR     PC,SC            ;CALL SPECIAL CONDITIONS
 784 037470  104413                  2$:     RESREG                   ;RESTORE R0 - R5
 785 037472  105037  034420                  CLRB    ACTDRV           ;CLEAR ''ACTIVE DRIVER'' FLAG
 786 037476  000002                          RTI                      ;RETURN
 787
 788                                 ;TRANSFER DONE ROUTINE
 789
 790 037500  105061  034344          TD:     CLRB    DRVACT(R1)       ;SET DRIVE ACTIVE INDICATOR TO IDLE
 791 037504  012737  177777  034466          MOV     #-1,DTUW         ;NO DATA TRANSFERS UNDERWAY
 792 037512  006301                          ASL     R1
 793 037514  012761  177777  034446          MOV     #-1,TIMER(R1)    ;CANCEL TIMEOUT
 794 037522  006201                          ASR     R1
 795 037524  013702  034414                  MOV     TRNSWT,R2        ;GET ''DPB'' ADDRESS FROM THE
 796 037530  005037  034414                  CLR     TRNSWT           ;TRANSFER WAIT QUEUE--CLEAR QUEUE
 797 037534  052762  000200  000016          BIS     #BIT07,16(R2)    ;SET DONE
 798 037542  010164  000010                  MOV     R1,RPCS2(R4)     ;SELECT THE DRIVE
 799 037546  004037  041626                  JSR     R0,RD.RP         ;TRANSFER ERROR(TRE=1)?
 800 037552  000000                          RPCS1
 801 037554  036742                          CI7
 802 037556  006126                          ROL     (SP)+
 803 037560  100421                          BMI     3$               ;BR IF YES
 804 037562  005737  034442                  TST     SAVEFG           ;SAVE THE RH11/RP04/5/6 REGISTERS?
```

```
805 037566  100002                          BPL     1$              ;BRANCH IF NO
806 037570  004737  042170                  JSR     PC,SVRH11       ;YES--SAVE THE REGISTERS
808 037574  004737  037654          1$:     JSR     PC,WC           ;SEE IF WRITE CHECK TO BE PUT IN QUEUE
809 037600  004737  043024                  JSR     PC,GETREQ       ;GET DPB POINTER
810 037604  005702                          TST     R2              ;ENTRY FOR DRIVE ?
811 037606  001403                          BEQ     2$              ;BR IF NOT
812 037610  004737  035632                  JSR     PC,OPT          ;CALL OPTIMIZER
816 037614  000462                          BR      SC              ;CHECK OTHER DRIVES
817 037616  012714  000113          2$:     MOV     #113,(R4)       ;RELEASE THE DRIVE
818 037622  000457                          BR      SC              ;CHECK FOR OTHER DRIVES
819 037624  052762  100100  000016  3$:     BIS     #BIT15!BIT06,16(R2) ;SET DATA ERROR FLAG
820 037632  004737  042730                  JSR     PC,EMPTYQ       ;EMPTY THE 'DRIVE'S WAIT'' QUEUE
821 037636  004737  042170                  JSR     PC,SVRH11       ;SAVE THE RH11/RP04/5/6 REGISTERS
822 037642  012714  040111                  MOV     #40111,(R4)     ;ISSUE A 'DRIVE CLEAR''
823 037646  012714  000113                  MOV     #113,(R4)       ;ISSUE A RELEASE TO THE DRIVE
824 037652  000443                          BR      SC              ;CHECK FOR OTHER DRIVES
825
827                                 ;FORCED WRITE CHECK ROUTINE
828
829 037654  005737  001424          WC:     TST     AUTOCK          ;AUTOMATIC WRITE CHECKS ?
830 037660  001437                          BEQ     2$              ;BR IF NOT
831 037662  122762  000002  000024          CMPB    #2,$CODE(R2)    ;LAST OPERATION WRITE DATA ?
832 037670  001404                          BEQ     1$              ;BR IF IT WAS
833 037672  122762  000003  000024          CMPB    #3,$CODE(R2)    ;LAST OPERATION WRITE HEADER & DATA ?
834 037700  001027                          BNE     2$              ;BR IF NOT
835 037702  004037  042750          1$:     JSR     R0,DRVQUE       ;PUT THE OPERATION IN THE QUEUE
836 037706  000424                          BR      2$              ;QUEUE IS FULL
837 037710  005062  000016                  CLR     16(R2)          ;CLEAR 'DONE' BIT IN DPB
838 037714  116262  000234  000027          MOVB    $RPCS1(R2),$PREVO(R2)   ;SAVE WRITE OPERATION CODE
839 037722  016262  000012  000034          MOV     $CYL(R2),$PREVA+2(R2)   ;SAVE CYLINDER
840 037730  016262  000010  000032          MOV     $SEC(R2),$PREVA(R2)   ;SAVE SECTOR AND TRACK ADDRESSES
841 037736  142762  000002  000024          BICB    #2,$CODE(R2)    ;CHANGE WRITE TO CHECK
842 037744  142762  000020  000002          BICB    #20,$COMND(R2)  ;CHANGE DRIVER CODE TO WRITE CHECK
843 037752  152762  000010  000002          BISB    #10,$COMND(R2)  ;FINISH CHANGING CODE TO WRITE CHECK
844 037760  000207                  2$:     RTS     PC              ;EXIT
845
847                                 ;SPECIAL CONDITION ROUTINE
848
849 037762  116403  000016          SC:     MOVB    RPAS(R4),R3     ;READ ''RPAS''
850 037766  001012                          BNE     2$              ;BRANCH IF ANY 'ATA' BITS SET
851 037770  004037  041626                  JSR     R0,RD.RP        ;READ CONTROL AND STATUS REGISTER
852 037774  000000                          RPCS1
853 037776  037042                          CI8
854 040000  106126                          ROLB    (SP)+           ;IS ''IE''=1?
855 040002  100403                          BMI     1$              ;YES, NO DRIVES TO CHECK
856 040004  104001                          EMT     1
857 040006  004737  042306                  JSR     PC,SET.IE       ;SET INTERRUPT ENABLE
858 040012  000207                  1$:     RTS     PC              ;RETURN
859 040014  005046                  2$:     CLR     -(SP)           ;PROCESS ALL DRIVES THAT HAVE
860 040016  110316                          MOVB    R3,(SP)         ;AN ''ATA''=1
861 040020  012703  000001                  MOV     #1,R3
862 040024  005001                          CLR     R1
863 040026  030316                  SC3:    BIT     R3,(SP)         ;ATA=1?
864 040030  001005                          BNE     SC5             ;YES--BRANCH
865 040032  005201                  SC4:    INC     R1              ;MOVE TO THE NEXT DRIVE
866 040034  106303                          ASLB    R3
867 040036  001373                          BNE     SC3             ;BRANCH IF MORE TO CHECK?
```

```
    868 040040  005726                          TST     (SP)+               ;CLEAN OFF THE STACK
    869 040042  000207                          RTS     PC                  ;RETURN TO USER
    870 040044  105761  034374          SC5:    TSTB    DPINT(R1)           ;INITIALIZING THE DRIVE ?
    871 040050  001402                          BEQ     1$                  ;BR IF NOT
    872 040052  000137  040740                  JMP     SC13                ;PROCESS THE DRIVE
    873 040056  105761  034404          1$:     TSTB    DPRQS(R1)           ;PORT REQUEST OUTSTANDING ?
    874 040062  001402                          BEQ     2$                  ;BR IF NOT
    875 040064  000137  040740                  JMP     SC13                ;START THE OUTSTANDING COMMAND
    876 040070  105761  034354          2$:     TSTB    DRVSTA(R1)          ;CHECK THE DRIVE STATUS
    877 040074  003025                          BGT     5$                  ;BRANCH IF ONLINE
    878 040076  105761  034422                  TSTB    ULDFLG(R1)          ;UNLOAD IN PROGRESS?
    879 040102  003422                          BLE     5$                  ;BRANCH IF NOT
    880 040104  004737  043024                  JSR     PC,GETREQ           ;GET DPB POINTER
    881 040110  004737  042170                  JSR     PC,SVRH11           ;SAVE THE RH11/RP04/5/6 REGISTERS
    882 040114  004737  040670                  JSR     PC,SC12             ;SAVE RPDS1, RPER1, RPER2, AND RPER3
    883                                                                     ;ALSO DO A DRIVE INIT (DRVINT)
    884 040120  105761  034354                  TSTB    DRVSTA(R1)          ;DID DRIVE COME ONLINE?
    885 040124  003416                          BLE     6$                  ;NO---BRANCH
    886 040126  032737  040000  034334          BIT     #BIT14,RPERRS       ;WAS THERE AN ERROR?
    887 040134  001002                          BNE     3$                  ;BR IF ERROR
    888 040136  000137  040600                  JMP     SC11                ;NO ERROR
    889 040142  013705  034336          3$:     MOV     RPERRS+2,R5         ;YES -- PICKUP RPER1 AND
    890 040146  000476                          BR      SC6A                ;GO PROCESS THE ERROR
    891 040150  105761  034344          5$:     TSTB    DRVACT(R1)          ;DRIVE ACTIVE WITH COMMAND OR ERROR RECOVERY ?
    892 040154  001027                          BNE     SC6                 ;BR IF EITHER
    893 040156  004737  040670                  JSR     PC,SC12             ;SAVE RPDS1, RPER1, RPER2, AND RPER3
    894                                                                     ;ALSO DO A DRVINT
    895 040162  105761  034374          6$:     TSTB    DPINT(R1)           ;TRYING TO INIT THE DRIVE ?
    896 040166  001321                          BNE     SC4                 ;BR IF YES, CHECK ON MORE DRIVES
    897 040170  105761  034354                  TSTB    DRVSTA(R1)          ;CHECK ON DRIVE'S STATUS
    898 040174  100412                          BMI     7$                  ;BR IF UNSAFE
    899 040176  032737  020000  034342          BIT     #BIT13,RPERRS+6     ;ADDRESS PLUG CHANGED ?
    900 040204  001011                          BNE     8$                  ;BR IF YES
    901 040206  012746  000113                  MOV     #113,-(SP)          ;RELEASE COMMAND
    902 040212  004037  042002                  JSR     R0,WRT.RP           ;WRITE THE COMMAND INTO RPCS1
    903 040216  000000                          RPCS1                       ;REGISTER INDEX
    904 040220  040550                          SC8                         ;PARITY EXIT ADDRESS
    905 040222  011605                  7$:     MOV     (SP),R5             ;PICKUP (RPAS) BEFORE THE ERROR CALL
    906 040224  104002                          EMT     2
    907 040226  000701                          BR      SC4                 ;GO CHECK FOR MORE ATA'S
    908 040230                          8$:
        040230  104005                          EMT     5
    909 040232  000677                          BR      SC4                 ;CHECK FOR MORE DRIVES
    910 040234  006301                  SC6:    ASL     R1                  ;SETUP TO ADDRESS WORDS
    911 040236  012761  177777  034446          MOV     #-1,TIMER(R1)       ;STOP THE TIMER
    912 040244  006201                          ASR     R1                  ;RESTORE THE DRIVE ADDRESS
    913 040246  004737  043024                  JSR     PC,GETREQ           ;GET THE DPB POINTER FROM THE QUEUE
    914 040252  010164  000010                  MOV     R1,RPCS2(R4)        ;SELECT DRIVE
    915 040256  004037  041626                  JSR     R0,RD.RP            ;READ THE RP04'S STATUS REG.
    916 040262  000012                          RPDS1
    917 040264  040550                          SC8
    918 040266  011605                          MOV     (SP),R5             ;AND PUT IT IN R5
    919 040270  006126                          ROL     (SP)+               ;WAS THERE AN ERROR?
    920 040272  100407                          BMI     1$                  ;BR IF ERROR
    921 040274  105761  034344                  TSTB    DRVACT(R1)          ;CHECK DRIVE'S STATE
    922 040300  003137                          BGT     SC11                ;BR IF DRIVE ACTIVE WITH ORDER
    923 040302  052762  100210  000016          BIS     #BIT15!BIT07!BIT03,16(R2)  ;INFORM USER OF ERROR RECOVER COMPLETION
```

```
040310 000470                         BR      SC7
040312 004037 041626          1$:     JSR     R0,RD.RP            ;READ ERROR REGISTER #1
040316 000014                         RPER1
040320 040550                         SC8
040322 012605                         MOV     (SP)+,R5            ;AND SAVE IT IN R5
040324 004737 042170                  JSR     PC,SVRH11           ;SAVE RH11/RP04/5/6 REGISTERS
040330 012746 000111                  MOV     #111,-(SP)          ;ISSUE A DRIVE CLEAR
040334 004037 042002                  JSR     R0,WRT.RP
040340 000000                         RPCS1
040342 040550                         SC8
040344 006105                 SC6A:   ROL     R5                  ;WAS ''UNSAFE'' CONDITION =1?
040346 100406                         BMI     1$                  ;BRANCH IF YES
040350 005702                         TST     R2                  ;ANYTHING IN QUEUE ?
040352 001447                         BEQ     SC7                 ;BR IF NOT
040354 052762 100240 000016           BIS     #BIT15!BIT07!BIT05,16(R2)         ;INFORM USER OF ERROR
040362 000443                         BR      SC7
040364 004037 041626          1$:     JSR     R0,RD.RP            ;READ DRIVE STATUS REG. #1
040370 000012                         RPDS1
040372 040550                         SC8
040374 011605                         MOV     (SP),R5             ;SAVE RPDS1 IN R5
040376 006126                         ROL     (SP)+               ;''ERR''=1?
040400 100011                         BPL     2$                  ;BR IF NO--UNSAFE CLEARED
040402 112761 177777 034354           MOVB    #-1,DRVSTA(R1)      ;DRIVE IS UNSAFE
040410 004737 042170                  JSR     PC,SVRH11           ;SAVE RH11/RP04/5/6 REGISTERS
040414 052762 110000 000016           BIS     #BIT15!BIT12,16(R2)      ;INFORM USER OF UNSAFE ERROR
040422 000423                         BR      SC7
040424 032705 010000          2$:     BIT     #BIT12,R5           ;''MOL'' = 1 ?
040430 001015                         BNE     3$                  ;BR IF YES
040432 112761 177777 034344           MOVB    #-1,DRVACT(R1)      ;ACTIVE ERROR RECOVER
040440 112761 000001 034354           MOVB    #1,DRVSTA(R1)       ;ONLINE
040446 006301                         ASL     R1
040450 012761 072460 034446           MOV     #30000.,TIMER(R1) ;START 30 SECOND TIMER
040456 006201                         ASR     R1
040460 000137 040032                  JMP     SC4
040464 052762 100220 000016   3$:     BIS     #BIT15!BIT07!BIT04,16(R2)         ;INFORM USER OF ERROR
040472 105061 034344          SC7:    CLRB    DRVACT(R1)          ;DRIVE IS IDLE
040476 004737 042730                  JSR     PC,EMPTYQ           ;DUMP THE QUEUE
040502 105761 034422                  TSTB    ULDFLG(R1)          ;UNLOAD IN PROGRESS OR QUEUE?
040506 003002                         BGT     1$                  ;BR IF NOT
040510 105061 034422                  CLRB    ULDFLG(R1)          ;CLEAR UNLOAD FLAG
040514 116164 034470 000016   1$:     MOVB    ATABIT(R1),RPAS(R4)  ;CLEAR ATTENTION BIT
040522 105761 034354                  TSTB    DRVSTA(R1)          ;IS THE DRIVE UNSAFE ?
040526 100406                         BMI     2$                  ;BR IF IT IS
040530 012746 000113                  MOV     #113,-(SP)          ;RELEASE COMMAND
040534 004037 042002                  JSR     R0,WRT.RP           ;WRITE THE COMMAND INTO RPCS1
040540 000000                         RPCS1                       ;REGISTER INDEX
040542 040550                         SC8                         ;PARITY EXIT ADDRESS
040544 000137 040032          2$:     JMP     SC4                 ;CHECK FOR MORE DRIVES
040550 105761 034344          SC8:    TSTB    DRVACT(R1)          ;IS DRIVE IDLE?
040554 001405                         BEQ     1$                  ;YES--BRANCH
040556 004737 043024                  JSR     PC,GETREQ           ;GET DPB POINTER
040562 004737 036742                  JSR     PC,CI7              ;PROCESS THE PARITY ERROR
040566 000402                         BR      2$                  ;CONTINUE
040570 004737 036770          1$:     JSR     PC,CI7B             ;PROCESS THE UNCORRECTABLE PARITY ERROR
040574 000137 040032          2$:     JMP     SC4                 ;CHECK MORE DRIVES
040600 105761 034422          SC11:   TSTB    ULDFLG(R1)          ;''UNLOAD IN PROGRESS''?
040604 003402                         BLE     1$                  ;BRANCH IF NO
```

```
040606  105061  034422                      CLRB    ULDFLG(R1)              ;CLEAR UNLOAD FLAG
040612  105061  034344              1$:     CLRB    DRVACT(R1)              ;SET DRIVE IDLE
040616  136137  034470  034416              BITB    ATABIT(R1),SRCHWT   ;DOING A SEARCH OPERATION FOR
                                                                            ;AN I/O COMMAND?
040624  001012                              BNE     2$                      ;BRANCH IF YES
040626  004737  043046                      JSR     PC,POPQUE               ;REMOVE REQUEST FROM QUEUE
040632  052762  000200  000016              BIS     #BIT07,16(R2)           ;SET 'DONE'' BIT
040640  005737  034442                      TST     SAVEFG                  ;SAVE THE REGISTERS?
040644  100002                              BPL     2$                      ;BRANCH IF NO
040646  004737  042170                      JSR     PC,SVRH11               ;YES--SAVE ALL OF THE RH11/RP04/5/6 REG'S
040652  116164  034470  000016      2$:     MOVB    ATABIT(R1),RPAS(R4)   ;CLEAR ATTENTION BIT
040660  004737  035632                      JSR     PC,OPT                  ;START A REQUEST
040664  000137  040032                      JMP     SC4                     ;CHECK FOR MORE DRIVES
040670  010164  000010              SC12:   MOV     R1,RPCS2(R4)            ;SELECT DRIVE
040674  016437  000012  034334              MOV     RPDS1(R4),RPERRS                ;SAVE THE FOUR REGISTERS THAT
040702  016437  000014  034336              MOV     RPER1(R4),RPERRS+2              ;WILL TELL US SOMETHING
040710  016437  000040  034340              MOV     RPER2(R4),RPERRS+4
040716  016437  000042  034342              MOV     RPER3(R4),RPERRS+6
040724  004037  034732                      JSR     R0,DRVINT               ;INIT. THE STATE OF THE DRIVE
040730  000401                              BR      1$                      ;TAKE ERROR EXIT
040732  000207                              RTS     PC                      ;RETURN
040734  005726                      1$:     TST     (SP)+                   ;POP PC OFF OF THE STACK
040736  000704                              BR      SC8                     ;PROCESS THE PARITY ERROR
040740  006301                      SC13:   ASL     R1                      ;SETUP TO ADDRESS WORDS
040742  012761  177777  034446              MOV     #-1,TIMER(R1)           ;STOP THE TIMER
040750  006201                              ASR     R1                      ;
040752  010164  000010                      MOV     R1,RPCS2(R4)            ;SELECT THE DRIVE
040756  116164  034470  000016              MOVB    ATABIT(R1),RPAS(R4)   ;CLEAR THE ATTENTION BIT
040764  032714  004000                      BIT     #BIT11,(R4)             ;DRIVE AVAILABLE ?
040770  001006                              BNE     1$                      ;BR IF AVAILABLE
040772  006301                              ASL     R1
040774  012761  023420  034446              MOV     #10000.,TIMER(R1)   ;START 10 SEC TIMER AGAIN
041002  006201                              ASR     R1
041004  000433                              BR      3$                      ;EXIT
041006  105761  034374              1$:     TSTB    DPINT(R1)               ;INITIALIZING THE DRIVE ?
041012  001424                              BEQ     2$                      ;BR IF NOT
041014  105061  034374                      CLRB    DPINT(R1)               ;CLEAR THE INIT INDICATOR
041020  004037  034732                      JSR     R0,DRVINT               ;GO INIT THE DRIVE
041024  000240                              NOP                             ;DUMMY PARITY ERROR RETURN
041026  105761  034354                      TSTB    DRVSTA(R1)              ;DRIVE ONLINE ?
041032  003014                              BGT     2$                      ;BR IF YES -- START ORDER
041034  005702                              TST     R2                      ;QUEUE ENTRY FOR THE DRIVE
041036  001416                              BEQ     3$                      ;BR IF NOT
041040  004737  043024                      JSR     PC,GETREQ               ;GET DPB ADDRESS
041044  052762  140000  000016              BIS     #BIT15!BIT14,16(R2)   ;INFORM USER THAT DRIVE OFFLINE
041052  004737  042170                      JSR     PC,SVRH11               ;SAVE THE REGISTERS
041056  004737  042730                      JSR     PC,EMPTYQ               ;EMPTY THE REQUEST QUEUE
041062  000404                              BR      3$
041064  105061  034404              2$:     CLRB    DPRQS(R1)               ;CLEAR THE PORT REQUEST INDICATOR
041070  004737  035632                      JSR     PC,OPT                  ;START THE PENDING REQUEST
041074  000137  040032              3$:     JMP     SC4                     ;PROCESS OTHER DRIVES

                                    ;RP04/5/6 TIMER ROUTINE
                                    ;CALL
                                    ;       MOV     #TIME,-(SP)             ;ELASPED TIME IN MILLISECONDS ON THE STACK
                                    ;       JSR     PC,RPTMR                ;CALL RP04/5/6 TIME ROUTINE

```

```
041100  005737  034420          RPTMR:  TST     ACTDRV                  ;CHECK "ACTDRV & ACTSTR"
041104  001030                          BNE     4$                      ;IF NON ZERO EXIT
041106  112737  000001  034421          MOVB    #1,ACTSTR               ;SET "ACTSTR"
041114  104412                          SAVREG                          ;SAVE R0 - R5
041116  005001                          CLR     R1                      ;START WITH DRIVE 0
041120  005003                          CLR     R3
041122  005763  034446          1$:     TST     TIMER(R3)               ;IS THE TIMER RUNNING?
041126  002407                          BLT     2$                      ;BRANCH IF NO
041130  166663  000002  034446          SUB     2(SP),TIMER(R3)         ;COUNT THE INTERVAL
041136  003003                          BGT     2$                      ;BR IF NO SOFTWARE TIMEOUT
041140  004737  041172                  JSR     PC,STO                  ;CALL SOFTWARE TIMEOUT ROUTINE
041144  000405                          BR      3$                      ;GO TO THE EXIT
041146  005201                  2$:     INC     R1                      ;MOVE TO NEXT DRIVE
041150  005723                          TST     (R3)+
041152  022701  000010                  CMP     #8.,R1                  ;OUT OF DRIVES?
041156  003361                          BGT     1$                      ;BRANCH IF NO
041160  104413                  3$:     RESREG                          ;RESTORE R0 - R5
041162  105037  034421                  CLRB    ACTSTR                  ;ZERO ACTIVE SOFTWARE TIMEOUT ROUTINE FLAG
041166  012616                  4$:     MOV     (SP)+,(SP)              ;ADJUST THE STACK
041170  000207                          RTS     PC                      ;RETURN

                                ;SOFTWARE TIMEOUT ROUTINE
                                ;
                                ;NOTE:  THIS ROUTINE MUST BE ENTERED AT PRIORITY 6
                                ;       OR GREATER
                                ;
                                ;CALL:  STO
                                ;       MOV     #DRVNUM,R1              ;DRIVE NUMBER
                                ;       JSR     PC,STO                  ;CALL
                                ;       RETURN

041172  010146                  STO:    MOV     R1,-(SP)                ;SAVE R1
041174  010346                          MOV     R3,-(SP)                ;SAVE R3
041176  013704  034502                  MOV     RPADR,R4                ;GET ADDRESS OF "RPCS1"
041202  010164  000010                  MOV     R1,RPCS2(R4)            ;SELECT THE DRIVE
041206  004037  041626                  JSR     R0,RD.RP                ;READ "DRIVE STATUS REG"
041212  000012                          RPDS1
041214  041514                          STO5
041216  105726                          TSTB    (SP)+                   ;IS "DRY"=1?
041220  100477                          BMI     STO2                    ;BR IF YES
041222  105761  034374          STO1:   TSTB    DPINT(R1)               ;TRYING TO INTIALIZE THE DRIVE ?
041226  001074                          BNE     STO2                    ;BR IF YES
041230  105761  034404                  TSTB    DPRQS(R1)               ;OUTSTANDING PORT REQUEST FOR THE DRIVE ?
041234  001071                          BNE     STO2                    ;BR IF YES
041236  013702  034414                  MOV     TRNSWT,R2               ;PICKUP TRANSFER WAIT QUEUE
041242  020137  034466                  CMP     R1,DTUW                 ;TRANSFER UNDERWAY ON THIS DRIVE?
041246  001402                          BEQ     1$                      ;BRANCH IF YES
041250  004737  043024                  JSR     PC,GETREQ               ;GET DPB ADDRESS
041254  052762  101000  000016  1$:     BIS     #BIT15!BIT09,16(R2)     ;SET THE ERROR FLAGS
041262  004737  042170                  JSR     PC,SVRH11               ;SAVE RH11/RP04/5/6 REGISTERS
041266  012764  000040  000010          MOV     #BIT05,RPCS2(R4)        ;"INIT" THE MASS BUS
041274  105061  034344                  CLRB    DRVACT(R1)              ;DRIVE IS IDLE
041300  105061  034422                  CLRB    ULDFLG(R1)              ;CLEAR THE UNLOAD FLAG
041304  005001                          CLR     R1                      ;START WITH DRIVE 0
041306  005003                          CLR     R3
041310  004037  034732          2$:     JSR     R0,DRVINT               ;INIT. THIS DRIVE
041314  000477                          BR      STO5                    ;PARITY ERROR RETURN
```

```
041316  105761  034344                    TSTB    DRVACT(R1)        ;DRIVE IDLE BEFORE THE INIT.?
041322  001414                            BEQ     4$                ;YES--BRANCH
041324  013702  034414                    MOV     TRNSWT,R2         ;GET TRANSFER WAIT QUEUE
041330  023701  034466                    CMP     DTUW,R1           ;WAS THERE I/O ON THIS DRIVE?
041334  001402                            BEQ     3$                ;YES--BRANCH
041336  004737  043024                    JSR     PC,GETREQ         ;GET THE DPB POINTER FROM QUEUE
041342  052762  100400  000016  3$:       BIS     #BIT15!BIT08,16(R2) ;INFORM USER OF INIT.
041350  105061  034344                    CLRB    DRVACT(R1)        ;SET DRIVE ACTIVE TO IDLE
041354  105061  034422          4$:       CLRB    ULDFLG(R1)        ;NO UNLOAD
041360  012763  177777  034446            MOV     #-1,TIMER(R3)     ;STOP THE TIMER
041366  005723                            TST     (R3)+             ;UPDATE THE INDEX
041370  005201                            INC     R1                ;INCREMENT THE DRIVE NUMBER
041372  022701  000010                    CMP     #8.,R1            ;LAST DRIVE BEEN CHECKED?
041376  003344                            BGT     2$                ;NO--LOOP
041400  012737  177777  034466            MOV     #-1,DTUW          ;NO DATA TRANSFERS UNDERWAY
041406  005037  034414                    CLR     TRNSWT            ;CLEAR TRANSFER WAIT QUEUE
041412  004737  042652                    JSR     PC,CLRQUE         ;CLEAR ALL REQUEST QUEUES
041416  000500                            BR      ST09              ;EXIT
041420  116405  000016          ST02:     MOVB    RPAS(R4),R5       ;READ ATTENTION REG
041424  136105  034470                    BITB    ATABIT(R1),R5     ;IS ATTENTION FOR THIS DRIVE UP ?
041430  001017                            BNE     ST03              ;YES--BRANCH
041432  105761  034374                    TSTB    DPINT(R1)         ;TRYING TO INTIALIZE THE DRIVE ?
041436  001031                            BNE     ST06              ;BR IF YES - DRIVE NOT ONLINE
041440  105761  034404                    TSTB    DPRQS(R1)         ;OUTSTANDING PORT REQUEST FOR THE DRIVE ?
041444  001045                            BNE     ST07              ;BR IF YES - NO RESPONSE TO REQUEST
041446  020137  034466                    CMP     R1,DTUW           ;DATA TRANSFER UNDERWAY FOR THIS DRIVE
041452  001263                            BNE     ST01              ;BR IF NO
041454  004037  041626                    JSR     R0,RD.RP          ;YES--CHECK 'RDY''
041460  000000                            RPCS1
041462  041514                            ST05
041464  105726                            TSTB    (SP)+
041466  100255                            BPL     ST01              ;BR IF 'RDY''=0
041470  105761  034374          ST03:     TSTB    DPINT(R1)         ;INITIALIZING THE DRIVE ?
041474  001003                            BNE     1$                ;BR IF INIT PENDING
041476  105761  034404                    TSTB    DPRQS(R1)         ;PORT REQUEST PENDING ?
041502  001446                            BEQ     ST09              ;BR IF NOT
041504  012763  177777  034446  1$:       MOV     #-1,TIMER(R3)     ;STOP THE TIMER
041512  000442                            BR      ST09              ;EXIT
041514  004737  037042          ST05:     JSR     PC,CI8            ;GO HANDLE THE PARITY ERROR
041520  000437                            BR      ST09
041522  105061  034374          ST06:     CLRB    DPINT(R1)         ;CLEAR THE INITIALIZE INDICATOR
041526  105061  034354                    CLRB    DRVSTA(R1)        ;SET UNIT OFFLINE
041532  012763  177777  034446            MOV     #-1,TIMER(R3)   ;STOP THE TIMER
041540  004737  043024                    JSR     PC,GETREQ         ;GET THE DPB ADDRESS
041544  005702                            TST     R2                ;REQUEST IN QUEUE ?
041546  001424                            BEQ     ST09              ;BR IF NOT
041550  052762  140000  000016            BIS     #BIT15!BIT14,16(R2)   ;INFORM THE USER DRIVE NOT AVAILABLE
041556  000414                            BR      ST08              ;FINISH
041560  012763  177777  034446  ST07:     MOV     #-1,TIMER(R3)     ;STOP THE TIMER
041566  105061  034404                    CLRB    DPRQS(R1)         ;CLEAR PORT REQUEST INDICATOR
041572  004737  043024                    JSR     PC,GETREQ         ;GET DPB ADDRESS
041576  005702                            TST     R2                ;QUEUE ENTRY FOR DRIVE ?
041600  001407                            BEQ     ST09              ;BR IF NONE
041602  012762  100004  000016            MOV     #BIT15!BIT2,16(R2)   ;INFORM USER OF PORT REQUEST ERROR
041610  004737  042730          ST08:     JSR     PC,EMPTYQ         ;CLEAR THE QUEUE FOR THE DRIVE
041614  004737  04217[illegible]                    JSR     PC,SVRH11         ;SAVE THE REGISTERS
041620  012603                  ST09:     MOV     (SP)+,R3          ;RESTORE R3
```

```
041622  012601                              MOV     (SP)+,R1        ;RESTORE R1
041624  000207                              RTS     PC              ;RETURN

                                    ;ROUTINE TO READ A RH11/RP04/5/6 REGISTER
                                    ;
                                    ;CALL
                                    ;       JSR     R0,RD.RP        ;GO READ A REGISTER
                                    ;       INDEX                   ;REG. INDEX FROM BASE
                                    ;       ERRADR                  ;ERROR ADDRESS--PROCESS ERROR STARTING
                                    ;                               ;AT THIS ADDRESS
                                    ;       RETURN                  ;CONTENTS OF REG. IS ON THE STACK
                                    ;
041626  013737  034500  041770      RD.RP:  MOV     MCPEMX,RD.RP2   ;MAX. RETRYS ALLOWED
041634  011646                              MOV     (SP),-(SP)      ;SAVE R0 FOR RETURN
041636  013737  034502  041652              MOV     RPADR,RD.ADR    ;FORM THE DESIRED ADDRESS
041644  062037  041652                      ADD     (R0)+,RD.ADR    ;USING THE BASE AND THE INDEX
041650  013727                      RD.RP1: MOV     @(PC)+,(PC)+    ;READ THE DESIRED REGISTER OF THE RP04
041652  000000                      RD.ADR: .WORD   0               ;ADDRESS IS FORMED HERE
041654  000000                      RD.WRD: .WORD   0               ;REG. CONTENTS PUT HERE
041656  013766  041654  000002              MOV     RD.WRD,2(SP)    ;RETURN IT TO THE USER
041664  013746  034502                      MOV     RPADR,-(SP)     ;PUT THE ADDRESS ON THE STACK
041670  062716  000010                      ADD     #RPCS2,(SP)     ;FORM THE ADDRESS OF RPCS2
041674  032736  010000                      BIT     #BIT12,@(SP)+   ;CHECK THE 'NED' BIT
041700  001035                              BNE     RD.RP3          ;BR IF DRIVE NON-EXISTENT
041702  017746  172574                      MOV     @RPADR,-(SP)    ;READ RPCS1
041706  032716  020000                      BIT     #BIT13,(SP)     ;DID MCPE SET?
041712  001002                              BNE     1$              ;BRANCH IF YES
041714  022620                              CMP     (SP)+,(R0)+     ;ADJUST FOR RETURN
041716  000430                              BR      RD.RP4          ;EXIT
041720                              1$:
041720  104003                              EMT     3
041722  005737  034466                      TST     DTUW            ;DATA TRANSFER UNDERWAY?
041726  100405                              BMI     2$              ;NO--BRANCH
041730  032716  040000                      BIT     #BIT14,(SP)     ;NO--''TRE''=1?
041734  001402                              BEQ     2$              ;NO--BRANCH
041736  005726                              TST     (SP)+           ;YES--CLEAN OFF THE STACK AND
041740  000415                              BR      RD.RP3          ;TAKE THE FATAL ERROR EXIT
041742  052716  040000              2$:     BIS     #BIT14,(SP)     ;CLEAR 'MCPE'' BY SENDING A ''1'' TO ''TRE''
041746  000316                              SWAB    (SP)            ;POSITION BEFORE WRITING
041750  013737  034502  041764              MOV     RPADR,3$        ;FORM ADDRESS OF HIGH BYTE
041756  005237  041764                      INC     3$
041762  112637                              MOVB    (SP)+,@(PC)+    ;WRITE THE HIGH BYTE OF RPCS1
041764  000000                      3$:     .WORD   0               ;ADDRESS STORAGE
041766  005327                              DEC     (PC)+           ;EXCEEDED MAX. RETRYS
041770  000003                      RD.RP2: .WORD   3
041772  002326                              BGE     RD.RP1          ;BRANCH IF NO
041774  011000                      RD.RP3: MOV     (R0),R0         ;FATAL ERROR EXIT
041776  012616                              MOV     (SP)+,(SP)
042000  000200                      RD.RP4: RTS     R0

                                    ;ROUTINE TO WRITE A REGISTER
                                    ;
                                    ;CALL
                                    ;       MOV     DATA,-(SP)      ;DATA TO BE LOADED ON THE STACK
                                    ;       JSR     R0,WRT.RP       ;CALL THE ROUTINE TO LOAD(WRITE) THE REG.
                                    ;       INDEX                   ;INDEX OF THE REGISTER TO BE LOADED
                                    ;       ERRADR                  ;ADDRESS TO RETURN TO ON AN ERROR
```

```
                                        ;         RETURN                          ;ERROR FREE RETURN

 042002  013737  034500  042152  WRT.RP:  MOV     MCPEMX,WRT.R2            ;MAX RETRYS ALLOWED
 042010  016637  000002  042070           MOV     2(SP),WRT.WD             ;SAVE THE WORD TO WRITE
 042016  012616                           MOV     (SP)+,(SP)               ;ADJUST THE STACK
 042020  012037  042072                   MOV     (R0)+,WRT.AD             ;GET INDEX OF REGISTER TO BE WRITTEN
 042024  001015                           BNE     1$                       ;BRANCH IF NOT RPCS1
 042026  122737  000150  042070           CMPB    #150,WRT.WD              ;IS THE COMMAND FOR DATA TRANSFERS?
 042034  002411                           BLT     1$                       ;YES--DON'T GET THE OLD A16 & A17, & PSEL
 042036  004037  041626                   JSR     R0,RD.RP                 ;NO---COMBINE A16&A17, & PSEL WITH
 042042  000000                           RPCS1                            ;THE COMMAND BEFORE SENDING IT TO
 042044  042160                           WRT.R3                           ;THE RH11/RP04
 042046  000316                           SWAB    (SP)
 042050  042716  177770                   BIC     #^C7,(SP)
 042054  112637  042071                   MOVB    (SP)+,WRT.WD+1
 042060  063737  034502  042072  1$:      ADD     RPADR,WRT.AD             ;FORM THE ADDRESS OF THE DISK REG.
 042066  012737                  WRT.R1:  MOV     (PC)+,@(PC)+             ;LOAD THE DESIRED REG.
 042070  000000                  WRT.WD:  .WORD   0                        ;WORD TO WRITE GOES HERE
 042072  000000                  WRT.AD:  .WORD   0                        ;ADDRESS IS FORMED HERE
 042074  013746  034502                   MOV     RPADR,-(SP)              ;PUT THE ADDRESS ON THE STACK
 042100  062716  000010                   ADD     #RPCS2,(SP)              ;FORM THE ADDRESS OF RPCS2
 042104  032736  010000                   BIT     #BIT12,@(SP)+            ;CHECK THE 'NED' BIT
 042110  001023                           BNE     WRT.R3                   ;BR IF DRIVE NON-EXISTENT
 042112  004037  041626                   JSR     R0,RD.RP                 ;CHECK FOR PARITY ERROR ON WRITE
 042116  000014                           RPER1
 042120  042160                           WRT.R3
 042122  032726  000010                   BIT     #BIT03,(SP)+
 042126  001416                           BEQ     WRT.R4                   ;BRANCH IF "PAR=0"
 042130  016037  177776  042142           MOV     -2(R0),1$                ;PICKUP THE INDEX
 042136  004037  041626                   JSR     R0,RD.RP                 ;READ THE REG.
 042142  000000                  1$:      .WORD   0                        ;REG. INDEX
 042144  042160                           WRT.R3                           ;RETURN TO THIS ADDRESS ON ERROR
 042146  104004                           EMT     4
 042150  005327                           DEC     (PC)+                    ;DECREMENT THE ERROR COUNT
 042152  000003                  WRT.R2:  .WORD   3                        ;RETRY COUNTER
 042154  002344                           BGE     WRT.R1                   ;TRY AGAIN IF NOT FINISHED
 042156  005726                           TST     (SP)+                    ;CLEAN OFF THE STACK
 042160  011000                  WRT.R3:  MOV     (R0),R0                  ;TAKE THE "PARITY ON WRITE" ERROR EXIT
 042162  000401                           BR      WRT.R5                   ;EXIT
 042164  005720                  WRT.R4:  TST     (R0)+                    ;ADJUST FOR ERROR FREE EXIT
 042166  000200                  WRT.R5:  RTS     R0

                                 ;ROUTINE TO SAVE THE RH11/RP04/5/6 REGISTERS AS PER DPB+14
                                 ;
                                 ;CALL
                                 ;        MOV     #DPBNUM,R2               ;DPB POINTER TO R2
                                 ;        JSR     PC,SVRH11                ;SAVE THE DRIVES REG'S

 042170  104412                  SVRH11:  SAVREG                           ;SAVE R0 - R5
 042172  005702                           TST     R2                       ;QUEUE ENTRY FOR THE DRIVE ?
 042174  001430                           BEQ     4$                       ;BR IF NONE
 042176  013704  034502                   MOV     RPADR,R4
 042202  111264  000010                   MOVB    (R2),RPCS2(R4)           ;SELECT DRIVE
 042206  016203  000014                   MOV     14(R2),R3                ;GET THE ERROR TABLE POINTER
 042212  001433                           BEQ     6$                       ;EXIT IF NO ADDRESS
 042214  005037  042250                   CLR     3$                       ;COUNTER & POINTER
 042220  023727  042250  000022  1$:      CMP     3$,#RPDB                 ;REACHED THE BUFFER REGISTER ?
```

```
042226  001006                            BNE     2$                ;BR IF NOT
042230  032764  000200  000010            BIT     #BIT07,RPCS2(R4)  ;'OR' SET ?
042236  001002                            BNE     2$                ;BR IF SET
042240  005023                            CLR     (R3)+             ;STORE RPDB AS ZEROES
042242  000405                            BR      4$                ;CONTINUE
042244  004037  041626            2$:     JSR     R0,RD.RP          ;READ THE SELECTED REGISTER
042250  000000                    3$:     .WORD   0                 ;REGISTER INDEX
042252  042276                                    5$                ;ERROR RETURN ADDRESS
042254  012623                            MOV     (SP)+,(R3)+       ;STORE THE REGISTER CONTENTS
042256  023727  042250  000046    4$:     CMP     3$,#RPEC2         ;REACHED THE END ?
042264  001406                            BEQ     6$                ;BR IF YES
042266  062737  000002  042250            ADD     #2,3$             ;INCREMENT THE REGISTER INDEX
042274  000751                            BR      1$                ;CONTINUE READING THE REGISTERS
042276  004737  036742            5$:     JSR     PC,CI7            ;PROCESS THE UNCORRECTABLE PARITY ERROR
042302  104413                    6$:     RESREG                    ;RESTORE R0 - R5
042304  000207                            RTS     PC                ;RETURN

                                  ;ROUTINE TO SET THE INTERRUPT WITHOUT GETTING A "TRE"
                                  ;CALL
                                  ;       MOV     #DRVNUM,R1        ;DRIVE NUMBER TO R1
                                  ;       JSR     PC,SET.IE         ;SET "IE"
                                  ;       RETURN

042306  010446                    SET.IE: MOV     R4,-(SP)          ;SAVE R4
042310  013704  034502                    MOV     RPADR,R4          ;PICKUP ADDRESS OF RPCS1
042314  010164  000010                    MOV     R1,RPCS2(R4)      ;SELECT DRIVE
042320  011446                            MOV     (R4),-(SP)        ;READ RPCS1
042322  052716  040000                    BIS     #BIT14,(SP)       ;SET THE "TRE" BIT OF THE WORD READ
042326  000316                            SWAB    (SP)              ;ADJUST FOR DATO
042330  112714  000100                    MOVB    #BIT06,(R4)       ;SET "IE"
042334  032764  010000  000010            BIT     #BIT12,RPCS2(R4)  ;IS "NED"=1?
042342  001002                            BNE     1$                ;YES--CLEAR "TRE"
042344  005726                            TST     (SP)+             ;CLEAN OFF THE STACK
042346  000402                            BR      2$
042350  112664  000001            1$:     MOVB    (SP)+,1(R4)       ;CLEAR "TRE"
042354  012604                    2$:     MOV     (SP)+,R4          ;RESTORE R4
042356  000207                            RTS     PC                ;RETURN TO CALLER

                                  ;QUEUE COUNT
042360     000                    QCNT:   .BYTE   0                 ;DRIVE 0
042361     000                            .BYTE   0                 ;DRIVE 1
042362     000                            .BYTE   0                 ;DRIVE 2
042363     000                            .BYTE   0                 ;DRIVE 3
042364     000                            .BYTE   0                 ;DRIVE 4
042365     000                            .BYTE   0                 ;DRIVE 5
042366     000                            .BYTE   0                 ;DRIVE 6
042367     000                            .BYTE   0                 ;DRIVE 7

                                  ;QUEUE INPUT POINTERS

042370  042452                    QINPT:  .WORD   QDRV0             ;DRIVE 0
042372  042472                            .WORD   QDRV1             ;DRIVE 1
042374  042512                            .WORD   QDRV2             ;DRIVE 2
042376  042532                            .WORD   QDRV3             ;DRIVE 3
042400  042552                            .WORD   QDRV4             ;DRIVE 4
042402  042572                            .WORD   QDRV5             ;DRIVE 5
042404  042612                            .WORD   QDRV6             ;DRIVE 6
```

```
042406  042632                          .WORD   QDRV7           ;DRIVE 7

                                ;QUEUE OUTPUT POINTERS

042410  042452                  QOUTPT: .WORD   QDRV0           ;DRIVE 0
042412  042472                          .WORD   QDRV1           ;DRIVE 1
042414  042512                          .WORD   QDRV2           ;DRIVE 2
042416  042532                          .WORD   QDRV3           ;DRIVE 3
042420  042552                          .WORD   QDRV4           ;DRIVE 4
042422  042572                          .WORD   QDRV5           ;DRIVE 5
042424  042612                          .WORD   QDRV6           ;DRIVE 6
042426  042632                          .WORD   QDRV7           ;DRIVE 7

042430  042452                  QSTART: .WORD   QDRV0           ;DRIVE 0 START ADDRESS
042432  042472                  QSTOP:  .WORD   QDRV1           ;DRIVE 0 STOP ADDRESS & DRIVE 1 START ADDRESS
042434  042512                          .WORD   QDRV2           ;STOP DRIVE 1--START DRIVE 2
042436  042532                          .WORD   QDRV3           ;STOP DRIVE 2--START DRIVE 3
042440  042552                          .WORD   QDRV4           ;STOP DRIVE 3--START DRIVE 4
042442  042572                          .WORD   QDRV5           ;STOP DRIVE 4--START DRIVE 5
042444  042612                          .WORD   QDRV6           ;STOP DRIVE 5--START DRIVE 6
042446  042632                          .WORD   QDRV7           ;STOP DRIVE 6--START DRIVE 7
042450  042652                          .WORD   QTERM           ;STOP DRIVE 7

                                ;DRIVE REQUEST QUEUES

042452                          QDRV0:  .BLKW   10
042472                          QDRV1:  .BLKW   10
042512                          QDRV2:  .BLKW   10
042532                          QDRV3:  .BLKW   10
042552                          QDRV4:  .BLKW   10
042572                          QDRV5:  .BLKW   10
042612                          QDRV6:  .BLKW   10
042632                          QDRV7:  .BLKW   10
        042652                  QTERM=.

                                ;ROUTINE TO CLEAR ALL OF THE REQUEST QUEUES
                                ;
                                ;CALL
                                ;       JSR     PC,CLRQUE

042652  104412                  CLRQUE: SAVREG                  ;SAVE R0 - R5
042654  012702  042360                  MOV     #QCNT,R2        ;ZERO THE QUEUE COUNTS
042660  005022                          CLR     (R2)+           ;DRIVES 0 & 1
042662  005022                          CLR     (R2)+           ;DRIVES 2 & 3
042664  005022                          CLR     (R2)+           ;DRIVES 4 & 5
042666  005022                          CLR     (R2)+           ;DRIVES 6 & 7
042670  012703  000010                  MOV     #8.,R3          ;MOVE THE STARTING
042674  012701  042430                  MOV     #QSTART,R1      ;ADDRESS OF THE QUEUE INTO
042700  012122                  1$:     MOV     (R1)+,(R2)+     ;THE QUEUE INPUT POINTER
042702  005303                          DEC     R3
042704  001375                          BNE     1$
042706  012703  000010                  MOV     #8.,R3          ;MOVE THE STARTING ADDRESS
042712  012701  042430                  MOV     #QSTART,R1      ;OF THE QUEUE INTO THE
042716  012122                  2$:     MOV     (R1)+,(R2)+     ;QUEUE OUTPUT POINTER
042720  005303                          DEC     R3
042722  001375                          BNE     2$
042724  104413                          RESREG                  ;RESTORE R0 - R5
```

```
042726  000207                          RTS     PC

                                ;EMPTY THE QUEUE SPECIFIED BY R1
                                ;
                                ;CALL
                                ;       MOV     DRVNUM,R1       ;DRIVE NUMBER TO R1
                                ;       JSR     PC,EMPTYQ

042730  105061  042360          EMPTYQ: CLRB    QCNT(R1)        ;CLEAR NUMBER OF ITEMS IN QUEUE
042734  006301                          ASL     R1
042736  016161  042370  042410          MOV     QINPT(R1),QOUTPT(R1) ;SET OUTPUT QUEUE POINTER=INPUT POINTER
042744  006201                          ASR     R1
042746  000207                          RTS     PC

                                ;ROUTINE TO PUT A REQUEST IN QUEUE
                                ;
                                ;CALL
                                ;       MOV     #DRVNUM,R1      ;DRIVE NUMBER
                                ;       MOV     #DPB,R2         ;ADDRESS OF PARAMETER BLOCK
                                ;       JSR     R0,DRVQUE       ;GO PUT REQUEST IN QUEUE
                                ;       RETURN1                 ;RETURN HERE IF QUEUE IS FULL
                                ;       RETURN2                 ;RETURN HERE IF REQUEST IS IN QUEUE

042750  122761  000010  042360  DRVQUE: CMPB    #10,QCNT(R1)    ;IS QUEUE FULL?
042756  001421                          BEQ     2$              ;BR IF YES-TAKE RETURN1
042760  105261  042360                  INCB    QCNT(R1)        ;INCREMENT QUEUE COUNT
042764  006301                          ASL     R1
042766  010271  042370                  MOV     R2,@QINPT(R1)   ;PUT THIS REQUEST IN QUEUE
042772  062761  000002  042370          ADD     #2,QINPT(R1)    ;UPDATE THE QUEUE POINTER
043000  026161  042370  042432          CMP     QINPT(R1),QSTOP(R1) ;TIME TO RESET THE POINTER
043006  001003                          BNE     1$              ;BRANCH IF NO
043010  016161  042430  042370          MOV     QSTART(R1),QINPT(R1) ;YES--RESET POINTER
043016  006201                  1$:     ASR     R1
043020  005720                          TST     (R0)+           ;TAKE RETURN 2
043022  000200                  2$:     RTS     R0              ;RETURN TO USER

                                ;ROUTINE TO GET THE "DPB" ADDRESS OF NEXT REQUEST IN QUEUE
                                ;
                                ;CALL
                                ;       MOV     #DRVNUM,R1      ;DRIVE NUMBER TO R1
                                ;       JSR     PC,GETREQ       ;GO GET THE REQUEST
                                ;       RETURN                  ;R2="DPB" ADDRESS OF THE REQUEST
                                ;                               ;R2=0 IF NO REQUEST IN QUEUE

043024  005002                  GETREQ: CLR     R2
043026  105761  042360                  TSTB    QCNT(R1)        ;IS THERE ANY REQUEST IN QUEUE?
043032  001404                          BEQ     2$              ;NO---BRANCH
043034  006301                  1$:     ASL     R1
043036  017102  042410                  MOV     @QOUTPT(R1),R2  ;PICKUP "DPB" POINTER FOR THIS DRIVE
043042  006201                          ASR     R1
043044  000207                  2$:     RTS     PC              ;RETURN TO USER

                                ;ROUTINE TO "POP" THE REQUEST FROM QUEUE
                                ;
                                ;CALL
                                ;       MOV     #DRVNUM,R1      ;DRIVE NUMBER TO R1
                                ;       JSR     PC,POPQUE       ;CALL TO REMOVE REQUEST
```

```
  1436                                          ;       RETURN          ;R2=ADDRESS OF DPB REMOVED
  1437
  1438 043046   105361  042360          POPQUE: DECB    QCNT(R1)        ;DECREMENT QUEUE COUNT
  1439 043052   006301                          ASL     R1
  1440 043054   017102  042410                  MOV     @QOUTPT(R1),R2  ;GET THE 'DPB'' POINTER
  1441 043060   062761  000002  042410          ADD     #2,QOUTPT(R1)   ;UPDATE THE QUEUE POINTER
  1442 043066   026161  042410  042432          CMP     QOUTPT(R1),QSTOP(R1) ;TIME TO RESET THE POINTER?
  1443 043074   001003                          BNE     1$              ;NO--BRANCH TO EXIT
  1444 043076   016161  042430  042410          MOV     QSTART(R1),QOUTPT(R1) ;YES--RESET THE POINTER
  1445 043104   006201                  1$:     ASR     R1
  1446 043106   000207                          RTS     PC              ;RETURN TO USER
  1447
```

```
                                .SBTTL  DATA, CONTROL, & STATUS BLOCKS

                                ;BLOCK LOCATION EQUATE STATEMENTS

        000001                  $FMT    =       1               ;FMT,HCI,ECI OR OFFSET CODE
        000002                  $COMND  =       $FMT+1          ;OPERATION CODE
        000003                  $PSEL   =       $FMT+2          ;PORT SELECT & BITS A16, A17
        000004                  $WRDM   =       $FMT+3          ;WORD COUNT (2'S COMP)
        000006                  $BUF    =       $FMT+5          ;BUFFER ADDR OR REGISTER TABLE POINTER
        000010                  $SEC    =       $FMT+7          ;SECTOR ADDRESS OR 1ST REG ADDR
        000011                  $TRK    =       $FMT+10         ;TRACK ADDRESS OF LAST REG ADDR
        000012                  $CYL    =       $FMT+11         ;CYLINDER ADDR
        000014                  $REG    =       $FMT+13         ;REGISTER STORAGE (IF ERROR)
        000016                  $STATUS =       $FMT+15         ;STATUS WORD (SET BY DRIVER)

                                ;DRIVE'S HISTORY AND CURRENT INDICATOR STORAGE EQUATES

        000020                  $WRDL   =       $FMT+17         ;WORD COUNT (NOT 2'S COMP)
        000022                  $SSEC   =       $WRDL+2         ;SECTOR SIZE FOR CURRENT OPERATION
        000024                  $CODE   =       $WRDL+4         ;PRESENT COMMAND SELECTION CODE
        000026                  $PACK   =       $WRDL+6         ;WRITE DATA PACK INDICATOR
        000027                  $PREVO  =       $WRDL+7         ;PREVIOUS COMMAND SELECTION CODE
        000030                  $PATTC  =       $WRDL+10        ;PATTERN CODE
        000032                  $PREVA  =       $WRDL+12        ;PREVIOUS ADDRESS - TRACK, SECTOR, CYLINDER
        000036                  $OPERC  =       $WRDL+16        ;OPERATION COUNT
        000042                  $POSIT  =       $WRDL+22        ;SEEK COUNT
        000046                  $TRANS  =       $WRDL+26        ;TOTAL BITS XFERED COUNT (R & W)
        000052                  $READ   =       $WRDL+32        ;TOTAL BITS READ COUNT
        000056                  $TOTAL  =       $WRDL+36        ;TOTAL ERRORS (ALL TYPES) COUNT
        000060                  $SOFT   =       $WRDL+40        ;'SOFT' ERROR COUNT
        000062                  $HARD   =       $WRDL+42        ;'HARD' ERROR COUNT
        000064                  $SKI    =       $WRDL+44        ;'SKI' OR 'OCYL' ERROR COUNT
        000066                  $MISPO  =       $WRDL+46        ;PROG DETECTED MISPOSITIONING ERROR S COUNT
        000070                  $PASSC  =       $WRDL+50        ;PASS COUNTER
        000072                  $FAIR   =       $WRDL+52        ;OPERATION QUEUE 'FAIRNESS' COUNT

                                ;INDEX EQUATES TO THE NEXT OPERATION PARAMETERS

        000074                  $NCODE  =       $WRDL+54        ;NEXT OPERATION CODE
        000075                  $NPATC  =       $NCODE+1        ;NEXT PATTERN
        000076                  $NSEC   =       $NCODE+2        ;NEXT SECTOR
        000077                  $NTRK   =       $NCODE+3        ;NEXT TRACK
        000100                  $NCYL   =       $NCODE+4        ;NEXT CYLINDER
        000102                  $NWRDL  =       $NCODE+6        ;NEXT BUFFER SIZE
        000104                  $NEXT   =       $NCODE+10       ;PARAMETER SELECTION INDICATOR

                                ;INDEX EQUATES FOR MAXIMUM/MINIMUM ADDRESSES

        000106                  MAXCYL  =       $NCODE+12       ;MAXIMUM CYLINDER ADDRESS
        000110                  MINCYL  =       MAXCYL+2        ;MINIMUM CYLINDER ADDRESS
        000112                  MAXTRK  =       MAXCYL+4        ;MAXIMUM TRACK ADDRESS
        000114                  MINTRK  =       MAXCYL+6        ;MINIMUM TRACK ADDRESS
        000116                  MAXSEC  =       MAXCYL+10       ;MAXIMUM SECTOR ADDRESS
        000120                  MINSEC  =       MAXCYL+12       ;MINIMUM SECTOR ADDRESS
        000122                  $FIRST  =       MAXCYL+14       ;FIRST OPERATION INDICATOR

                                ;BAD SECTOR/TRACK ADDRESS STORAGE AREA INDEX EQUATE
```

```
                000124                   $BDSEC  =       MAXCYL+16       ;BAD SECTOR STORAGE TABLE

                                         ;DRIVE ID AREA INDEX EQUATE

                000224                   $DRVID  =       $BDSEC+100      ;DRIVE ID

                                         ;RH11/RP04/5/6 REGISTER EQUATES

                000234                   $RPCS1  =       $DRVID+10       ;RP04 REGISTER STORAGE
                000236                   $RPWC   =       $RPCS1+2
                000240                   $RPBA   =       $RPCS1+4
                000242                   $RPDA   =       $RPCS1+6
                000244                   $RPCS2  =       $RPCS1+10
                000246                   $RPDS1  =       $RPCS1+12
                000250                   $RPER1  =       $RPCS1+14
                000252                   $RPAS   =       $RPCS1+16
                000254                   $RPLA   =       $RPCS1+20
                000256                   $RPDB   =       $RPCS1+22
                000260                   $RPMR   =       $RPCS1+24
                000262                   $RPDT   =       $RPCS1+26
                000264                   $RPSN   =       $RPCS1+30
                000266                   $RPOF   =       $RPCS1+32
                000270                   $RPCA   =       $RPCS1+34
                000272                   $RPCC   =       $RPCS1+36
                000274                   $RPER2  =       $RPCS1+40
                000276                   $RPER3  =       $RPCS1+42
                000300                   $RPEC1  =       $RPCS1+44
                000302                   $RPEC2  =       $RPCS1+46


                                         ;BLOCK FOR DRIVE 0

        043110     000      000          DRIVE0: .BYTE   0,0             ;DRIVE NUMBER
        043112                                   .BLKW   5
        043124  043344                           .WORD   .+$RPCS1-$REG
        043126                                   .BLKB   $RPEC2-$REG

                                         ;BLOCK FOR DRIVE 1

        043414     001      000          DRIVE1: .BYTE   1,0             ;DRIVE NUMBER
        043416                                   .BLKW   5
        043430  043650                           .WORD   .+$RPCS1-$REG
        043432                                   .BLKB   $RPEC2-$REG

                                         ;BLOCK FOR DRIVE 2

        043720     002      000          DRIVE2: .BYTE   2,0             ;DRIVE NUMBER
        043722                                   .BLKW   5
        043734  044154                           .WORD   .+$RPCS1-$REG
        043736                                   .BLKB   $RPEC2-$REG

                                         ;BLOCK FOR DRIVE 3
```

```
      044224      003     000           DRIVE3: .BYTE   3,0             ;DRIVE NUMBER
      044226                                    .BLKW   5
      044240   044460                           .WORD   .+$RPCS1-$REG
      044242                                    .BLKB   $RPEC2-$REG

                                        ;BLOCK FOR DRIVE 4

      044530      004     000           DRIVE4: .BYTE   4,0             ;DRIVE NUMBER
      044532                                    .BLKW   5
      044544   044764                           .WORD   .+$RPCS1-$REG
      044546                                    .BLKB   $RPEC2-$REG

                                        ;BLOCK FOR DRIVE 5

      045034      005     000           DRIVE5: .BYTE   5,0             ;DRIVE NUMBER
      045036                                    .BLKW   5
      045050   045270                           .WORD   .+$RPCS1-$REG
      045052                                    .BLKB   $RPEC2-$REG

                                        ;BLOCK FOR DRIVE 6

      045340      006     000           DRIVE6: .BYTE   6,0             ;DRIVE NUMBER
      045342                                    .BLKW   5
      045354   045574                           .WORD   .+$RPCS1-$REG
      045356                                    .BLKB   $RPEC2-$REG

                                        ;BLOCK FOR DRIVE 7

      045644      007     000           DRIVE7: .BYTE   7,0             ;DRIVE NUMBER
      045646                                    .BLKW   5
      045660   046100                           .WORD   .+$RPCS1-$REG
      045662                                    .BLKB   $RPEC2-$REG

   98
   99                                   ;GENERAL PURPOSE DPB - USED BY 'READHD', 'RECALT', 'OFFSET',&'RTNCTR'
  100
  101 046150   000000  000000  177774   GENDPB: .WORD   0,0,-4,CYLDER
  102 046160   000000  000000  046170           .WORD   0,0,GENREG,0
  103 046170                            GENREG: .BLKW   24              ;REGISTER STORAGE IF ERROR
```

```
                                          .SBTTL  ERROR MESSAGES

 046240     122     110     061   EM1:    .ASCIZ  /RH11 INTERRUPT OCCURRED (RPAS = 0)/
 046303     125     116     105   EM2:    .ASCIZ  /UNEXPECTED ATTENTION OCCURRED/
 046341     115     101     123   EM3:    .ASCIZ  /MASSBUS PARITY ERROR (MCPE=1)/
 046377     115     101     123   EM4:    .ASCIZ  /MASSBUS PARITY ERROR (PAR=1)/
 046434     101     104     104   EM5:    .ASCIZ  /ADDRESS PLUG CHANGE BIT SET/
 046470     122     110     061   EM6:    .ASCIZ  /RH11 DIDN'T RESPOND TO ADDRESSING/
 046532     125     116     103   EM10:   .ASCIZ  /UNCORRECTABLE MASSBUS PARITY ERROR/
 046575     106     101     124   EM11:   .ASCIZ  /FATAL MASSBUS PARITY ERROR/
 046630     120     105     122   EM12:   .ASCIZ  /PERSISTENT DEVICE UNSAFE/
 046661     117     120     105   EM13:   .ASCIZ  /OPERATION NOT COMPLETED WITHIN TIME LIMIT/
 046733     104     122     111   EM14:   .ASCIZ  /DRIVE WENT OFFLINE/
 046756     116     117     040   EM15:   .ASCIZ  /NO RESPONSE TO PORT REQUEST/
 047012     110     105     101   EM20:   .ASCIZ  /HEADER CRC ERROR/
 047033     104     101     124   EM21:   .ASCIZ  /DATA CHECK ('DCK') ERROR/
 047064     127     122     111   EM22:   .ASCIZ  /WRITE CHECK ERROR - DATA CHECK ('DCK') SET/
 047137     127     122     111   EM23:   .ASCIZ  /WRITE CHECK ERROR - DATA CHECK ('DCK') NOT SET/
 047216     110     105     101   EM24:   .ASCIZ  /HEADER READ ERROR - 'FMT' BIT DROPPED/
 047264     110     105     101   EM25:   .ASCIZ  /HEADER READ ERROR - HEADER COMPARE ('HCE') ERROR/
 047345     106     117     122   EM26:   .ASCIZ  /FORMAT ERROR ('FER')/
 047372     110     105     101   EM27:   .ASCIZ  /HEADER COMPARE ('HCE') ERROR/
 047427     115     111     123   EM30:   .ASCIZ  /MISCELLANEOUS DRIVE ERROR/
 047461     117     120     105   EM31:   .ASCIZ  /OPERATION INCOMPLETE ('OPI') ERROR/
 047524     104     122     111   EM32:   .ASCIZ  /DRIVE TIMING ('DTE') ERROR/
 047557     120     101     122   EM33:   .ASCIZ  /PARITY ('PAR') ERROR AFTER OPERATION STARTED/
 047634     127     122     111   EM34:   .ASCIZ  /WRITE CLOCK FAILURE ('WCF') ERROR/
 047676     111     116     126   EM35:   .ASCIZ  /INVALID ADDRESS ('IAE') ERROR/
 047734     127     122     111   EM36:   .ASCIZ  /WRITE LOCK ('WLE') ERROR/
 047765     104     101     124   EM37:   .ASCIZ  /DATA CHECK ('DCK') SET DURING WRITE CHECK COMMAND/
 050047     122     110     061   EM40:   .ASCIZ  /RH11 OR UNIBUS TRANSFER ERROR/
 050105     102     125     123   EM41:   .ASCIZ  /BUS ADDRESS OR WORD COUNT INCORRECT/
 050151     104     101     124   EM42:   .ASCIZ  /DATA COMPARE ERRORS - NO OTHER ERROR(S) DETECTED/
 050232     103     101     116   EM43:   .ASCIZ  /CAN'T MATCH DATA READ WITH A PATTERN/
 050277     105     122     122   EM44:   .ASCIZ  /ERROR BIT(S) SET, BUT NO ERROR SIGNALED BY THE RH11/
 050363     105     103     103   EM45:   .ASCIZ  /ECC LOGIC FAILURE - POSITION REGISTER VALUE TOO LARGE/
 050451     102     125     123   EM46:   .ASCIZ  /BUS ADDRESS AND WORD COUNT NOT CONSISTENT/
 050523     123     105     105   EM50:   .ASCIZ  /SEEK INCOMPLETE ('SKI') OR OFF CYLINDER ('OCYL') ERROR/
 050612     120     122     117   EM51:   .ASCIZ  /PROGRAM DETECTED POSITIONING ERROR/
 050655     104     122     111   EM60:   .ASCIZ  /DRIVE UNSAFE ERROR/

 050700     122     120     101   DH1:    .ASCIZ  /RPAS/
 050705     104     122     111   DH2:    .ASCIZ  /DRIVE   RPDS1   RPER1   RPER2   RPER3   RPAS/
 050762     104     122     111   DH3:    .ASCIZ  /DRIVE   REG ADR  DATA/
 051010     104     122     111   DH4:    .ASCIZ  /DRIVE   REG ADR    GOOD    BAD/
 051047     044     122     120   DH6:    .ASCIZ  /$RPADR/
 051056     104     122     126   DH14:   .ASCII  /DRV RPCS1   RPCS2   RPDS1   RPER1    /
 051122     122     120     105           .ASCIZ  /RPER2   RPER3   RPEC1   RPEC2/<CR><LF>
 051162     122     120     127   DH15:   .ASCII  /RPWC    RPBA    RPDA    RPAS    RPLA    /
 051232     122     120     104           .ASCIZ  /RPDB    RPMR    RPDT/<CR><LF>
 051261     122     120     123   DH16:   .ASCIZ  /RPSN    RPOF    RPCA    RPCC    STATUS/<CR><LF>
                                          .EVEN

 051332  001244  000000           DT1:    .WORD   ATTN,0
 051336  001242  034334  034336   DT2:    .WORD   DRIVE,RPERRS,RPERRS+2,RPERRS+4,RPERRS+6,ATTN,0
 051354  001242  041652  041654   DT3:    .WORD   DRIVE,RD.ADR,RD.WRD,0
 051364  001242  042072  042070   DT4:    .WORD   DRIVE,WRT.AD,WRT.WD,RD.WRD,0
```

```
     58 051376  001170  000000          DT6:     .WORD   $RPADR,0
     59 051402  000234  000244  000246  DT14:    .WORD   $RPCS1,$RPCS2,$RPDS1,$RPER1,$RPER2,$RPER3,$RPEC1,$RPEC2,0
     60 051424  000236  000240  000242  DT15:    .WORD   $RPWC,$RPBA,$RPDA,$RPAS,$RPLA,$RPDB,$RPMR,$RPDT,0
     61 051446  000264  000266  000270  DT16:    .WORD   $RPSN,$RPOF,$RPCA,$RPCC,$STATUS,0
     62
     63 051462     120     122     105  LIN2C:   .ASCIZ  /PRESENT ORDER = /
     64 051503     040     040     120  LIN2P:   .ASCIZ  /  PREVIOUS ORDER = /
     65 051527     052     040     105  LIN2S:   .ASCIZ  @* ERROR AT BAD TRACK/SECTOR@
     66 051563     105     122     122  LINM3:   .ASCIZ  /ERROR AT C/
     67 051576     040     124     000  T:       .ASCIZ  / T/
     68 051601     120     122     105  LINN3:   .ASCIZ  /PRESENT ADDR = C/
     69 051622     040     123     000  S:       .ASCIZ  / S/
     70 051625     040     040     040  LINP3:   .ASCIZ  /    PREV ADDR = C/
     71 051646     123     124     101  LINS3:   .ASCIZ  /START CYL = /
     72 051663     040     040     105  LINEN3:  .ASCIZ  /  END CYL = /
     73 051700     040     040     101  LINA3:   .ASCIZ  /  ACTUAL CYL = /
     74 051720     040     040     124  LINT3:   .ASCIZ  /  TRK = /
     75 051731     040     122     120  LINCA3:  .ASCIZ  / RPCA = /
     76 051742     122     120     104  LINDA3:  .ASCIZ  /RPDA = /
     77 051752     122     120     102  LINB3:   .ASCIZ  /RPBA = /
     78 051762     040     040     122  LINW3:   .ASCIZ  /  RPWC = /
     79 051774     123     124     101  LINST3:  .ASCIZ  /START TRK = /
     80 052011     123     124     101  LINSS3:  .ASCIZ  /START SEC = /
     81 052026     102     125     106  LINM4:   .ASCIZ  /BUFFER ADDR = /
     82 052045     040     040     123  LINS4:   .ASCIZ  /  SIZE = /
     83 052057     040     040     101  LINX4:   .ASCIZ  /  ACTUAL NMBR WRDS XFRD = /
     84 052112     107     117     117  LIND5:   .ASCIZ  /GOOD DATA = /
     85 052127     040     040     102  LINB5:   .ASCIZ  /  BAD DATA = /
     86 052145     040     040     123  LINP5:   .ASCIZ  /  SECT POS = /
     87 052163     110     105     101  LINS5:   .ASCIZ  /HEADER FROM ERROR SECTOR = /
     88 052217     122     120     105  LINEP5:  .ASCIZ  /RPEC1 = /
     89 052230     040     122     120  LINEO5:  .ASCIZ  / RPEC2 = /
     90 052242     123     105     103  LINB6:   .ASCIZ  /SECTOR IS ECC CORRECTABLE /
     91 052275     123     105     103  LINC6:   .ASCIZ  /SECTOR READ CORRECTLY /
     92 052324     103     117     122  LING6:   .ASCIZ  /CORRECTED ON /
     93 052342     040     122     105  LINR6:   .ASCIZ  / RETRIES/
     94 052353     125     116     103  LINU06:  .ASCIZ  /UNCORRECTABLE AFTER /
     95 052400     040     040     124  LIN7M:   .ASCIZ  /  TOTAL MISPOS ERR = /
     96 052426     117     122     104  LIN7O:   .ASCIZ  /ORDERS:/
     97 052436     040     124     117  LIN7P:   .ASCIZ  / TOTAL SEEKS = /
     98 052456     040     124     117  LIN7S:   .ASCIZ  / TOTAL SKI,OCYL ERR = /
     99 052505     040     040     105  LIN7T:   .ASCIZ  /  ERRORS:/
    100 052517     040     040     127  LIN7X:   .ASCIZ  /  WRDS XFR:/
    101 052533     040     040     127  LIN7R:   .ASCIZ  /  WRDS READ:/
    102 052550     104     111     106  LIN8M:   .ASCIZ  /DIFFERENT ERROR DURING RETRY/
    103 052605     104     101     124  LIN9B:   .ASCIZ  /DATA COMPARISON ERRORS/
    104 052634     040     040     040  LIN9H:   .ASCII  /          GOOD     BAD/<CRLF>
    105 052661     114     117     103           .ASCIZ  /LOC       DATA     DATA/<CRLF>
    106 052710     114     117     103  LIN9I:   .ASCIZ  /LOC       DATA/<CRLF>
    107 052727     124     117     124  LIN9E:   .ASCIZ  /TOTAL COMPARE ERRORS = /
    108 052757     124     110     105  LIN9G:   .ASCIZ  /THE DATA COMPARED OK/<CRLF>
    109 053005     105     122     122  LIN10A:  .ASCIZ  /ERROR BURST BEGINS AT WORD /
    110 053041     040     111     116  LIN10B:  .ASCIZ  / IN DATA FIELD OF ERROR SECTOR/<CRLF>
    111 053101     105     122     122  LIN10C:  .ASCII  /ERROR WAS NOT IN THE DATA READ - /<CRLF>
    112 053143     105     103     103           .ASCIZ  /ECC CORRECTION CAN'T BE PERFORMED/
    113 053205     105     103     103  LIN10H:  .ASCII  /ECC CORRECTION RESULTS/<CRLF>
    114 053234     101     104     104           .ASCIZ  /ADDR    BAD      CORRECTED /<CRLF>
```

```
   115 053270     103     117     116  LIN11H: .ASCIZ  /CONTENTS OF ERROR SECTOR (REPORTED ABOVE)/<CRLF>
   116 053343     101     104     104          .ASCIZ  /ADDR    DATA/<CRLF>
   117 053361     040     040              LIN4SP: .ASCII  /  /
   118 053363     040                      LINSP:  .ASCII  / /
   119 053364     040     000              LINSPO: .ASCIZ  / /
```

```
                                        .SBTTL  TELETYPE MESSAGES

  053366    200     127     101         USE:    .ASCIZ  <CRLF>/WARNING: PROGRAMMABLE DRIVES MAY BE USED/<CRLF>
  053441    040     120     122         NOUSE:  .ASCIZ  / PROGRAMMABLE-DRIVE WILL NOT BE USED/
  053506    104     122     111         UNTMSG: .ASCIZ  /DRIVE/
  053514    040     117     106         UNTOFF: .ASCIZ  / OFFLINE/
  053525    040     117     116         UNTON:  .ASCIZ  / ONLINE/
  053535    040     116     117         UNTNOT: .ASCIZ  / NOT BEING TESTED/
  053557    040     101     114         UNTASN: .ASCIZ  / ALREADY BEING TESTED/
  053605    040     116     117         NOTRP:  .ASCIZ  @ NOT AN RP04/5/6@
  053626    040     116     117         NOTPRS: .ASCIZ  / NOT PRESENT/
  053643    040     116     117         NOTAVL: .ASCIZ  / NOT AVAILABLE/
  053662    040     125     116         NOTSAF: .ASCIZ  / UNSAFE/
  053672    125     116     111         SYSTAT: .ASCIZ  /UNIT STATUS:/<CRLF><LF>
  053711    122     120     060         RP04B:  .ASCIZ  /RP04/
  053716    122     120     060         RP05:   .ASCIZ  /RP05/
  053723    122     120     060         RP06:   .ASCIZ  /RP06/
  053730    104     122     111         STATHD: .ASCII  /DRIVE PERFORMANCE SUMMARY/<CRLF>
  053762    104     122     126                 .ASCII  /DRV PASS ORDERS    SEEKS   WRDS XFER   WRDS READ /
  054042    123     117     106                 .ASCIZ  /SOFT HARD  SKI MISP OTHER/<CRLF>
  054075    104     117     116         PDONE:  .ASCIZ  /DONE/<CRLF><LF>
  054104    007     077     106         DROPNG: .ASCIZ  <07>/?FATAL OR EXCESSIVE ERRORS/
  054140    105     116     104         ENDPAS: .ASCIZ  /END OF PASS/
  054154    200     105     116         ENDTST: .ASCIZ  <CRLF>/END OF TEST/
  054171    200     104     122         DEASSG: .ASCIZ  <CRLF>/DRIVE DEASSIGNED/
  054213    200     052     052         DRNUM:  .ASCIZ  <CRLF>/*********** DRIVE #/
  054237    040     123     124         ASGND:  .ASCIZ  / STARTED/<CRLF>
  054251    200     077     040         NEDCLK: .ASCIZ  <CRLF>/? 'L' OR 'P' CLOCK REQUIRED ON SYSTEM/<CRLF>
  054321    056     000                 PERIOD: .ASCIZ  /./
  054323    077     000                 QUES:   .ASCIZ  /?/
  054325    111     116     126         INVLD:  .ASCIZ  /INVALID COMMAND/<CRLF>
  054346    200     105     116         ENTCOM: .ASCIZ  <CRLF>/ENTER COMMANDS: /
  054370    105     116     124         ENTDRV: .ASCIZ  /ENTER I.D. FOR DRV #/
  054415    200     105     116         ENTLMT: .ASCIZ  <CRLF>/ENTER ADDRESS LIMITS FOR DRV #/
  054455    105     116     124         ENTADR: .ASCIZ  @ENTER BAD TRK/SEC ADRS FOR DRV #@
  054516    072     000                 COLON:  .ASCIZ  /:/
  054520    200     104     101         DATEIS: .ASCIZ  <CRLF>/DATE: /
  054530    200     117     120         IDIS:   .ASCIZ  <CRLF>/OPERATOR I.D.: /
  054551    200     012     104         HEDLIN: .ASCIZ  <CRLF><LF>/DRV  DRV I.D./<CRLF>
  054572    116     117     116         NONE:   .ASCIZ  /NONE/<CRLF>
  054600    077     040     111         BADENT: .ASCIZ  /? INVALID ENTRY/<CRLF>
  054621    123     131     123         BUSY:   .ASCIZ  /SYSTEM BUSY.../<CRLF>
  054641    200     120     122         INTDON: .ASCII  <CRLF>/PROGRAM INITIALIZATION COMPLETE/
  054701    200     124     131                 .ASCIZ  <CRLF>/TYPE A 'CONTROL C' TO ENTER COMMANDS/<CRLF><LF>

                                        .EVEN

                                        ;PARAMETER ENTRY TABLE

  054752  055100  000000  001404        PARLST: .WORD   PAR1,0,MAXDL
  054760  055106  177777  001410                .WORD   PAR2,-1,INTRVL
  054766  055236  177777  001402                .WORD   PAR19,-1,PASCNT
  054774  055115  177777  001414                .WORD   PAR3,-1,CMPLMT
  055002  055205  000001  001420                .WORD   PAR11,1,WCSEL
  055010  055213  000007  001422                .WORD   PAR14,7,RATIO
  055016  055230  000001  001430                .WORD   PAR16,1,ENDET
  055024  055176  000001  001416                .WORD   PAR10,1,FORMAT
```

```
055032  055221  000001  001424              .WORD   PAR15,1,AUTOCK
055040  055245  000001  001426              .WORD   PAR20,1,NOTPRT
055046  000000                              .WORD   0                       ;TABLE TERMINATOR

055050     105     116     124  ASKPAR:  .ASCIZ  /ENTER PARAMETERS:  /
055074     040     057     040  SLASH:   .ASCIZ  @ / @

055100     115     101     130  PAR1:    .ASCIZ  /MAXDL/
055106     111     116     124  PAR2:    .ASCIZ  /INTRVL/
055115     103     115     120  PAR3:    .ASCIZ  /CMPLMT/
055124     115     101     130  PAR4:    .ASCIZ  /MAXCYL/
055133     115     111     116  PAR5:    .ASCIZ  /MINCYL/
055142     115     101     130  PAR6:    .ASCIZ  /MAXTRK/
055151     115     111     116  PAR7:    .ASCIZ  /MINTRK/
055160     115     101     130  PAR8:    .ASCIZ  /MAXSEC/
055167     115     111     116  PAR9:    .ASCIZ  /MINSEC/
055176     106     117     122  PAR10:   .ASCIZ  /FORMAT/
055205     127     103     123  PAR11:   .ASCIZ  /WCSEL/
055213     122     101     124  PAR14:   .ASCIZ  /RATIO/
055221     101     125     124  PAR15:   .ASCIZ  /AUTOCK/
055230     105     116     104  PAR16:   .ASCIZ  /ENDET/
055236     120     101     123  PAR19:   .ASCIZ  /PASCNT/
055245     116     117     124  PAR20:   .ASCIZ  /NOTPRT/

                                         .EVEN

                                ;PARAMETER TABLE POINTERS FOR ADDRESS LIMITS

055254  055274                  TABLE:   .WORD   TABLE0          ;PARAMETER TABLE FOR DRIVE 0
055256  055342                           .WORD   TABLE1          ;PARAMETER TABLE FOR DRIVE 1
055260  055410                           .WORD   TABLE2          ;PARAMETER TABLE FOR DRIVE 2
055262  055456                           .WORD   TABLE3          ;PARAMETER TABLE FOR DRIVE 3
055264  055524                           .WORD   TABLE4          ;PARAMETER TABLE FOR DRIVE 4
055266  055572                           .WORD   TABLE5          ;PARAMETER TABLE FOR DRIVE 5
055270  055640                           .WORD   TABLE6          ;PARAMETER TABLE FOR DRIVE 6
055272  055706                           .WORD   TABLE7          ;PARAMETER TABLE FOR DRIVE 7

                                ;PARAMETER TABLE FOR ADDRESS LIMITS

055274  055133  000000  043220  TABLE0:  .WORD   PAR5,0,MINCYL+DRIVE0
055302  055124  000000  043216           .WORD   PAR4,0,MAXCYL+DRIVE0
055310  055151  000022  043224           .WORD   PAR7,18.,MINTRK+DRIVE0
055316  055142  000022  043222           .WORD   PAR6,18.,MAXTRK+DRIVE0
055324  055167  000025  043230           .WORD   PAR9,21.,MINSEC+DRIVE0
055332  055160  000025  043226           .WORD   PAR8,21.,MAXSEC+DRIVE0,0

055342  055133  000000  043524  TABLE1:  .WORD   PAR5,0,MINCYL+DRIVE1
055350  055124  000000  043522           .WORD   PAR4,0,MAXCYL+DRIVE1
055356  055151  000022  043530           .WORD   PAR7,18.,MINTRK+DRIVE1
055364  055142  000022  043526           .WORD   PAR6,18.,MAXTRK+DRIVE1
055372  055167  000025  043534           .WORD   PAR9,21.,MINSEC+DRIVE1
055400  055160  000025  043532           .WORD   PAR8,21.,MAXSEC+DRIVE1,0

055410  055133  000000  044030  TABLE2:  .WORD   PAR5,0,MINCYL+DRIVE2
055416  055124  000000  044026           .WORD   PAR4,0,MAXCYL+DRIVE2
055424  055151  000022  044034           .WORD   PAR7,18.,MINTRK+DRIVE2
055432  055142  000022  044032           .WORD   PAR6,18.,MAXTRK+DRIVE2
```

```
    055440  055167  000025  044040              .WORD   PAR9,21.,MINSEC+DRIVE2
    055446  055160  000025  044036              .WORD   PAR8,21.,MAXSEC+DRIVE2,0

    055456  055133  000000  044334  TABLE3:     .WORD   PAR5,0,MINCYL+DRIVE3
    055464  055124  000000  044332              .WORD   PAR4,0,MAXCYL+DRIVE3
    055472  055151  000022  044340              .WORD   PAR7,18.,MINTRK+DRIVE3
    055500  055142  000022  044336              .WORD   PAR6,18.,MAXTRK+DRIVE3
    055506  055167  000025  044344              .WORD   PAR9,21.,MINSEC+DRIVE3
    055514  055160  000025  044342              .WORD   PAR8,21.,MAXSEC+DRIVE3,0

    055524  055133  000000  044640  TABLE4:     .WORD   PAR5,0,MINCYL+DRIVE4
    055532  055124  000000  044636              .WORD   PAR4,0,MAXCYL+DRIVE4
    055540  055151  000022  044644              .WORD   PAR7,18.,MINTRK+DRIVE4
    055546  055142  000022  044642              .WORD   PAR6,18.,MAXTRK+DRIVE4
    055554  055167  000025  044650              .WORD   PAR9,21.,MINSEC+DRIVE4
    055562  055160  000025  044646              .WORD   PAR8,21.,MAXSEC+DRIVE4,0

    055572  055133  000000  045144  TABLE5:     .WORD   PAR5,0,MINCYL+DRIVE5
    055600  055124  000000  045142              .WORD   PAR4,0,MAXCYL+DRIVE5
    055606  055151  000022  045150              .WORD   PAR7,18.,MINTRK+DRIVE5
    055614  055142  000022  045146              .WORD   PAR6,18.,MAXTRK+DRIVE5
    055622  055167  000025  045154              .WORD   PAR9,21.,MINSEC+DRIVE5
    055630  055160  000025  045152              .WORD   PAR8,21.,MAXSEC+DRIVE5,0

    055640  055133  000000  045450  TABLE6:     .WORD   PAR5,0,MINCYL+DRIVE6
    055646  055124  000000  045446              .WORD   PAR4,0,MAXCYL+DRIVE6
    055654  055151  000022  045454              .WORD   PAR7,18.,MINTRK+DRIVE6
    055662  055142  000022  045452              .WORD   PAR6,18.,MAXTRK+DRIVE6
    055670  055167  000025  045460              .WORD   PAR9,21.,MINSEC+DRIVE6
    055676  055160  000025  045456              .WORD   PAR8,21.,MAXSEC+DRIVE6,0

    055706  055133  000000  045754  TABLE7:     .WORD   PAR5,0,MINCYL+DRIVE7
    055714  055124  000000  045752              .WORD   PAR4,0,MAXCYL+DRIVE7
    055722  055151  000022  045760              .WORD   PAR7,18.,MINTRK+DRIVE7
    055730  055142  000022  045756              .WORD   PAR6,18.,MAXTRK+DRIVE7
    055736  055167  000025  045764              .WORD   PAR9,21.,MINSEC+DRIVE7
    055744  055160  000025  045762              .WORD   PAR8,21.,MAXSEC+DRIVE7,0

    055754  000000  000000  000000  CYLDER:     .WORD   0,0,0,0         ;HEADER BUFFER FOR 'READHD' ROUTINE

            055764                  ENDPGM  =           .                       ;LAST LOCATION OF PROG + 2

                                    ;ROUTINE TO GET THE DATE AND THE OPERATOR FROM THE OPERATOR
                                    ;CALL:
                                    ;           JSR     PC,OPRDAT
                                    ;           RETURN
                                    ;
                                    ;NOTE: THIS ROUTINE IS ENTERED ONLY AT INITIAL START

    055764  104401  056076          OPRDAT:     TYPE    ,ENTDAT         ;'ENTER DATE'
    055770  104411                              RDLIN                   ;READ THE ENTRY
    055772  012605                              MOV     (SP)+,R5        ;PUT THE ENTRY ADDRESS INTO R5
    055774  112537  001220                      MOVB    (R5)+,DATE      ;STORE THE DATE
    056000  112537  001221                      MOVB    (R5)+,DATE+1    ;STORE THE DATE
    056004  112537  001222                      MOVB    (R5)+,DATE+2    ;STORE THE DATE
    056010  112537  001223                      MOVB    (R5)+,DATE+3    ;STORE THE DATE
    056014  112537  001224                    MOVB    (R5)+,DATE+4      ;STORE THE DATE
    056020  112537  001225                    MOVB    (R5)+,DATE+5      ;STORE THE DATE
    056024  112537  001226                    MOVB    (R5)+,DATE+6      ;STORE THE DATE
    056030  112537  001227                    MOVB    (R5)+,DATE+7      ;STORE THE DATE
    056034  104401  056114                    TYPE    ,ENTID            ;'ENTER OPERATOR I.D.'
    056040  104411                            RDLIN                     ;READ THE ENTRY
    056042  012605                            MOV     (SP)+,R5          ;ENTRY ADDRESS
    056044  112537  001232                    MOVB    (R5)+,OPERID      ;STORE THE I.D.
    056050  112537  001233                    MOVB    (R5)+,OPERID+1    ;STORE THE I.D.
    056054  112537  001234                    MOVB    (R5)+,OPERID+2    ;STORE THE I.D.
    056060  112537  001235                    MOVB    (R5)+,OPERID+3    ;STORE THE I.D.
    056064  112537  001236                    MOVB    (R5)+,OPERID+4    ;STORE THE I.D.
    056070  112537  001237                    MOVB    (R5)+,OPERID+5    ;STORE THE I.D.
    056074  000207                            RTS     PC                ;RETURN

    056076     200     105     116  ENTDAT:   .ASCIZ  <CRLF>/ENTER DATE: /
    056114     105     116     124  ENTID:    .ASCIZ  /ENTER OPERATOR I.D.: /
                                              .EVEN
```

```
                                        .SBTTL  ROUTINE TO SIZE MEMORY

                                        ;;*******************************************************************
                                        ;*CALL:
                                        ;*       JSR     PC,$SIZE
                                        ;*       RETURN
                                        ;*$LSTAD WILL CONTAIN THE LAST AVAILABLE MEMORY LOCATION

  056142  010046                        $SIZE:  MOV     R0,-(SP)            ;;SAVE R0 ON THE STACK
  056144  010146                                MOV     R1,-(SP)            ;;SAVE R1 ON THE STACK
  056146  013746  000114                        MOV     @#114,-(SP)         ;;SAVE MEMORY ERROR VECTOR PS & PC
  056152  013746  000116                        MOV     @#116,-(SP)
  056156  012737  000116  000114                MOV     #116,@#114          ;;IGNORE PARITY ERRORS WHILE SIZING
  056164  012737  000002  000116                MOV     #RTI,@#116
  056172  013746  000004                        MOV     @#ERRVEC,-(SP)      ;;SAVE PRESENT ERROR VECTOR PS & PC
  056176  013746  000006                        MOV     @#ERRVEC+2,-(SP)
  056202  010600                                MOV     SP,R0               ;;SAVE THE STACK POINTER
                                        ;;SET THE ERRVEC PS TO THE PRESENT PS
  056204  104400                                TRAP                        ;;PUSH OLD PSW AND PC ON STACK
  056206  012637  000006                        MOV     (SP)+,@#ERRVEC+2          ;;SAVE THE PSW IN @#ERRVEC+2
  056212  012737  056232  000004                MOV     #2$,@#ERRVEC        ;;SET FOR TIMEOUT
  056220  012701  020000                        MOV     #20000,R1           ;;FIRST ADDRESS
  056224  005711                        1$:     TST     (R1)                ;;TEST THIS ADDRESS
  056226  005721                                TST     (R1)+               ;;STEP TO NEXT ADDRESS
  056230  000775                                BR      1$                  ;;TRY ANOTHER
  056232  162701  000002                2$:     SUB     #2,R1               ;;DROP BACK
  056236  010006                                MOV     R0,SP               ;;RESTORE THE STACK
  056240  012637  000006                        MOV     (SP)+,@#ERRVEC+2    ;;RESTORE ERROR VECTOR
  056244  012637  000004                        MOV     (SP)+,@#ERRVEC
  056250  012637  000116                        MOV     (SP)+,@#116         ;;RESTORE MEMORY ERROR VECTOR
  056254  012637  000114                        MOV     (SP)+,@#114
  056260  010137  056272                        MOV     R1,$LSTAD           ;;LAST ADDRESS
  056264  012601                                MOV     (SP)+,R1            ;;RESTORE R1
  056266  012600                                MOV     (SP)+,R0            ;;RESTORE R0
  056270  000207                                RTS     PC
  056272  000000                        $LSTAD: .WORD   0                   ;;CONTAINS THE LAST ADDRESS
```

```
                                        .SBTTL  BUSADR - GET BUS ADDRESS AND VECTOR ADDRESS FOR RH11
                                        ;THIS ROUTINE IS USED TO INSURE THE BUS ADDRESS
                                        ;OF THE RH11 IS SETUP FOR THE PROPER ADDRESS.
                                        ;IT WILL ALSO READ THE ADDRESS FROM THE TTY IF
                                        ;REQUIRED.
                                        ;NOTE: THIS ROUTINE DESTROYS R0-R4
                                        ;CALL
                                        ;
                                        ;       JSR     PC,BUSADR
                                        ;       RETURN

056274  177700                          HIAD:   .WORD 177700
056276  000776                          HIVEC:  .WORD 776
056300  000000                          BOUND:  .WORD 0

056302  005737  001260                  BUSADR: TST     CHGADR          ;INPUT FROM TTY REQUESTED?
056306  001456                                  BEQ     7$              ;NO--BRANCH
056310  005037  001260                          CLR     CHGADR          ;YES--CLEAR THE REQUEST FLAG
056314  012700  001170                  1$:     MOV     #$RPADR,R0      ;FIRST ADDRESS
056320  104401  056526                          TYPE    ,MRPCS1         ;'RPCS1="
056324  012046                                  MOV     (R0)+,-(SP)     ;PRESENT RPCS1 ADDRESS
056326  104402                                  TYPOC                   ;TYPE IT
056330  104401  053363                          TYPE    ,LINSP          ;2 SPACES
056334  104411                                  RDLIN                   ;GET THE ENTRY
056336  012601                                  MOV     (SP)+,R1        ;ADDRESS OF ASCII TEXT
056340  013737  056274  056300                  MOV     HIAD,BOUND      ;SET THE ADDRESS MAX
056346  004537  056550                          JSR     R5,CK.NUM       ;CHECK THE NUMBER
056352  056372                                  3$                      ;CARRIAGE RETURN ONLY ENTERED
056354  056444                                  7$                      ;PERIOD ONLY ENTERED
056356  056314                                  1$                      ;ILLEGAL INPUT
056360  056366                                  2$                      ;TERMINATED WITH A CARRIAGE RETURN
056362  056314                                  1$                      ;TERMINATED WITH A ","
056364  056440                                  4$                      ;TERMINATED WITH A ";"
056366  010260  177776                  2$:     MOV     R2,-2(R0)       ;SAVE NEW RPCS1
056372  104401  056537                  3$:     TYPE    ,MRHVEC         ;'RHVEC="
056376  012046                                  MOV     (R0)+,-(SP)     ;PRESENT RH11 VECTOR ADDRESS ON THE STACK
056400  104402                                  TYPOC                   ;TYPE IT
056402  104401  053363                          TYPE    ,LINSP          ;2 SPACES
056406  104411                                  RDLIN                   ;READ THE ENTRY
056410  012601                                  MOV     (SP)+,R1        ;ASCII TEXT ADDRESS
056412  013737  056276  056300                  MOV     HIVEC,BOUND     ;SET THE VECTOR-MAX
056420  004537  056550                          JSR     R5,CK.NUM       ;CHECK THE NUMBER
056424  056444                                  7$                      ;CARRIAGE RETURN ONLY ENTERED
056426  056444                                  7$                      ;PERIOD ONLY ENTERED
056430  056372                                  3$                      ;ILLEGAL INPUT
056432  056440                                  4$                      ;TERMINATED WITH A CARRIAGE RETURN
056434  056372                                  3$                      ;TERMINATED WITH A ","
056436  056440                                  4$                      ;TERMINATED WITH A ";"
056440  010260  177776                  4$:     MOV     R2,-2(R0)       ;SAVE INPUT
056444  013701  000004                  7$:     MOV     ERRVEC,R1       ;SAVE THE ERROR VECTOR
056450  012737  056504  000004                  MOV     #8$,ERRVEC      ;SETUP FOR TRAP
056456  005777  122506                          TST     @$RPADR         ;CHECK FOR RH11
056462  010137  000004                          MOV     R1,ERRVEC       ;RESTORE ERROR VECTOR
056466  012700  001170                          MOV     #$RPADR,R0      ;FIRST ADDRESS OF NEW PARAMETERS
056472  012701  034502                          MOV     #RPADR,R1       ;FIRST ADDRESS OF WHERE TO PUT THEM
056476  012021                                  MOV     (R0)+,(R1)+     ;BUS ADDRESS
056500  012021                                  MOV     (R0)+,(R1)+     ;VECTOR ADDRESS
```

```
056502  000207                              RTS     PC              ;RETURN
056504  010137  000004          8$:         MOV     R1,ERRVEC       ;RESTORE ERROR VECTOR
056510  022626                              CMP     (SP)+,(SP)+     ;CLEAN OFF THE STACK
056512  104006                              EMT     6
056514  005737  000042                      TST     @#42            ;IS THERE A MONITOR?
056520  001675                              BEQ     1$              ;NO--GO ASK FOR ADDRESS
056522  000137  005646                      JMP     $GET42          ;GO TO END OF PROGRAM

056526     122     120     103  MRPCS1:     .ASCIZ  @RPCS1 = @
056537     122     110     126  MRHVEC:     .ASCIZ  @RHVEC = @
```

```
                                .SBTTL  CK.NUM - CHECK NUMBER (OCTAL)
                                ;THIS ROUTINE CHECKS AN ASCIZ STRING FOR LEGAL CHARACTERS
                                ;AND FORMS AN OCTAL NUMBER IN R2
                                ;CALL:
                                ;       MOV     #ADR,R1         ;ADDRESS OF ASCIZ STRING
                                ;       MOV     #NUM,R2         ;MAX SIZE OF INPUT NUMBER
                                ;       JSR     R5,CK.NUM       ;GO FORM THE NUMBER
                                ;       RETURN  ADR1            ;"CR" ONLY ENTERED -- R2 = 0
                                ;       RETURN  ADR2            ;"PERIOD" ONLY ENTERED -- R2 = 0
                                ;       RETURN  ADR3            ;ILLEGAL CHARACTER IN THE INPUT STRING
                                ;       RETURN  ADR4            ;"CR" ENTERED -- R2 = NUMBER
                                ;       RETURN  ADR5            ;"COMMA" -- R2 = NUMBER
                                ;       RETURN  ADR6            ;"PERIOD" -- R2 = NUMBER

056550  010446                  CK.NUM: MOV     R4,-(SP)        ;SAVE R4
056552  010346                          MOV     R3,-(SP)        ;SAVE R3
056554  010246                          MOV     R2,-(SP)        ;SAVE R2
056556  005004                          CLR     R4              ;RETURN POINTER
056560  005003                          CLR     R3              ;START NUMBER AT ZERO
056562  005002                          CLR     R2              ;STORE RESULT
056564  004537  030072                  JSR     R5,CK.CHR       ;CHECK ONE CHARACTER
056570  056670                          6$                      ;ILLEGAL CHARACTER
056572  056674                          8$                      ;CARRIAGE RETURN
056574  056670                          6$                      ;","
056576  056672                          7$                      ;"."
056600  056604                          1$                      ;DIGIT 0-7
056602  056670                          6$                      ;DIGIT 8-9
056604  062705  000004          1$:     ADD     #4,R5           ;INCREMENT RETURN PAST "CR" AND "PERIOD" ONLY RETURNS
056610  006303                  2$:     ASL     R3              ;FOR THE OCTAL NUMBER IN R3
056612  103426                          BCS     6$              ;DON'T LET IT GET TO BIG
056614  006303                          ASL     R3
056616  103424                          BCS     6$
056620  006303                          ASL     R3
056622  103422                          BCS     6$
056624  060203                          ADD     R2,R3
056626  004537  030072                  JSR     R5,CK.CHR       ;CHECK ONE CHARACTER
056632  056674                          8$                      ;ILLEGAL CHARACTER
056634  056656                          5$                      ;CARRIAGE RETURN
056636  056654                          4$                      ;","
056640  056646                          3$                      ;"."
056642  056610                          2$                      ;DIGIT 0-7
056644  056674                          8$                      ;DIGIT 8-9
056646  105711                  3$:     TSTB    (R1)            ;DOES A "CR" FOLLOW THE "PERIOD"
056650  001011                          BNE     8$              ;BR IF NOT
056652  005724                          TST     (R4)+           ;INCREMENT THE RETURN
056654  005724                  4$:     TST     (R4)+           ;INCREMENT THE RETURN INDEX
056656  005724                  5$:     TST     (R4)+           ;INCREMENT THE RETURN INDEX
056660  023703  056300                  CMP     BOUND,R3        ;INPUT VALUE TOO LARGE?
056664  101003                          BHI     8$              ;BR IF IT IS
056666  000401                          BR      7$              ;BR IF NOT
056670  005725                  6$:     TST     (R5)+           ;INCREMENT THE RETURN ADDRESS
056672  005725                  7$:     TST     (R5)+           ;INCREMENT THE RETURN ADDRESS
056674  060405                  8$:     ADD     R4,R5           ;SETUP FOR PROPER RETURN
056676  010302                          MOV     R3,R2           ;LOAD ENTERED VALUE
056700  005726                          TST     (SP)+           ;CLEAN OFF THE STACK
056702  012603                          MOV     (SP)+,R3        ;RESTORE R3
056704  012604                          MOV     (SP)+,R4        ;RESTORE R4
```

```
056706  011505                            MOV     (R5),R5            ;GET RETURN ADDRESS
056710  000205                            RTS     R5                 ;RETURN

056712     200      124      117  LOADRV: .ASCII  <CRLF>/TO TEST DRIVE 0, REPLACE THE 'XXDP' PACK ON DRIVE 0/<CRLF>
056777     127      111      124          .ASCII  /WITH ANOTHER PACK, CLEAR MEMORY LOCATION 40,/<CRLF>
057054     101      116      104          .ASCIZ  /AND RESTART THE PROGRAM/<CRLF>
057105     200      123      131  NOLOAD: .ASCIZ  <CRLF>/SYSTEM HAS 16K MEMORY, 'XXDP' LOADER WILL BE OVERWRITTEN/<CRLF>

        000200                    .END 200
```

SYMBOL TABLE

```
ABNRML  027036      BIT12 = 010000      CMSEC   001316      DROP    026760      EM12    046630
ABS   = 000200      BIT13 = 020000      CMSTR   013566      DROPNG  054104      EM13    046661
ACK   = 000123      BIT14 = 040000      CMTRK   001317      DRQ   = 004000      EM14    046733
ACL   = 000040      BIT15 = 100000      COLON   054516      DRVACT  034344      EM15    046756
ACTDRV  034420      BIT2  = 000004      COMTBL  001760      DRVCLR= 000111      EM2     046303
ACTSTR  034421      BIT3  = 000010      CPSAVE  032244      DRVER   011342      EM20    047012
ACU   = 100000      BIT4  = 000020      CR    = 000015      DRVINT  034732      EM21    047033
AOE   = 001000      BIT5  = 000040      CRLF  = 000200      DRVPRM  025672      EM22    047064
ASGND   054237      BIT6  = 000100      CSF   = 000002      DRVQUE  042750      EM23    047137
ASGN1   024516      BIT7  = 000200      CSU   = 000010      DRVSTA  034354      EM24    047216
ASGN2   024576      BIT8  = 000400      CTRAP   031646      DRVTYP  034364      EM25    047264
ASGN3   024650      BIT9  = 001000      CYLDER  055754      DRY   = 000200      EM26    047345
ASGN4   024766      BLKADR  001740      CYLIMT  001350      DSWR  = 177570      EM27    047372
ASGN5   024776      BOUND   056300      DATAPK  025442      DTE   = 010000      EM3     046341
ASGN6   025004      BPTVEC= 000014      DATA0   003026      DTEER   012200      EM30    047427
ASGN7   025014      BUFTBL  001616      DATA1   003066      DTSY  = 000200      EM31    047461
ASKPAR  055050      BUSADR  056302      DATE    001220      DTUW    034466      EM32    047524
ASNERR  026730      BUSY    054621      DATEIS  054520      DT00  = 000001      EM33    047557
ASNLST  001462      CFLAG   001262      DCK   = 100000      DT01  = 000002      EM34    047634
ASNMSG  026750      CHGADR  001260      DCKER   010072      DT02  = 000004      EM35    047676
ASSIGN  024510      CI1     036112      DCKER1  010224      DT03  = 000010      EM36    047734
ATA   = 100000      CI3     036220      DCL   = 000100      DT04  = 000020      EM37    047765
ATABIT  034470      CI4     036326      DCU   = 000001      DT05  = 000040      EM4     046377
ATTN    001244      CI5     036704      DDISP = 177570      DT06  = 000100      EM40    050047
AT0   = 000001      CI6     036726      DEASGN  025064      DT07  = 000200      EM41    050105
AT1   = 000002      CI7     036742      DEASSG  054171      DT08  = 000400      EM42    050151
AT2   = 000004      CI7B    036770      DE1   = 000040      DT1     051332      EM43    050232
AT3   = 000010      CI8     037042      DFF20 = 000002      DT14    051402      EM44    050277
AT4   = 000020      CKBUS   013372      DH1     050700      DT15    051424      EM45    050363
AT5   = 000040      CKCLK   022714      DH14    051056      DT16    051446      EM46    050451
AT6   = 000100      CKCLK1  023010      DH15    051162      DT2     051336      EM5     046434
AT7   = 000200      CKCLK2  023056      DH16    051261      DT3     051354      EM50    050523
AUTOCK  001424      CKCLK3  023106      DH2     050705      DT4     051364      EM51    050612
AVAIL   001530      CKERR   013272      DH3     050762      DT6     051376      EM6     046470
A16   = 000400      CKFMT   011374      DH4     051010      DUNIT   001464      EM60    050655
A17   = 001000      CKHCE   011570      DH6     051047      DVA   = 004000      ENDCMP  014460
BADENT  054600      CKSWR = 104407      DIGB  = 000004      ECBAD0  001334      ENDCN   001352
BADSEC  001264      CK.CHR  030072      DISPLA  001142      ECBAD1  001342      ENDCON  001372
BADTMO  004106      CK.DEC  030044      DISPLY= 104414      ECBIT   001320      ENDET   001430
BAI   = 000010      CK.DIG  030144      DISPRE  000174      ECC     014604      ENDPAS  054140
BEGCOD  001434      CK.NUM  056550      DLT   = 100000      ECCX    015336      ENDPGM= 055764
BEGPAT  001432      CK.OCT  030016      DL64  = 000020      ECC1    015202      ENDSEK  001376
BEGSIZ  001436      CLKFLG  001210      DMD   = 000001      ECC2    015332      ENDSK   001356
BIT0  = 000001      CLOCK   024050      DONE    007566      ECGD    001332      ENDTST  054154
BIT00 = 000001      CLR   = 000040      DPINT   034374      ECGD1   001340      ENTADR  054455
BIT01 = 000002      CLRDPB  025466      DPR   = 000400      ECH   = 000100      ENTCOM  054346
BIT02 = 000004      CLRQUE  042652      DPRQS   034404      ECI   = 004000      ENTDAT  056076
BIT03 = 000010      CMCNT   001312      DRIVE   001242      ECMSK0  001324      ENTDRV  054370
BIT04 = 000020      CMCYL   001314      DRIVE0  043110      ECMSK1  001326      ENTID   056114
BIT05 = 000040      CMDAT   013740      DRIVE1  043414      ECSEC   001322      ENTLMT  054415
BIT06 = 000100      CMHED   013644      DRIVE2  043720      ECWRD   001330      ENTPR   005256
BIT07 = 000200      CMPAR   013456      DRIVE3  044224      ECWRD1  001336      EOP     027064
BIT08 = 000400      CMPARD  013474      DRIVE4  044530      EMPTYQ  042730      EOPX    027336
BIT09 = 001000      CMPLMT  001414      DRIVE5  045034      EMTVEC= 000030      EOP1    027116
BIT1  = 000002      CMPRES  020032      DRIVE6  045340      EM1     046240      EOP2    027140
BIT10 = 002000      CMPRT   014204      DRIVE7  045644      EM10    046532      ERCTR   001306
BIT11 = 004000      CMPRX   014176      DRNUM   054213      EM11    046575      ERPRC1  007146
```

```
ERPROC  007132        IE    = 000100        LINKDV  027366        MAXDL   001404        OFMSG0  002274
ERR   = 040000        ILF   = 000001        LINM3   051563        MAXER   001406        OFMSG1  002327
ERROR = 104000        ILR   = 000002        LINM4   052026        MAXSEC= 000116        OFMSG2  002363
ERRVEC= 000004        INCHRD  023632        LINN3   051601        MAXTRK= 000112        OFMSG3  002417
EXT1  = 000001        INCMIS  023702        LINOCT  022544        MCLK  = 000002        OFMSG4  002453
EXT10 = 000010        INCSKI  023656        LINP3   051625        MCPE  = 020000        OFMSG5  002507
EXT2  = 000002        INCSOF  023606        LINP5   052145        MCPEMX  034500        OFMSG6  002543
EXT20 = 000020        INCTOT  023726        LINR6   052342        MHS   = 001000        OFMSG7  002577
EXT4  = 000004        INTDON  054641        LINSP   053363        MINCYL= 000110        OFMSG8  002633
EXT40 = 000040        INTRVL  001410        LINSP0  053364        MINSEC= 000120        OFMSG9  002667
FACTOR  016172        INVLD   054325        LINSS3  052011        MINTRK= 000114        OFMTBL  002240
FAIRNS  001254        IOTVEC= 000020        LINST3  051774        MINUTE  001270        OFREV = 000200
FALPAR  007332        IR    = 000100        LINS3   051646        MINX  = 000004        OF100 = 000004
FALPR1  007342        ISR     037434        LINS4   052045        MNDLTA  034514        OF200 = 000010
FEN   = 000200        IXE   = 004000        LINS5   052163        MNTBL   002010        OF25  = 000001
FER   = 000020        KSR     024212        LINT3   051720        MOH   = 020000        OF400 = 000020
FILBUF  016544        KSR1    024246        LINU06  052353        MOL   = 010000        OF50  = 000002
FMTER   012374        LA      037276        LINW3   051762        MONTR   005570        OF800 = 000040
FMT22 = 010000        LACNT   034432        LINX4   052057        MPE   = 000400        OPE   = 020000
FORMAT  001416        LF    = 000012        LIN10A  053005        MRD   = 000020        OPERID  001232
FRSTER  001300        LIMIT   001310        LIN10B  053041        MRHVEC  056537        OPI   = 020000
F1    = 000002        LINA3   051700        LIN10C  053101        MRPCS1  056526        OPIER   012070
F2    = 000004        LINB3   051752        LIN10H  053205        MSE   = 000020        OPIER1  012134
F3    = 000010        LINB5   052127        LIN11H  053270        MSTCK = 000010        OPRDAT  055764
F4    = 000020        LINB6   052242        LIN2C   051462        MWR   = 000040        OPT     035632
F5    = 000040        LINCA3  051731        LIN2P   051503        MXDLTA  034512        OPTBL   001766
GENDPB  046150        LINC6   052275        LIN2S   051527        MXF   = 001000        OR    = 000200
GENREG  046170        LINDA3  051742        LIN3.1  021420        MXLACT  034510        ORDERQ  001440
GETADR  026264        LINDEC  022576        LIN3.3  021532        MXWNDW  034516        PACK    001216
GETBUF  016174        LIND5   052112        LIN3.4  021564        M.DPID  027434        PAR   = 000010
GETID   026154        LINEN3  051663        LIN4SP  053361        M.DP40  027472        PARENT  026604
GETPAR  017624        LINEO5  052230        LIN6.1  022216        M.DP41  027526        PARER   012222
GETPAT  017576        LINEP5  052217        LIN6.2  022240        M.DP42  027536        PARLST  054752
GETREG= 000141        LINE1   020502        LIN7M   052400        M.DP44  027570        PARQ    001574
GETREM  027340        LINE2   020546        LIN7O   052426        M.DP50  027602        PAR1    055100
GETREQ  043024        LINE2A  020716        LIN7P   052436        NBA   = 100000        PAR10   055176
GO    = 000001        LINE2B  020734        LIN7R   052533        NED   = 010000        PAR11   055205
GODRIV  016672        LINE3   021154        LIN7S   052456        NEDCLK  054251        PAR14   055213
GRV   = 000010        LINE3A  021162        LIN7T   052505        NEM   = 004000        PAR15   055221
GTSWR = 104406        LINE3B  021170        LIN7X   052517        NEWASN  025054        PAR16   055230
HCE   = 000200        LINE3C  021202        LIN8M   052550        NEWUNT  001506        PAR19   055236
HCEER   012452        LINE3D  021212        LIN9B   052605        NHS   = 002000        PAR2    055106
HCI   = 002000        LINE3E  021260        LIN9E   052727        NOLOAD  057105        PAR20   055245
HCRC  = 000400        LINE3F  021346        LIN9G   052757        NOMTCH  013050        PAR3    055115
HCRCER  011212        LINE4   021630        LIN9H   052634        NONE    054572        PAR4    055124
HEDLIN  054551        LINE5   021720        LIN9I   052710        NOTAVL  053643        PAR5    055133
HIAD    056274        LINE5A  022020        LKPAR   005232        NOTPRS  053626        PAR6    055142
HIVEC   056276        LINE5B  022112        LOADRV  056712        NOTPRT  001426        PAR7    055151
HOUR    001266        LINE6   022154        LST   = 002000        NOTRP   053605        PAR8    055160
HT    = 000011        LINE6A  022166        LSTAD   001256        NOTSAF  053662        PAR9    055167
HZ      001212        LINE6B  022174        MAIN    005676        NOUSE   053441        PASCNT  001402
IAE   = 002000        LINE6C  022202        MAIN1   006024        OCYL  = 100000        PAT   = 000020
IAEER   012314        LINE6D  022210        MAIN2   006144        OFFCOD  002220        PCLOCK  001206
IBSAVE  032246        LINE7   022264        MAIN3   006400        OFFSET= 000115        PDONE   054075
IDIS    054530        LINE7A  022412        MASK    001250        OFFST   015574        PERIOD  054321
IDLE    006504        LINE8   022532        MATCH   014526        OFLIN   007440        PFECH   032426
IDLE1   006602        LING6   052324        MAXCYL= 000106        OFMSGA  002724        PFECH1  032436
```

```
PFECH2  032520          RDHD  = 000173          RTNCTR  015516          ST02    041420          TRFER   012534
PFECH3  032552          RDLIN = 104411          R6    =%000006          ST03    041470          TRKLMT  001346
PFECH4  032562          RDY   = 000200          R7    =%000007          ST05    041514          TRK1  = 004000
PFTSTN  032566          RD.ADR  041652          S       051622          ST06    041522          TRK10 = 040000
PGE   = 002000          RD.RP   041626          SAVEFG  034442          ST07    041560          TRK2  = 010000
PGM   = 001000          RD.RP1  041650          SAVER1  001302          ST08    041610          TRK20 = 100000
PIP   = 020000          RD.RP2  041770          SAVER5  001304          ST09    041620          TRK4  = 020000
PIRQ  = 177772          RD.RP3  041774          SAVREG= 104412          SVRH11  042170          TRNSWT  034414
PIRQVE= 000240          RD.RP4  042000          SC      037762          SWR     001140          TRTVEC= 000014
PLU   = 020000          RD.WRD  041654          SCMND   025172          SWREG   000176          TSTPGM  035336
POPQUE  043046          READDR  022616          SCOPE = 000004          SWTIM   007402          TUF   = 000100
POSER   012014          READHD  015620          SC1   = 000100          SW0   = 000001          TYPDS = 104405
PROCES  007042          READIN= 000121          SC10  = 001000          SW00  = 000001          TYPE  = 104401
PRTBAD  015344          RECAL = 000107          SC11    040600          SW01  = 000002          TYPEST  023200
PRTIM   007530          RECALT  015542          SC12    040670          SW02  = 000004          TYPOC = 104402
PR0   = 000000          RECALO  015546          SC13    040740          SW03  = 000010          TYPON = 104404
PR1   = 000040          REDAPK  025454          SC2   = 000200          SW04  = 000020          TYPOS = 104403
PR2   = 000100          REFMT   006676          SC20  = 002000          SW05  = 000040          TYPRI4  027742
PR3   = 000140          REFMTX  007040          SC3     040026          SW06  = 000100          UCPAR   007320
PR4   = 000200          RELBUF  016330          SC4     040032          SW07  = 000200          ULDFLG  034422
PR5   = 000240          RELSE = 000113          SC5     040044          SW08  = 000400          UNIT    001246
PR6   = 000300          REPLZ   027612          SC6     040234          SW09  = 001000          UNLOAD= 000103
PR7   = 000340          RESREG= 104413          SC6A    040344          SW1   = 000002          UNS   = 040000
PS    = 177776          RESVEC= 000010          SC7     040472          SW10  = 002000          UNSAF   012770
PSEL  = 002000          RETRY   001252          SC8     040550          SW11  = 004000          UNTASN  053557
PSU   = 000001          RMR   = 000004          SDETAL  023250          SW12  = 010000          UNTMSG  053506
PSW   = 177776          RNOP  = 000101          SEARCH= 000131          SW13  = 020000          UNTNOT  053535
PUNSAF  007230          RPADR   034502          SECLMT  001344          SW14  = 040000          UNTOFF  053514
PWRVEC= 000024          RPAS  = 000016          SECOND  001272          SW15  = 100000          UNTON   053525
QCNT    042360          RPBA  = 000004          SEEK  = 000105          SW2   = 000004          UPE   = 020000
QDRV0   042452          RPCA  = 000034          SEEKFG  034444          SW3   = 000010          USE     053366
QDRV1   042472          RPCC  = 000036          SELDRV= 000145          SW4   = 000020          US1   = 000001
QDRV2   042512          RPCS1 = 000000          SELPAR  016750          SW5   = 000040          US2   = 000002
QDRV3   042532          RPCS2 = 000010          SETFMT= 000143          SW6   = 000100          US4   = 000004
QDRV4   042552          RPDA  = 000006          SETVEC  005304          SW7   = 000200          UWR   = 000010
QDRV5   042572          RPDB  = 000022          SET.IE  042306          SW8   = 000400          VUF   = 000002
QDRV6   042612          RPDS1 = 000012          SHDTYP  023226          SW9   = 001000          VU30  = 010000
QDRV7   042632          RPDT  = 000026          SIXTEE  001274          SYSTAT  053672          VV    = 000100
QINPT   042370          RPEC1 = 000044          SIZMEM  005062          T       051576          WAIT    001552
QOUTPT  042410          RPEC2 = 000046          SKI   = 040000          TABLE   055254          WAO   = 000002
QSTART  042430          RPERRS  034334          SKIER   012634          TABLE0  055274          WC      037654
QSTOP   042432          RPER1 = 000014          SLASH   055074          TABLE1  055342          WCE   = 040000
QTERM = 042652          RPER2 = 000040          SPOTCK  020344          TABLE2  055410          WCF   = 000040
QUES    054323          RPER3 = 000042          SRCHWT  034416          TABLE3  055456          WCFER   012672
QVCON   001362          RPINIT  034520          STACK = 001100          TABLE4  055524          WCKD  = 000151
QVSEK   001366          RPLA  = 000020          START   004166          TABLE5  055572          WCKER   010664
RANCYL  017204          RPMR  = 000024          START1  004200          TABLE6  055640          WCKHD = 000153
RANPAT  017452          RPOF  = 000032          START2  004230          TABLE7  055706          WCSEL   001420
RANSEC  017030          RPSN  = 000030          START3  004216          TAP   = 040000          WCU   = 000001
RANSIZ  017316          RPTMR   041100          STATHD  053730          TBITVE= 000014          WLE   = 004000
RANTRK  017104          RPVEC   034504          STATIN  001214          TD      037500          WLEER   012346
RANWRT  017472          RPWC  = 000002          STATIS  016030          TDF   = 000040          WRCHK = 000000
RANXIT  017462          RP04    035340          STATPR  023116          TIMER   034446          WRL   = 004000
RATIO   001422          RP04B   053711          STKLMT= 177774          TKVEC = 000060          WRTDAT= 000161
RAW   = 000020          RP05    053716          STNDAT  002762          TPVEC = 000064          WRTHD = 000163
RDCHR = 104410          RP06    053723          ST0     041172          TRAPVE= 000034          WRTPK   020050
RDDAT = 000171          RTC   = 000117          ST01    041222          TRE   = 040000          WRTPK1  020100
```

```
WRTPK2  020260          $DRVID= 000224          $MISPO= 000066          $RPCA = 000270          $TKB    001146
WRTPK3  020304          $DSPLY  027770          $MNEW   031635          $RPCC = 000272          $TKCNT  030362
WRTPK4  020336          $DTBL   033514          $MSWR   031624          $RPCS1= 000234          $TKINT  030400
WRT.AD  042072          $ENDAD  005662          $NCODE= 000074          $RPCS2= 000244          $TKQEN= 030377
WRT.RP  042002          $ERFLG  001103          $NCYL = 000100          $RPDA = 000242          $TKQIN  030364
WRT.R1  042066          $ERMAX  001115          $NEXT = 000104          $RPDB = 000256          $TKQOU  030366
WRT.R2  042152          $ERROR  031712          $NPATC= 000075          $RPDS1= 000246          $TKQSR  030370
WRT.R3  042160          $ERRPC  001116          $NSEC = 000076          $RPDT = 000262          $TKS    001144
WRT.R4  042164          $ERRTB  004026          $NTRK = 000077          $RPEC1= 000300          $TKSRV  030450
WRT.R5  042166          $ERRTY  032250          $NULL   001154          $RPEC2= 000302          $TN   = 000001
WRT.WD  042070          $ERTTL  001112          $NWRDL= 000102          $RPER1= 000250          $TNPWR  034042
WRU   = 000400          $FAIR = 000072          $OCNT   033304          $RPER2= 000274          $TOTAL= 000056
WSU   = 000004          $FILLC  001156          $OCTVL  034230          $RPER3= 000276          $TPB    001152
ZROIND  001276          $FILLS  001155          $OMODE  033306          $RPLA = 000254          $TPFLG  001157
$AUTOB  001134          $FIRST= 000122          $OPERC= 000036          $RPMR = 000260          $TPS    001150
$BDADR  001122          $FMT  = 000001          $PACK = 000026          $RPOF = 000266          $TRANS= 000046
$BDDAT  001126          $GDADR  001120          $PASS   001100          $RPSN = 000264          $TRAP   034246
$BDSEC= 000124          $GDDAT  001124          $PASSC= 000070          $RPVEC  001172          $TRAP2  034270
$BELL   001160          $GET42  005646          $PATTC= 000030          $RPWC = 000236          $TRK  = 000011
$BUF  = 000006          $GTSWR  030770          $POSIT= 000042          $SAVRE  033636          $TRP  = 000015
$CHARC  033056          $HARD = 000062          $PREVA= 000032          $SB2D   030302          $TRPAD  034302
$CKSWR  030700          $HD   = 000000          $PREVO= 000027          $SB2O   030332          $TSTNM  001102
$CMTAG  001100          $HINUM  033632          $PSEL = 000003          $SEC  = 000010          $TTYIN  031566
$CM3  = 000000          $ICNT   001104          $QUES   001164          $SETUP= 000146          $TYPDS  033310
$CNTLC  031605          $INTAG  001135          $RAND   033534          $SIZE   056142          $TYPE   032570
$CNTLG  031617          $ITEMB  001114          $RDCHR  031242          $SKI  = 000064          $TYPEC  032740
$CNTLU  031612          $LF     001166          $RDLIN  031332          $SOFT = 000060          $TYPEX  033060
$CODE = 000024          $LKCSB  001176          $RDSZ = 000017          $SSEC = 000022          $TYPOC  033106
$COMND= 000002          $LKCSR  001174          $READ = 000052          $STUP = 177777          $TYPON  033122
$CRLF   001165          $LKS    001202          $REG  = 000014          $SUPRS  027702          $TYPOS  033062
$CYL  = 000012          $LLVEC  001204          $RESRE  033674          $SVPC = 000224          $WRDL = 000020
$DBLK   033524          $LONUM  033634          $RETRY  015702          $SWR  = 122000          $WRDM = 000004
$DB2D   033732          $LPADR  001106          $RPADR  001170          $TATUS= 000016          $XOFF = 000023
$DB2O   034126          $LPERR  001110          $RPAS = 000252          $TERM = 000032          $XON  = 000021
$DECVL  034112          $LPVEC  001200          $RPBA = 000240          $TIME   023752          $OFILL  033305
$DOAGN  005672          $LSTAD  056272

. ABS.  057200     000
        000000     001
ERRORS DETECTED:  0

VIRTUAL MEMORY USED:  52736 WORDS  ( 206 PAGES)
DYNAMIC MEMORY AVAILABLE FOR  70 PAGES
CZRJDE.BIC,CZRJDE/C=CZRJDE.DOC,CZRJDE,[20,0]SYSMAC/M
```

```
CROSS REFERENCE TABLE (CREF V04.00 )

$0FILL   25-1      25-1#     25-1*     25-1*
$40CAT   22-1
$AUTOB    6-0#     11-34*    11-58*    21-1      21-1      21-1
$BDADR    6-0#
$BDDAT    6-0#
$BDSEC   14->42    18-127    33-59#    33-63
$BELL     6-0#     15-8      21-1      21-1      21-1      22-1      22-1      22-1
$BUF     12-88*    12-123*   14-107*   14-561    14-583    14-999    14-:77    14-:83    14-:88    14-:92    14-;31    15-84     15-229    15-236
         33-9#
$CHARC   24-1      24-1#     24-1*     24-1*     24-1*
$CKSWR   21-1#     31-1      31-1
$CM3      6-0       6-0#
$CMTAG    6-0#     11-29     11-29     11-29
$CNTLC   21-1      21-1      21-1      21-1      21-1#
$CNTLG   21-1      21-1#
$CNTLU   21-1      21-1      21-1#
$CODE    14-76     14-575    14-600    14-658    14-777    14-863    14-:04    14-:08    14-;29    14-;33    14-=07    14-=09    14-=50*   14-=60
         14->22*   14->26*   18-25*    18-34     32-831    32-833    32-841*   33-20#
$COMND   14-890*   14-910*   14-926*   14-944*   14-=52*   14->23*   14->27*   18-27*    32-842*   32-843*   33-6#
$CRLF     6-0#     11-138    11-171    11-175    13-43     13-47     13-59     13-65     13-69     13-89     13-93     14-517    14-526    14-526
         14-526    14-526    14-702    14-722    14-734    14-836    14-844    14-878    14-880    15-11     15-28     15-61     15-63     15-88
         15-101    15-120    15-140    15-159    15-174    15-197    15-220    15-239    15-260    15-272    15-285    15-292    15-325    15-334
         15-358    15-359    15-384    15-385    16-108    16-168    16-197    17-210    17-216    18-60     18-99     18-125    18-212    18-225
         18-226    18-231    19-18     19-30     19-34     19-49     21-1      21-1      21-1      21-1      22-1      22-1      22-1      23-1
         23-1      23-1      23-1      24-1      24-1      24-1
$CYL     14-99*    14-586    14-943*   14-:14    14-;35    14-;71    14-=41    14-=55*   14->02*   14->03    14->10*   15-147    18-82*    32-839
         33-12#
$DB2D    15-343    15-351    15-356    15-371    15-376    16-136    16-136    16-144    16-152    20-283    29-1#
$DB2O    20-300    30-1#
$DBLK    26-1      26-1      26-1#
$DECVL   29-1      29-1#
$DOAGN   11-207#
$DRVID   17-197    18-106*   18-109*   18-109*   18-109*   18-109*   18-109*   33-63#    33-67
$DSPLY   20-151#   31-2
$DTBL    26-1      26-1#
$ENDAD    5-9      11-203#
$ERFLG    6-0#     22-1      22-1      22-1*
$ERMAX    6-0#
$ERROR   11-29     22-1#
$ERRPC    6-0#     22-1      22-1      22-1      22-1*     22-1*     23-1      23-1
$ERRTB   10-0#     23-1
$ERRTY   22-1      23-1#
$ERTTL    6-0#     22-1      22-1      22-1*
$ESCAP   32-856    32-906    32-908    32-;81    32-<40    32->48
$FAIR    12-91*    12-104    12-106*   12-128*   33-35#
$FILLC    6-0#     24-1      24-1      24-1
$FILLS    6-0#     24-1      24-1
$FIRST   18-65*    33-55#
$FMT     14-125*   33-5#     33-6      33-7      33-8      33-9      33-10     33-11     33-12     33-13     33-14     33-18
$GDADR    6-0#
$GDDAT    6-0#
$GET42   11-198#   12-12     16-55     37-52
$GTSWR   21-1#     31-1      31-1
$HARD    16-164    16-188    16-188*   33-31#
$HD       4-54      4-54      4-54
$HINUM   11-29*    20-8      27-1      27-1      27-1#     27-1*
```

```
$ICNT     6-0#
$INTAG    6-0#     21-1      21-1      21-1      21-1      21-1*
$ITEMB    6-0#     22-1      22-1      22-1      22-1      22-1*     22-1*     23-1
$LF       6-0#     21-1      21-1      21-1      22-1      22-1      24-1      24-1
$LKCSB    7-0#     16-39*
$LKCSR    7-0#     16-33     16-40*
$LKS      7-0#     16-44     16-49*
$LLVEC    7-0#     16-46
$LONUM   11-29*    14-;92    20-7      27-1      27-1      27-1#     27-1*
$LPADR    6-0#
$LPERR    6-0#
$LPVEC    7-0#     16-36
$LSTAD   11-89     36-1#     36-1*
$MAIL    11-29     11-34     11-58     22-1      24-1
$MISPO   15-379    16-164    16-190    16-190*   33-33#
$MNEW    21-1      21-1#
$MSWR    21-1      21-1#
$NCODE   12-200*   14-;95*   14-<48    14-<71    14-<86    14-=09*   14-=10*   14-=16*   14-=39    14-=50    14-=51    33-39#    33-40     33-41
         33-42     33-43     33-44     33-45     33-49
$NCYL    12-192*   14-<47*   14-=55    33-43#
$NEXT    12-71     12-203*   14-<93*   14-=63*   14->30*   18-16     33-45#
$NPATC   12-202*   14-<92*   14-=53    33-40#
$NSEC    12-195*   14-<11*   14-=54    33-41#
$NTRK    12-194*   14-<27*   33-42#
$NULL     6-0#     24-1      24-1      24-1
$NWRDL   12-196*   12-199*   14-<50*   14-<53*   14-<67*   14-<80*   14-=56    14-=57    33-44#
$OCNT    25-1#     25-1*     25-1*
$OCTVL   30-1      30-1#
$OMODE   25-1      25-1#     25-1*     25-1*     25-1*     25-1*
$OPERC   14-;69*   14-;70*   14-=89    14-=91    15-341    15-369    16-121    19-39     33-25#
$PACK    12-73     12-97     12-133    14-=93    14->19    17-115*   33-21#
$PASS     6-0#
$PASSC   16-124    17-112*   19-20     19-25     19-47*    33-34#
$PATTC   14-;43    14-=53*   14->29*   18-28*    18-29*    33-23#
$POSIT   14-:16*   14-:17*   14-;73*   14-;74*   15-374    19-13     19-16     33-26#
$PREVA   14-;71    14-=41*   14-=42*   14-=45*   14-=46*   14-=47*   14-=98*   14-=99*   14->00*   15-187    15-191    15-195    15-203    32-839*
         32-840*   33-24#
$PREVO   14-=38*   14-=96*   15-38     32-838*   33-22#
$PSEL    33-7#
$QUES     6-0#     21-1      21-1      21-1      21-1      22-1      22-1      24-1      24-1
$R2A     31-1
$RAND    14-;83    14-<65    14-<83    14-=25    14-=88    27-1#
$RDCHR   21-1#     31-1      31-1
$RDDEC   31-1
$RDLIN   21-1#     31-1      31-1
$RDOCT   31-1
$RDSZ    21-1      21-1#
$READ    14-:12*   14-:13*   15-354    19-6      19-9      33-28#
$REG     18-16     33-13#    33-97     33-97     33-97     33-97     33-97     33-97     33-97     33-97     33-97     33-97     33-97     33-97
         33-97     33-97     33-97     33-97
$RESRE   28-1#     31-1
$RETRY   14-67     14-169    14-233    14-278    14-318    14-354    14-382    14-452    14-482    14-498    14-963#   14-981
$RPADR    7-0#     11-62     34-58     37-19     37-40     37-42
$RPAS    33-74#    34-60
$RPBA    14-562    14-767    14-798    14-819    14-851    14-871    14-998    15-85     15-135    15-235    15-246*   15-247    33-69#    34-60
$RPCA    14-:14    14->02    15-172    33-81#    34-61
```

```
$RPCC    12-192    14-99     14-298    14-943    14-=47    14->00    14->45    15-179    15-206    33-82#    34-61
$RPCS1   13-6      13-8      14-536    14-=38    14-=96    15-32     32-838    33-67#    33-68     33-69     33-70     33-71     33-72     33-73
         33-74     33-75     33-76     33-77     33-78     33-79     33-80     33-81     33-82     33-83     33-84     33-85     33-86     33-97
         33-97     33-97     33-97     33-97     33-97     33-97     33-97     34-59
$RPCS2   14-6      14-43     14-179    14-226    14-270    14-310    14-416    14-431    14-448    14-538    33-71#    34-59
$RPDA    14->01    15-168    15-218    16-15     16-19     16-24     33-70#    34-60
$RPDB    15-250    33-76#    34-60
$RPDS1   13-10     14-8      14-37     33-72#    34-59
$RPDT    14-92     14-<32    18-8      18-37*    32-352*   33-78#    34-60
$RPEC1   14-56     14-58     14-783    15-279    33-85#    34-59
$RPEC2   14-60     14-801    15-283    17-213    18-21     33-86#    33-97     33-97     33-97     33-97     33-97     33-97     33-97     33-97
         34-59
$RPER1   14-11     14-14     14-17     14-20     14-23     14-26     14-29     14-32     14-35     14-40     14-46     14-86     14-117    14-119
         14-159    14-188    14-201    14-203    14-255    14-290    14-540    14-974    14-977    33-73#    34-59
$RPER2   14-542    33-83#    34-59
$RPER3   14-49     14-544    33-84#    34-59
$RPLA    33-75#    34-60
$RPMR    33-77#    34-60
$RPOF    33-80#    34-61
$RPSN    33-79#    34-61
$RPVEC    7-0#     11-63
$RPWC    14-557    14-585    14-768    14-853    15-82     15-138    15-252    33-68#    34-60
$SAVRE   28-1#     31-1      31-1
$SB2D    14-794    15-421    16-125    16-160    16-160    16-160    16-160    16-165    16-199    16-204    16-209    20-281#
$SB2O    15-405    20-298#
$SEC     14-98*    14-588    14-941*   14-;37    14-=42    14-=54*   14->01*   14->09*   15-157    18-80*    32-840    33-10#
$SETUP    4-343     4-343     4-343     4-343     4-343#    4-343#    4-343#    4-343#    4-343#   11-29     11-29     11-29     11-29     11-29
         11-29     11-29     11-29     11-29     11-29     11-29     11-29     11-34     11-34     11-34     11-58     21-1      21-1      21-1
         21-1      21-1      22-1      22-1      22-1      22-1
$SIZE    11-88     36-1#
$SKI     15-382    16-164    16-189    16-189*   33-32#
$SOFT    16-164    16-187    16-187*   33-30#
$SSEC    14-100    14-102    14-594    14-598    14-599    14-771    14-855    14-=59*   14-=62*   14->21*   14->25*   15-255    18-33*    18-36*
         33-19#
$STUP     4-343     4-343     4-343     4-343     4-343#    4-343#    4-343#    4-343#    4-343#    4-343#    4-343#    4-343#
$SUPRS   14-795    15-344    15-352    15-357    15-372    15-377    15-422    20-122#
$SVPC     5-9       5-9#
$SWR      4-44#     4-54      4-55      4-55      4-55      4-55      4-55      4-55      4-55      4-55      6-0       6-0       6-0      11-29
         22-1      22-1      22-1      22-1      22-1      22-1      22-1      22-1      22-1      22-1      22-1
$STATUS  12-148    13-4      13-21     13-27     13-29     13-31     13-33     13-35     13-37     14-3      14-109    14-112    14-129    14-192
         14-914    14-948    14-964    14-970    14-996    15-79     15-226    33-14#    34-61
$TERM    31-3#
$TESTN   23-1
$TIME    15-13     16-106    16-195#   17-22     22-1
$TKB      6-0#     17-62     21-1      21-1      21-1      21-1      21-1      21-1      21-1      24-1      24-1
$TKCNT   21-1      21-1      21-1#     21-1*     21-1*     21-1*     21-6*
$TKINT   11-57     11-192    17-23     21-1      21-1      21-1#
$TKQEN   21-1      21-1      21-1#     21-9
$TKQIN   21-1      21-1      21-1#     21-1*     21-1*     21-1*     21-1*     21-7*     21-8*     21-9      21-11*
$TKQOU   21-1      21-1      21-1#     21-1*     21-1*     21-1*
$TKQSR   21-1      21-1      21-1      21-1#     21-11
$TKS      6-0#     17-63*    21-1      21-1      21-1      21-1      21-1      21-1*     21-1*     21-1*     21-1*     21-1*     21-1*     24-1
         24-1
$TKSRV   21-1      21-1#
$TN       4-45#     4-54
$TNPWR   29-1      29-1      29-1#
```

```
$TOTAL   15-346    16-161    16-191    16-191*   18-238    33-29#
$TPB      6-0#     24-1      24-1      24-1*
$TPFLG    6-0#     24-1      24-1      24-1
$TPS      6-0#     24-1      24-1      24-1
$TRANS   14-:02*   14-:03*   14-:10*   14-:11*   15-349    33-27#
$TRAP    11-29     31-1#
$TRAP2   31-1      31-1#
$TRK     14-97*    14-940*   14->06    14->08*   15-152    18-81*    33-11#
$TRP     31-1      31-1      31-1      31-1      31-1      31-1      31-1      31-1      31-1      31-1      31-1      31-1      31-1      31-1
         31-1      31-1      31-1      31-1      31-1      31-1      31-1      31-1      31-1      31-1      31-1      31-1      31-1      31-1
         31-1      31-1      31-1      31-1      31-1      31-1      31-1      31-1      31-1      31-1      31-1      31-1      31-1      31-1
         31-1      31-1      31-1#     31-1#     31-1#     31-1#     31-1#     31-1#     31-1#     31-1#     31-1#     31-1#     31-1#     31-1#
         31-2      31-2      31-2      31-2      31-2#
$TRPAD   31-1      31-1#     31-3
$TSTNM    6-0#     22-1      22-1      22-1      23-1
$TTYIN   21-1      21-1      21-1      21-1      21-1      21-1#
$TYPBN   31-1
$TYPDS   26-1#     31-1      31-1
$TYPE    20-141    20-154    24-1#     31-1      31-1
$TYPEC   21-1      24-1      24-1      24-1      24-1#
$TYPEX   24-1      24-1      24-1#
$TYPOC   25-1#     31-1      31-1
$TYPON   25-1      25-1#     31-1
$TYPOS   25-1#     31-1
$WRDL    14-100    14-102*   14-103    14-559    14-584    14-769    14-852    14-:10    14-:36    14-:43    14-:45*   14-:46    14-:47*   14-:75
          14-:84   14-:89    14-:93    14-;32    14-=56*   14->16*   15-81     15-232    15-253    18-30*    33-18#    33-19     33-20     33-21
         33-22     33-23     33-24     33-25     33-26     33-27     33-28     33-29     33-30     33-31     33-32     33-33     33-34     33-35
         33-39
$WRDM    14-103*   14-104*   14-=57*   14-=58*   14->17*   14->18*   18-31*    18-32*    33-8#
$XOFF    24-1      24-1
$XON     21-1      24-1      24-1
A16       4-80#
A17       4-81#
ABNRML   12-156    18-236#
ABS       4-258#
ACK       4-331#
ACL       4-296#    4-305#
ACTDRV   32-111#   32-412*   32-463*   32-776*   32-785*   32-:38
ACTSTR   32-117#   32-:40*   32-:55*
ACU       4-247#
AOE       4-162#
ASGN1    17-71#
ASGN2    11-183    17-70     17-84#
ASGN3    17-81     17-91     17-96#    17-120
ASGN4    17-95     17-97     17-117#
ASGN5    17-118    17-121#
ASGN6    17-104    17-123#
ASGN7    17-103    17-125#
ASGND    12-50     35-27#
ASKPAR   11-114    35-62#
ASNERR   17-82     17-160    17-178    18-207#
ASNLST    8-0#     12-10     12-57*    16-68     16-75     17-74     17-96     17-147    17-149*   17-171    17-181    17-200    18-222*   19-52*
ASNMSG   17-72*    17-78*    17-123*   17-128*   17-130*   17-132*   17-159*   17-177*   18-211#
ASSIGN   17-69#    17-138    17-222    17-228
AT0       4-182#
AT1       4-183#
```

```
AT2        4-184#
AT3        4-185#
AT4        4-186#
AT5        4-187#
AT6        4-188#
AT7        4-189#
ATA        4-149#
ATABIT    12-57     16-75     17-74     17-96     17-147    17-149    17-171    17-181    17-200    18-222    19-52     32-183#   32-368    32-475
          32-580    32-964    32-983    32-991    32-:08    32-;14
ATTN       7-0#     22-1*     34-54     34-55
AUTOCK     7-0#     14-:06    14-=00    32-829    35-58
AVAIL      8-0#     12-13     12-51     12-55*    12-58     12-68     12-70     12-112    12-113    12-159
BADENT    18-162    18-190    35-41#
BADSEC     7-0#     12-155*   14->68*   15-59     16-187    16-188    16-189    16-190    16-191
BADTMO    11-3#     11-31
BAI        4-101#
BEGCOD     7-0#     18-25     18-26
BEGPAT     7-0#     18-28
BEGSIZ     7-0#     18-30     18-31
BIT0       4-72#
BIT00      4-72      4-72#    11-163    14-;33    17-145    18-52     22-1      22-1
BIT01      4-72      4-72#    11-166    13-35     14-92     14-:04    14-<32    18-55     32-455    32-702
BIT02      4-72      4-72#    14-575    14-;29    18-62
BIT03      4-72      4-72#    11-152    14-3      14-380    32-347    32-923    32-<34
BIT04      4-72      4-72#    14-3      14-14     32-958
BIT05      4-72      4-72#    14-481    32-273    32-722    32-938    32-:88
BIT06      4-72      4-72#    14-85     14-86     14-188    14-203    14-996    15-226    17-63     32-377    32-446    32-520    32-819    32-<94
BIT07      4-72      4-72#    13-21     14-17     14-996    32-377    32-654    32-797    32-923    32-938    32-958    32-987    32-<66
BIT08      4-72      4-72#    14-11     32-377    32-;01
BIT09      4-72      4-72#    13-33     22-1      32-345    32-:86
BIT1       4-72#
BIT10      4-72#    13-31     14-29     14-37     22-1      32-704
BIT11      4-72#    13-29     14-32     32-320    32-479    32-497    32-673    32-:09
BIT12      4-72#    13-27     14-40     14-85     14-86     14-119    14-297    14-301    14-335    14-587    14-;36    15-214    32-315    32-359
          32-377    32-457    32-491    32-669    32-688    32-700    32-713    32-948    32-950    32-;74    32-<29    32-<95
BIT13      4-72#    14-20     14-352    14-448    20-151    22-1      32-440    32-899    32-;77
BIT14      4-72#    13-8      13-10     13-35     14-6      14-8      14-43     14-49     14-179    14-226    14-270    14-310    14-416    14-431
          32-452    32-488    32-886    32-:25    32-;41    32-;84    32-;88    32-<92
BIT15      4-72#    13-6      14-49     14-85     14-159    14-201    14-448    32-440    32-452    32-455    32-457    32-488    32-491    32-673
          32-702    32-704    32-819    32-923    32-938    32-948    32-958    32-:25    32-:86    32-;01    32-;41    32-;48
BIT2       4-72#    13-37     32-;48
BIT3       4-72#    14-23
BIT4       4-72#    14-276
BIT5       4-72#    14-26
BIT6       4-72#    14-119    15-79     32-481
BIT7       4-72#    14-112    14-316    14-970
BIT8       4-72#    14-231    14-255    14-260    14-290    14-295    14-448
BIT9       4-72#    14-35     14-448
BLKADR     8-0#     11-71     16-73     17-106    17-107    17-151    17-174    32-351
BOUND     37-14#    37-26*    37-35*    38-36
BPTVEC     4-72#
BUFTBL     8-0#     11-90*    11-91*    11-92*    11-93*    11-95*    11-96*    11-101*   11-105    11-107    11-108    14-:33    14-:35    14-:49*
          14-:72    14-:73    14-:85*   14-:90*   14-:95    14-:96    14-;09*
BUSADR    11-61     37-16#
BUSY      17-15     35-42#
CFLAG      7-0#     11-81*    12-166    17-21*    17-27     18-92*    18-102    18-118*   18-138    18-173*   18-185    18-195*   21-5*
```

```
CHGADR    7-0#     11-14*   11-18*   11-23*   37-16    37-18*
CI1      32-510   32-538#
CI3      32-512   32-561#
CI4      32-501   32-582#
CI5      32-560   32-581   32-659#  32-668
CI6      32-598   32-608   32-610   32-612   32-664#
CI7      32-442   32-519   32-550   32-554   32-558   32-566   32-575   32-579   32-590   32-597   32-603   32-607   32-621   32-627
         32-631   32-641   32-652   32-667   32-669#  32-771   32-801   32-975   32-<78
CI7B     32-672   32-675#  32-977
CI8      32-670   32-678   32-687#  32-853   32-;33
CK.CHR   20-207#  20-244   20-252   38-21    38-30
CK.DEC   20-185#  20-213
CK.DIG   18-140   18-146   18-152   18-187   20-238#
CK.NUM   37-27    37-36    38-15#
CK.OCT   20-167#  20-215
CKBUS    13-13    14-557#
CKCLK    11-129   16-29#
CKCLK1   16-31    16-42#
CKCLK2   16-43    16-51#
CKCLK3   16-41    16-50    16-58#
CKERR    13-12    14-536#
CKFMT    14-16    14-255#
CKHCE    14-19    14-290#
CKSWR    22-1     22-1     31-1#
CLKFLG    7-0#    16-29*   16-34*   16-45*   16-195
CLOCK    16-37    16-47    16-216#
CLR       4-103#
CLRDPB   17-108   18-7#
CLRQUE   32-255   32-721   32-;11   32-=62#
CMCNT     7-0#    14-584*  14-585*  14-594   14-596   14-599*  14-656
CMCYL     7-0#    14-586*  14-587*  14-602   14-668*
CMDAT    14-601   14-618   14-622#
CMHED    14-602#
CMPAR    13-16    14-575#
CMPARD   14-148   14-579#
CMPLMT    7-0#    14-531   14-589   35-53
CMPRES   12-32    12-83    12-115   12-140   12-163   12-181   12-210   14-=73#  14-=76
CMPRT    14-620   14-634   14-644   14-654   14-675#
CMPRX    14-615   14-617   14-657   14-670#
CMSEC     7-0#    14-588*  14-605   14-660*  14-661   14-663*
CMSTR    14-591#  14-659   14-662   14-666   14-669
CMTRK     7-0#    14-664*  14-665   14-667*
COLON    16-202   16-207   35-36#
COMTBL    8-0#    14-=52   18-27
CPSAVE   22-1     22-1     22-1     22-1     22-1     22-1#    22-1*    22-1*    23-1
CR        4-72#   24-1     24-1     34-48    34-50    34-51
CRLF      4-72#   11-6     11-34    11-34    24-1     24-1     34-104   34-105   34-106   34-108   34-110   34-111   34-113   34-114
         34-115   34-116   35-3     35-3     35-14    35-18    35-20    35-21    35-24    35-25    35-26    35-27    35-28    35-28
         35-31    35-32    35-34    35-37    35-38    35-39    35-39    35-40    35-41    35-42    35-43    35-44    35-44    35-130
         38-49    38-49    38-50    38-51    38-52    38-52
CSF       4-234#   4-252#
CSU       4-236#   4-254#
CTRAP    21-1     21-1     21-5#
CYLDER   14-297*  14-298   14-301*  14-335*  15-213   15-271   15-271   15-271   15-271   33-101   35-103#
CYLIMT    7-0#    14-<31*  14-<34*  14-<40   14-<44   14-<46   18-61*   18-64*   18-67    18-75    18-76    18-140
DATAO     9-0      9-0#
```

```
DATA1    9-0       9-0       9-0       9-0       9-0       9-0       9-0       9-0       9-0       9-0       9-0       9-0       9-0       9-0
         9-0       9-0       9-0#
DATAPK  17-53     17-221#
DATE     7-0#     17-188    17-191    35-117*   35-120*   35-120*   35-120*   35-120*   35-120*   35-120*   35-120*
DATEIS  17-190    35-37#
DCK      4-168#
DCKER   14-48     14-56#
DCKER1  14-81#    14-370
DCL      4-297#    4-306#
DCU      4-303#
DDISP    4-72#     6-0      11-29
DE1      4-139#
DEASGN  17-43     17-142#
DEASSG  12-28     35-25#
DFF20    4-135#
DH1     10-8      34-42#
DH14    15-64     34-47#
DH15    15-73     34-49#
DH16    15-76     34-51#
DH2     10-15     10-36     34-43#
DH3     10-22     34-44#
DH4     10-29     34-45#
DH6     10-43     34-46#
DIGB     4-136#
DISPLA   6-0#     11-29*    11-29*    22-1*
DISPLY  13-49     13-79     13-95     13-106    14-63     14-80     14-115    14-152    14-162    14-183    14-222    14-246    14-266    14-306
        14-332    14-347    14-369    14-375    14-393    14-403    14-412    14-427    14-443    14-463    14-474    14-493    14-511    14-516
        14-517    14-518    14-526    14-526    14-526    14-526    14-526    14-526    14-526    14-526    14-547    14-565    14-696    14-701
        14-702    14-703    14-716    14-719    14-722    14-731    14-734    14-791    14-796    14-828    14-833    14-833    14-833    14-836
        14-841    14-841    14-841    14-843    14-844    14-867    14-873    14-878    14-880    14-987    15-11     15-14     15-28     15-54
        15-56     15-61     15-62     15-63     15-64     15-65     15-68     15-73     15-76     15-87     15-88     15-98     15-101    15-107
        15-113    15-120    15-134    15-137    15-140    15-146    15-149    15-150    15-154    15-155    15-159    15-167    15-170    15-171
        15-174    15-181    15-184    15-186    15-189    15-193    15-197    15-202    15-205    15-212    15-216    15-220    15-228    15-231
        15-234    15-239    15-245    15-249    15-258    15-260    15-266    15-271    15-271    15-271    15-271    15-272    15-278    15-281
        15-282    15-285    15-291    15-292    15-298    15-304    15-310    15-316    15-323    15-325    15-333    15-334    15-340    15-345
        15-348    15-353    15-358    15-359    15-368    15-373    15-378    15-381    15-384    15-385    15-394    15-408    20-130    31-2#
DISPRE   5-1#     11-29
DL64     4-138#
DLT      4-113#
DMD      4-172#
DONE    13-23     14-3#
DPINT   32-70#    32-284    32-287    32-385*   32-422    32-870    32-895    32-:15    32-:17*   32-:78    32-;16    32-;27    32-;35*
DPR      4-142#
DPRQS   32-83#    32-428    32-514*   32-873    32-:29*   32-:80    32-;18    32-;29    32-;44*
DRIVE    7-0#     22-1*     34-55     34-56     34-57
DRIVE0   8-0      17-197    33-97#    35-101    35-101    35-101    35-101    35-101    35-101
DRIVE1   8-0      33-97#    35-101    35-101    35-101    35-101    35-101    35-101
DRIVE2   8-0      33-97#    35-101    35-101    35-101    35-101    35-101    35-101
DRIVE3   8-0      33-97#    35-101    35-101    35-101    35-101    35-101    35-101
DRIVE4   8-0      33-97#    35-101    35-101    35-101    35-101    35-101    35-101
DRIVE5   8-0      33-97#    35-101    35-101    35-101    35-101    35-101    35-101
DRIVE6   8-0      33-97#    35-101    35-101    35-101    35-101    35-101    35-101
DRIVE7   8-0      33-97#    35-101    35-101    35-101    35-101    35-101    35-101
DRNUM   13-45     13-67     13-91     18-228    19-28     19-32     35-26#
DROP    13-55     13-101    18-220#   18-240
DROPNG  18-227    35-22#
```

```
DRQ       4-207#
DRVACT   32-27#    32-436    32-615*   32-662*   32-681*   32-692    32-707*   32-790*   32-891    32-921    32-952*   32-959*   32-972    32-982*
         32-:89*   32-:95    32-;02*
DRVCLR    4-326#
DRVER    14-52     14-245#
DRVINT   32-275    32-309#   32-424    32-483    32-999    32-:18    32-:93
DRVPRM   17-109    18-43#
DRVQUE   32-431    32-444    32-835    32->02#
DRVSTA   11-142    17-102    32-41#    32-265*   32-266*   32-267*   32-268*   32-278*   32-310*   32-319*   32-374*   32-379*   32-418    32-426
         32-450    32-485    32-489    32-616*   32-725*   32-876    32-884    32-897    32-946*   32-953*   32-965    32-:20    32-;36*
DRVTYP   11-145    11-152    11-163    11-166    17-125    18-52     18-55     18-62     32-57#    32-311*   32-326*   32-331*   32-336*   32-341*
         32-347*   32-453    32-726*
DRY       4-141#
DSWR      4-72#     6-0      11-29
DT00      4-198#
DT01      4-199#
DT02      4-200#
DT03      4-201#
DT04      4-202#
DT05      4-203#
DT06      4-204#
DT07      4-205#
DT08      4-206#
DT1      10-9      34-54#
DT14     15-69     34-59#
DT15     15-74     34-60#
DT16     15-77     34-61#
DT2      10-16     10-37     34-55#
DT3      10-23     34-56#
DT4      10-30     34-57#
DT6      10-44     34-58#
DTE       4-165#
DTEER    14-42     14-366#
DTSY      4-178#
DTUW     17-98     32-177#   32-261    32-503    32-559*   32-682    32-685*   32-695    32-708    32-710*   32-719*   32-779    32-791*   32-:83
         32-:98    32-;09*   32-;20    32-;82
DUNIT     8-0#     12-4      17-151*   18-224*   19-54*
DVA       4-130#
ECBAD0    7-0#     14-809*   14-833
ECBAD1    7-0#     14-823*   14-841
ECBIT     7-0#     14-783*   14-784*   14-785    14-786*   14-803    14-805*
ECC      14-141    14-767#
ECC1     14-816    14-822    14-828#
ECC2     14-800    14-843#
ECCX     14-835    14-842    14-844#
ECGD      7-0#
ECGD1     7-0#
ECH       4-159#
ECI       4-275#
ECMSK0    7-0#     14-801*   14-806*   14-811    14-813
ECMSK1    7-0#     14-802*   14-807*   14-815    14-824    14-826
ECSEC     7-0#     14-767*   14-776*   14-779*   14-781*   14-797
ECWRD     7-0#     14-785*   14-787*   14-788*   14-789*   14-790*   14-792    14-797*   14-798    14-809    14-812    14-813*   14-814*   14-817
         14-833    14-833
ECWRD1    7-0#     14-810*   14-817*   14-818*   14-819    14-821*   14-823    14-825    14-826*   14-827*   14-834    14-841    14-841
EM1      10-7      34-3#
```

```
EM10     13-60     34-9#
EM11     13-66     34-10#
EM12     13-44     13-49     34-11#
EM13     13-79     34-12#
EM14     13-90     13-95     34-13#
EM15     13-106    34-14#
EM2      10-14     34-4#
EM20     14-222    34-15#
EM21     14-80     34-16#
EM22     14-178    34-17#
EM23     14-162    34-18#
EM24     14-266    34-19#
EM25     14-306    34-20#
EM26     14-412    34-21#
EM27     14-427    34-22#
EM3      10-21     34-5#
EM30     14-246    34-23#
EM31     14-347    34-24#
EM32     14-369    34-25#
EM33     14-375    34-26#
EM34     14-474    34-27#
EM35     14-393    34-28#
EM36     14-403    34-29#
EM37     14-181    34-30#
EM4      10-28     34-6#
EM40     14-443    34-31#
EM41     14-565    34-32#
EM42     14-696    34-33#
EM43     14-511    14-516    34-34#
EM44     14-547    34-35#
EM45     14-63     34-36#
EM46     15-87     34-37#
EM5      10-35     34-7#
EM50     14-463    34-38#
EM51     14-332    34-39#
EM6      10-42     34-8#
EM60     14-493    34-40#
EMPTYQ   32-679    32-724    32-820    32-960    32-:27    32-;49    32-=87#
EMTVEC    4-72#    11-29*    11-29*    11-40*
ENDCMP   14-670    14-727#
ENDCN     7-0#     11-36     11-37
ENDCON    7-0#     11-36*    11-37*    19-6      19-9
ENDET     7-0#     19-4      35-56
ENDPAS   19-19     35-23#
ENDPGM   11-91     11-93     35-105#
ENDSEK    7-0#     11-38*    11-39*    19-13     19-16
ENDSK     7-0#     11-38     11-39
ENDTST   19-27     35-24#
ENTADR   18-119    35-35#
ENTCOM   17-24     35-32#
ENTDAT   35-114    35-130#
ENTDRV   18-93     35-33#
ENTID    35-121    35-131#
ENTLMT   18-47     35-34#
ENTPR    11-120#
EOP      12-157    19-4#
```

CROSS REFERENCE TABLE (CREF V04.00)

```
EOP1     19-5      19-8      19-13#
EOP2     14->11    19-7      19-10     19-14     19-18#
EOPX     19-11     19-15     19-17     19-48     19-55#
ERCTR     7-0#     14-582*   14-619*   14-630    14-643*   14-729    14-732
ERPRC1   13-22     13-27#    14-116    14-988
ERPROC   13-5      13-9      13-11     13-21#
ERR       4-148#
ERROR     4-72#
ERRVEC    4-72#    11-29     11-29*    11-29*    11-31*    11-32*    16-31*    16-32*    16-43*    16-58*    22-1      22-1*     22-1*     36-1
         36-1      36-1*     36-1*     36-1*     36-1*     37-38     37-39*    37-41*    37-47*
EXT1      4-214#
EXT10     4-217#
EXT2      4-215#
EXT20     4-218#
EXT4      4-216#
EXT40     4-219#
F1        4-125#
F2        4-126#
F3        4-127#
F4        4-128#
F5        4-129#
FACTOR   14-998*   14-999*   14-:01*   14-:02    14-:12    14-:20#
FAIRNS    7-0#     12-104
FALPAR   13-32     13-65#
FALPR1   13-61     13-67#
FEN       4-240#
FER       4-157#
FILBUF   12-90     12-126    14-;28#
FMT22     4-276#
FMTER    14-262    14-411#
FORMAT    7-0#     12-190    14-=13    35-57
FRSTER    7-0#     14-147*   14-149    14-508    14-529*   14-530*   14-578*   14-646    14-677    14-692    14-704*   14-727
GENDPB   14-125*   14-129    14-889*   14-890*   14-892    14-909*   14-910*   14-912    14-914    14-925*   14-926*   14-928    14-940*   14-941*
         14-942*   14-943*   14-944*   14-946    14-948    17-100*   33-101#
GENREG   14-260    14-295    33-102    33-103#
GETADR   17-111    18-115#
GETBUF   12-87     12-122    14-106    14-:30#
GETID    17-110    18-89#
GETPAR   12-85     14-=37#
GETPAT   12-201    14-<91    14-=21#   14-=26    14->28
GETREG    4-333#
GETREM   14-;87    14-<06    14-<22    14-<42    14-<62    14-<99    14-=22    20-7#
GETREQ   32-476    32-697    32-809    32-880    32-913    32-974    32-:24    32-:85    32-;00    32-;38    32-;45    32->23#
GNS       5-1       5-1      11-6      11-34     31-1      31-1      31-1      31-1      31-1      31-1      31-1      31-1      31-1      31-1
         31-1      31-1      31-1      31-1      31-1      31-1      31-1      31-1      31-1      31-1      31-1      31-1      31-2      31-2
GO        4-124#
GODRIV   12-92     12-127    14-108    14-191    14-963    14-;63#
GRV       4-137#
GTSWR    11-34     11-58     31-1#
HCE       4-160#
HCEER    14-302    14-426#
HCI       4-274#
HCRC      4-161#
HCRCER   14-13     14-219#   14-257    14-292
HEDLIN   17-196    35-39#
HIAD     37-12#    37-26
```

```
HIVEC    37-13#    37-35
HOUR      7-0#     11-76*    16-198    16-228*   16-229    16-231*
HT        4-72#    24-1      24-1
HZ        7-0#     11-75     16-218
IAE       4-163#
IAEER    14-31     14-392#
IBSAVE   22-1      22-1      22-1      22-1      22-1#     22-1*     22-1*     22-1*
IDIS     17-194    35-38#
IDLE     12-69     12-125    12-145#   12-169
IDLE1    12-147    12-166#
IE        4-78#
ILF       4-153#
ILR       4-154#
INCHRD   14-133    16-188#
INCMIS   14-337    16-190#
INCSKI   14-467    16-189#
INCSOF   14-140    14-146    14-229    14-274    14-314    16-187#
INCTOT   13-53     13-70     13-83     13-99     13-110    14-65     14-114    14-134    14-151    14-167    14-209    14-212    14-230    14-249
         14-275    14-315    14-338    14-351    14-379    14-396    14-405    14-420    14-435    14-447    14-466    14-478    14-496    14-551
         14-569    14-735    16-191#
INTDON   11-193    35-43#
INTRVL    7-0#     11-79*    16-223*   16-234    16-236    16-236    16-239*   35-51
INVLD    17-59     35-31#
IOTVEC    4-72#
IR        4-104#
ISR      32-270    32-776#
IXE       4-244#    4-262#
KIPAR0   36-1
KSR      12-168    17-9#
KSR1     17-12     17-19#
LA       32-507    32-741#
LACNT    32-136#   32-749*   32-750    32-767*
LF        4-72#    24-1      24-1      34-48     34-50     34-51     35-14     35-21     35-39     35-44
LIMIT     7-0#     14-531*   14-589*   14-590*   14-706    14-708*
LIN10A   14-791    34-109#
LIN10B   14-796    34-110#
LIN10C   14-843    34-111#
LIN10H   14-828    34-113#
LIN11H   14-867    34-115#
LIN2C    15-31     34-63#
LIN2P    15-37     34-64#
LIN2S    15-62     34-65#
LIN3.1   15-108    15-114    15-179#
LIN3.3   15-119    15-126    15-202#
LIN3.4   15-127    15-212#
LIN4SP   11-141    17-203    34-117#
LIN6.1   15-299    15-305    15-321#
LIN6.2   15-311    15-317    15-330#
LIN7M    15-378    34-95#
LIN7O    15-340    15-368    34-96#
LIN7P    15-373    34-97#
LIN7R    15-353    34-101#
LIN7S    15-381    34-98#
LIN7T    15-345    34-99#
LIN7X    15-348    34-100#
LIN8M    14-115    14-987    15-394    34-102#
```

```
LIN9B    14-701    34-103#
LIN9E    14-731    34-107#
LIN9G    14-152    34-108#
LIN9H    14-703    34-104#
LIN9I    14-518    34-106#
LINA3    15-212    34-73#
LINB3    15-134    34-77#
LINB5    15-249    34-85#
LINB6    15-291    15-304    34-90#
LINC6    15-298    34-91#
LINCA3   15-171    34-75#
LIND5    15-245    34-84#
LINDA3   15-167    34-76#
LINDEC   14-733    15-67     15-148    15-153    15-158    15-180    15-183    15-185    15-188    15-192    15-196    15-204    15-207    15-215
         15-219    15-233    15-238    15-259    15-332    15-347    15-380    15-383    15-420#
LINE1    13-48     13-78     13-94     13-105    14-62     14-79     14-161    14-177    14-221    14-245    14-265    14-305    14-331    14-346
         14-368    14-374    14-392    14-402    14-411    14-426    14-442    14-462    14-473    14-492    14-510    14-546    14-564    14-695
         15-6#
LINE2    13-50     13-80     13-96     13-109    14-64     14-81     14-166    14-187    14-223    14-247    14-267    14-307    14-333    14-348
         14-376    14-394    14-404    14-413    14-428    14-444    14-464    14-475    14-494    14-512    14-548    14-566    14-697    15-27#
         15-395
LINE2A   15-36     15-41     15-59#
LINE2B   15-60     15-63#
LINE3    13-51     13-81     13-97     13-109    14-82     14-166    14-187    14-224    14-248    14-268    14-308    14-349    14-414    14-429
         14-445    14-495    14-549    15-107#
LINE3A   14-476    14-513    14-698    15-113#
LINE3B   14-465    15-119#
LINE3C   14-334    15-126#
LINE3D   14-567    15-89     15-132#
LINE3E   14-377    15-146#
LINE3F   14-395    15-165#
LINE4    13-52     13-82     13-98     13-109    14-83     14-166    14-187    14-225    14-269    14-309    14-350    14-378    14-415    14-430
         14-446    14-477    14-514    14-550    14-568    14-699    15-90     15-226#
LINE5    14-166    14-187    14-228    14-272    14-312    14-418    14-433    15-245#
LINE5A   14-273    14-313    14-336    14-419    14-434    15-266#
LINE5B   14-139    15-278#
LINE6    14-88     15-291#
LINE6A   14-145    15-298#
LINE6B   14-138    15-304#
LINE6C   14-69     14-171    14-196    14-205    14-235    14-280    14-320    14-356    14-384    14-454    14-484    14-500    15-310#
LINE6D   14-71     14-143    14-173    14-238    14-283    14-323    14-359    14-387    14-456    14-487    14-502    15-316#
LINE7    13-54     13-84     13-100    13-111    14-72     14-135    14-153    14-210    14-213    14-236    14-239    14-250    14-281    14-284
         14-321    14-324    14-357    14-360    14-385    14-397    14-406    14-421    14-436    14-457    14-485    14-503    14-552    14-570
         14-736    14-984    15-340#
LINE7A   14-339    14-468    15-368#
LINE8    14-132    14-207    14-983    15-394#
LINEN3   15-205    34-72#
LINEO5   15-282    34-89#
LINEP5   15-278    34-88#
LING6    15-310    34-92#
LINKDV   14-772    14-856    14-<74    15-256    20-10     20-20#
LINM3    15-107    34-66#
LINM4    15-228    34-81#
LINN3    15-113    34-68#
LINOCT   14-526    14-526    14-526    14-526    14-526    14-526    14-526    14-526    14-715    14-718    14-721    14-833    14-833    14-833
         14-841    14-841    14-841    14-870    14-875    15-96     15-136    15-139    15-169    15-173    15-230    15-248    15-251    15-271
         15-271    15-271    15-271    15-280    15-284    15-404#
LINP3    15-186    34-70#
LINP5    15-258    34-86#
LINR6    15-333    34-93#
LINS3    15-146    15-202    34-71#
LINS4    15-231    34-82#
LINS5    15-266    34-87#
LINSP    11-161    14-526    14-526    14-526    14-526    14-716    14-719    14-833    14-833    14-833    14-841    14-841    14-841    14-873
         15-68     15-98     15-149    15-154    15-170    15-271    15-271    15-271    15-271    15-281    16-123    16-128    16-136    16-136
         16-144    34-118#   37-23     37-32
LINSPO   15-14     15-65     16-152    16-160    16-160    16-160    16-160    34-119#
LINSS3   15-155    34-80#
LINST3   15-150    34-79#
LINT3    15-216    34-74#
LINUO6   15-316    34-94#
LINW3    15-137    34-78#
LINX4    15-234    34-83#
LKPAR    11-112#
LOADRV   11-55     38-49#
LST       4-144#
LSTAD     7-0#     11-89*    11-92     11-97     11-188
M.DP40   20-52#    20-73
M.DP41   20-57     20-64#
M.DP42   20-63     20-68#
M.DP44   20-75     20-80#
M.DP50   20-50     20-84#
M.DPID   20-27     20-41#
MAIN     12-3#     12-182    14->15
MAIN1    11-194    11-200    12-11     12-37#
MAIN2    12-47     12-65#    12-141
MAIN3    12-66     12-120#
MASK      7-0#     14-85*    14-117    14-163*   14-231*   14-276*   14-316*   14-352*   14-380*   14-451*   14-481*   14-972    14-977
MATCH    14-622    14-747#
MAXCYL   14-<35    14->03    18-67*    33-49#    33-50     33-51     33-52     33-53     33-54     33-55     33-59     35-101    35-101    35-101
         35-101    35-101    35-101    35-101    35-101
MAXDL     7-0#     11-102    11-104*   11-105    11-107*   11-122    11-124*   12-197    12-199    14-<51    14-<53    14-<58    14->16    14->17
         35-50
MAXER     7-0#     11-82*    18-238
MAXSEC   14-;99    18-71*    33-53#    35-101    35-101    35-101    35-101    35-101    35-101    35-101    35-101
MAXTRK   14-<15    14->06    18-69*    33-51#    35-101    35-101    35-101    35-101    35-101    35-101    35-101    35-101
MCLK      4-173#
MCPE      4-83#
MCPEMX   32-195#   32-;64    32-<10
MHS       4-242#    4-260#
MINCYL   14-<36    14-<38    14-<43    14->10    18-68*    18-82     33-50#    35-101    35-101    35-101    35-101    35-101    35-101    35-101
         35-101
MINSEC   14-<00    14-<02    14-<07    14->09    18-21     18-72*    18-80     33-54#    35-101    35-101    35-101    35-101    35-101    35-101
         35-101    35-101
MINTRK   14-<16    14-<18    14-<23    14->08    18-70*    18-81     33-52#    35-101    35-101    35-101    35-101    35-101    35-101    35-101
         35-101
MINUTE    7-0#     11-77*    16-203    16-224*   16-225    16-227*
MINX      4-174#
MNDLTA   32-211#   32-765
MNTBL     8-0#     15-52
MOH       4-208#
MOL       4-146#
```

```
CROSS REFERENCE TABLE (CREF V04.00 )

MONTR  11-177  11-181#
MPE     4-106#
MRD     4-176#
MRHVEC 37-29   37-55#
MRPCS1 37-20   37-54#
MSE     4-237#
MSTCK   4-175#
MWR     4-177#
MXDLTA 32-208# 32-763
MXF     4-107#
MXLACT 32-205# 32-750
MXWNDW 32-214# 32-568
NBA     4-210#
NED     4-110#
NEDCLK 16-52   35-28#
NEM     4-109#
NEWASN 17-39   17-137#
NEWUNT  8-0#   12-41   12-55   12-56*  17-106*
NHS     4-243#  4-261#
NOLOAD 11-190  38-52#
NOMTCH 14-508# 14-624
NONE   17-206  35-40#
NOTAVL 17-78   35-12#
NOTPRS 11-150  17-130  35-11#
NOTPRT  7-0#   14->66  35-59
NOTRP  11-148  17-128  35-10#
NOTSAF 11-158  17-123  35-13#
NOUSE  11-154  35-4#
OCYL    4-299#  4-309#
OF100   4-269#
OF200   4-270#
OF25    4-267#
OF400   4-271#
OF50    4-268#
OF800   4-272#
OFFCOD  8-0#   14-125
OFFSET  4-328# 14-926
OFFST  14-128  14-925#
OFLIN  13-36   13-89#
OFMSG0  8-0     8-0     8-0#
OFMSG1  8-0     8-0#
OFMSG2  8-0     8-0#
OFMSG3  8-0     8-0     8-0#
OFMSG4  8-0     8-0     8-0#
OFMSG5  8-0     8-0#
OFMSG6  8-0     8-0#
OFMSG7  8-0     8-0#
OFMSG8  8-0     8-0#
OFMSG9  8-0     8-0#
OFMSGA  8-0     8-0#
OFMTBL  8-0#   15-322
OFREV   4-273#
OPE     4-307#
OPERID  7-0#   17-192  17-195  35-124* 35-127* 35-127* 35-127* 35-127* 35-127*
OPI     4-166#
OPIER  14-22   14-344#
```

```
OPIER1   14-353#
OPRDAT   11-67     35-114#
OPT      32-438    32-473#   32-812    32-992    32-:30
OPTBL     8-0#     15-43     15-45
OR        4-105#
ORDERQ    8-0#     11-69     12-93     12-129    12-145    17-9
PACK      7-0#     11-80*    17-113    17-115    17-137*   17-221*   17-227*
PAR       4-156#
PAR1     35-50     35-65#
PAR10    35-57     35-74#
PAR11    35-54     35-75#
PAR14    35-55     35-76#
PAR15    35-58     35-77#
PAR16    35-56     35-78#
PAR19    35-52     35-79#
PAR2     35-51     35-66#
PAR20    35-59     35-80#
PAR3     35-53     35-67#
PAR4     35-68#    35-101    35-101    35-101    35-101    35-101    35-101    35-101    35-101
PAR5     35-69#    35-101    35-101    35-101    35-101    35-101    35-101    35-101    35-101
PAR6     35-70#    35-101    35-101    35-101    35-101    35-101    35-101    35-101    35-101
PAR7     35-71#    35-101    35-101    35-101    35-101    35-101    35-101    35-101    35-101
PAR8     35-72#    35-101    35-101    35-101    35-101    35-101    35-101    35-101    35-101
PAR9     35-73#    35-101    35-101    35-101    35-101    35-101    35-101    35-101    35-101
PARENT   11-121    18-79     18-172#
PARER    14-25     14-374#
PARLST   11-108*   11-120    35-50#
PARQ      8-0#     12-77     12-99     12-135    12-176    12-178    12-180    12-204
PASCNT    7-0#     19-25     35-52
PAT       4-102#
PCLOCK    7-0#     16-30*    16-35*
PDONE    35-21#
PERIOD   35-29#
PFECH    23-1      23-1#
PFECH1   23-1      23-1#
PFECH2   23-1      23-1#
PFECH3   23-1      23-1#
PFECH4   23-1      23-1#
PFTSTN   23-1      23-1#     23-1*
PGE       4-108#
PGM       4-143#
PIP       4-147#
PIRQ      4-72#
PIRQVE    4-72#
PLU       4-246#    4-263#
POPQUE   32-487    32-538    32-653    32-986    32->38#
POSER    14-300    14-330#
PR0       4-72#
PR1       4-72#
PR2       4-72#
PR3       4-72#
PR4       4-72#    17-20
PR5       4-72#
PR6       4-72#    11-32
PR7       4-72#    11-136
PROCES   12-154    13-3#
```

```
PRTBAD   14-84     14-136    14-479    14-849#
PRTIM    13-38     13-105#
PS        4-72      4-72#    11-74*    11-136*   11-;76*   17-20*    17-64*    20-138    20-139*   20-153*   32-253    32-254*   32-283*   32-291*
         32-410    32-411*   32-464*   32-474    32-523*   32-756*
PSEL      4-82#
PSU       4-293#
PSW       4-72#
PUNSAF   13-28     13-43#
PWRVEC    4-72#
QCNT     32-=04#   32-=63    32-=87*   32->02    32->04*   32->24    32->38*
QDRV0    32-=15    32-=26    32-=35    32-=47#
QDRV1    32-=16    32-=27    32-=36    32-=48#
QDRV2    32-=17    32-=28    32-=37    32-=49#
QDRV3    32-=18    32-=29    32-=38    32-=50#
QDRV4    32-=19    32-=30    32-=39    32-=51#
QDRV5    32-=20    32-=31    32-=40    32-=52#
QDRV6    32-=21    32-=32    32-=41    32-=53#
QDRV7    32-=22    32-=33    32-=42    32-=54#
QINPT    32-=15#   32-=89    32->06*   32->07*   32->08    32->10*
QOUTPT   32-=26#   32-=89*   32->27    32->40    32->41*   32->42    32->44*
QSTART   32-=35#   32-=69    32-=74    32->10    32->44
QSTOP    32-=36#   32->08    32->42
QTERM    32-=43    32-=55#
QUES     18-207    35-30#
QVCON     7-0#
QVSEK     7-0#
R6        4-72#    11-29     11-29*    11-29*
R7        4-72#
RANCYL   14-<33    14-<35#
RANPAT   14-<52    14-<54    14-<91#
RANSEC   14-;99#
RANSIZ   14-<49    14-<58#   14-<66    14-<84
RANTRK   14-<15#
RANWRT   14-;90    14-<98#
RANXIT   14-<87    14-<93#
RATIO     7-0#     14-;88    35-55
RAW       4-255#
RD.ADR   32-;66*   32-;67*   32-;69#   34-56
RD.RP    32-322    32-363    32-369    32-601    32-625    32-639    32-743    32-757    32-799    32-851    32-915    32-925    32-940    32-:73
         32-;22    32-;64#   32-<17    32-<31    32-<37    32-<70
RD.RP1   32-;68#   32-;96
RD.RP2   32-;64*   32-;95#
RD.RP3   32-;75    32-;87    32-;97#
RD.RP4   32-;80    32-;99#
RD.WRD   32-;70#   32-;71    34-56     34-57
RDCHR    21-1      31-1#
RDDAT     4-340#    8-0
RDHD      4-341#    8-0      14-944    14->23
RDLIN    11-115    17-25     18-100    18-136    18-183    31-1#     35-115    35-122    37-24     37-33
RDY       4-79#
READDR   12-193    14-96     14-258    14-293    14-=44    14-=97    14->49    15-182    16-11#
READHD   14-259    14-294    14-940#
READIN    4-330#
RECAL     4-325#   14-910
RECALO   14-910#   17-101
RECALT   14-330    14-909#
```

```
REDAPK   17-57     17-227#
REFMT    12-186#   14-154    14-214    14-240    14-285    14-325    14-361    14-437
REFMTX   12-187    12-189    12-191    12-206    12-211#
RELBUF   12-164    14-95     14-:71#
RELSE     4-327#
REPLZ    16-126    16-136    16-136    16-144    16-152    16-160    16-160    16-160    16-160    16-166    16-200    16-205    16-210    20-95#
RESREG   14-;54    17-60     20-30     29-1      30-1      31-1#     32-292    32-458    32-461    32-524    32-728    32-784    32-:54    32-<79
         32-=78
RESVEC    4-72#
RETRY     7-0#     14-66*    14-90*    14-121*   14-122    14-122    14-127*   14-168*   14-190*   14-195*   14-198*   14-199    14-199    14-232*
         14-277*   14-317*   14-353*   14-381*   14-450*   14-480*   14-497*   14-967*   14-979*   14-980    14-980    15-331
RMR       4-155#
RNOP      4-322#
RP04     14-891    14-911    14-927    14-945    14-;65    32-410#
RP04B    11-162    18-51     35-15#
RP05     11-165    18-54     35-16#
RP06     11-168    18-57     35-17#
RPADR    11-62*    32-200#   32-272    32-417    32-541    32-561    32-582    32-741    32-778    32-:71    32-;66    32-;72    32-;76    32-;90
         32-<23    32-<27    32-<59    32-<89    37-43
RPAS     32-225#   32-368*   32-849    32-964*   32-991*   32-:08*   32-;13
RPBA     32-220#
RPCA     32-232#   32-553    32-565    32-596
RPCC     32-233#   32-744
RPCS1    32-218#   32-314*   32-320    32-357    32-479    32-495    32-557    32-578    32-620    32-640    32-651    32-666    32-677    32-800
         32-852    32-903    32-932    32-969    32-;23    32-<18
RPCS2    32-222#   32-273*   32-313*   32-315    32-430*   32-542*   32-562*   32-583*   32-669    32-688    32-700    32-713    32-722*   32-742*
         32-798*   32-914*   32-994*   32-:07*   32-:72*   32-:88*   32-;73    32-<28    32-<60*   32-<66    32-<90*   32-<95
RPDA     32-221#   32-549    32-574    32-589
RPDB     32-227#   32-<64
RPDS1    32-223#   32-345    32-364    32-481    32-916    32-941    32-995    32-:74
RPDT     32-229#   32-323
RPEC1    32-236#
RPEC2    32-237#   32-<74
RPER1    14-260    14-295    32-224#   32-370    32-926    32-996    32-<32
RPER2    32-234#   32-997
RPER3    32-235#   32-998
RPERRS   32-20#    32-256    32-886    32-889    32-899    32-995*   32-996*   32-997*   32-998*   34-55     34-55     34-55     34-55
RPINIT   11-130    32-252#
RPLA     32-226#   32-758
RPMR     32-228#
RPOF     32-231#   32-361    32-602    32-606    32-626    32-630
RPSN     32-230#
RPTMR    16-233    32-:38#
RPVEC    11-63*    32-201#   32-269    32-271    32-411
RPWC     32-219#
RTC       4-329#   14-890
RTNCTR   14-889#
S        15-184    15-193    34-69#
SAVEFG   11-131*   32-151#   32-655    32-804    32-988
SAVER1    7-0#     14-592*   14-649*   14-675    14-683    14-687*
SAVER5    7-0#     14-593*   14-650*   14-684    14-688*
SAVREG   14-;28    17-19     20-20     29-1      30-1      31-1#     32-252    32-413    32-473    32-687    32-777    32-:41    32-<56    32-=62
SC       32-783    32-816    32-818    32-824    32-849#
SC1       4-220#
SC10      4-223#
SC11     32-888    32-922    32-979#
```

```
SC12     32-882   32-893   32-994#
SC13     32-872   32-875   32-:04#
SC2        4-221#
SC20       4-224#
SC3      32-863#  32-867
SC4      32-865#  32-896   32-907   32-909   32-957   32-971   32-978   32-993   32-:31
SC5      32-864   32-870#
SC6      32-892   32-910#
SC6A     32-890   32-934#
SC7      32-924   32-937   32-939   32-949   32-959#
SC8      32-904   32-917   32-927   32-933   32-942   32-970   32-972#  32-:03
SCMND    17-47    17-167#
SCOPE      4-72#
SDETAL   16-77    16-96    16-119#
SEARCH     4-332#
SECLMT     7-0#   18-71    18-152
SECOND     7-0#   11-78*   16-208   16-219*  16-220   16-222*
SEEK       4-324#
SEEKFG   32-159#  32-257   32-505
SELDRV     4-335#
SELPAR   12-84    12-179   14-;83#
SET.IE   32-286   32-317   32-448   32-522   32-727   32-857   32-<88#
SETFMT     4-334#
SETVEC   11-113   11-118   11-123   11-129#
SHDTYP   16-70    16-93    16-106#
SIXTEE     7-0#   11-75*   16-216*  16-218*
SIZMEM   11-86#
SKI        4-298#   4-308#
SKIER    14-51    14-462#
SLASH    18-50    18-182   35-63#
SPOTCK   14-74    14-175   14-219   14-263   14-303   14-344   14-366   14->41#
SRCHWT   32-104#  32-475*  32-580*  32-983
STACK      4-72#  11-29    11-73
START      5-1    11-14#   11-207
START1     5-2    11-18#   16-57
START2   11-16    11-20    11-25#
START3     5-6    11-22#
STATHD   16-110   35-18#
STATIN     7-0#   11-68*   12-172   12-174*  16-238*
STATIS   12-152   14-996#
STATPR   12-175   16-66#   17-187
STKLMT     4-72#
STNDAT     9-0#   14-751   14-759   14-;44   14-;52
STO      32-:48   32-:69#
STO1     32-:78#  32-;21   32-;26
STO2     32-:77   32-:79   32-:81   32-;13#
STO3     32-;15   32-;27#
STO5     32-:75   32-:94   32-;24   32-;33#
STO6     32-;17   32-;35#
STO7     32-;19   32-;43#
STO8     32-;42   32-;49#
STO9     32-;12   32-;30   32-;32   32-;34   32-;40   32-;47   32-;51#
SVRH11   32-657   32-674   32-705   32-806   32-821   32-881   32-929   32-947   32-990   32-:26   32-:87   32-;50   32-<56#
SWO        4-72#  12-186   14-;84   17-51
SWOO       4-72     4-72#
SWO1       4-72     4-72#  13-14    14-579
```

```
SW02        4-72       4-72#      11-134     12-170
SW03        4-72       4-72#
SW04        4-72       4-72#      14->12     18-236     19-23
SW05        4-72       4-72#      15-71      15-132
SW06        4-72       4-72#      14-=48
SW07        4-72       4-72#      14-710
SW08        4-72       4-72#
SW09        4-72       4-72#
SW1         4-72#
SW10        4-72#     15-6
SW11        4-72#
SW12        4-72#
SW13        4-72#     15-9       22-1
SW14        4-72#
SW15        4-72#     13-71      15-360     15-386
SW2         4-72#
SW3         4-72#     14-632     14-849
SW4         4-72#
SW5         4-72#     15-165
SW6         4-72#
SW7         4-72#     12-188
SW8         4-72#
SW9         4-72#
SWR         6-0#      11-29      11-29      11-29*     11-29*     11-34      11-58      11-134     12-170     12-186     12-188     13-14      13-71      14-579
           14-632     14-710     14-849     14-;84     14-=48     14->12     15-6       15-9       15-71      15-132     15-165     15-360     15-386     17-51
           18-236     19-23      20-151     21-1       21-1       21-1*      22-1       22-1       22-1       22-1       22-1
SWREG       5-1#      11-29      11-34      11-58      21-1       21-1       21-1
SWTIM      13-34      13-78#
SYSTAT     11-139     35-14#
T          15-181     15-189     34-67#
TABLE      18-74      35-86#
TABLE0     35-86      35-101#
TABLE1     35-89      35-101#
TABLE2     35-89      35-101#
TABLE3     35-89      35-101#
TABLE4     35-89      35-101#
TABLE5     35-89      35-101#
TABLE6     35-89      35-101#
TABLE7     35-89      35-101#
TAP         4-209#
TBITVE      4-72#
TD         32-781     32-790#
TDF         4-238#     4-256#
TIMER      32-164#    32-383*    32-517*    32-660*    32-706*    32-793*    32-911*    32-955*    32-:05*    32-:12*    32-:44     32-:46*    32-;04*    32-;31*
           32-;37*    32-;43*
TKVEC       4-72#     21-1*      21-1*
TPVEC       4-72#
TRAPVE      4-72#     11-29*     11-29*     11-41*
TRE         4-84#
TRFER      14-10      14-442#
TRK1        4-225#
TRK10       4-228#
TRK2        4-226#
TRK20       4-229#
TRK4        4-227#
TRKLMT      7-0#      18-69      18-146
```

```
TRNSWT  32-96#    32-539*   32-684*   32-694    32-711*   32-720*   32-795    32-796*   32-:82    32-:97    32-;10*
TRTVEC   4-72#
TSTPGM  11-15*    11-19*    11-22*    11-46*    11-47     32-343    32-398#
TUF      4-239#    4-257#
TYPDS   18-181    19-21     31-1#
TYPE    11-6      11-34     11-49     11-55     11-114    11-138    11-139    11-141    11-148    11-150    11-154    11-156    11-158    11-160
        11-161    11-169    11-171    11-175    11-190    11-193    12-28     12-48     12-50     13-43     13-44     13-45     13-47     13-59
        13-60     13-65     13-66     13-67     13-69     13-89     13-90     13-91     13-93     15-8      16-52     16-123    16-128    16-136
        16-136    16-144    16-152    16-160    16-160    16-160    16-160    16-168    16-197    16-202    16-207    17-15     17-24     17-59
        17-190    17-191    17-194    17-195    17-196    17-203    17-206    17-208    17-210    17-216    18-47     18-50     18-58     18-60
        18-93     18-99     18-119    18-125    18-162    18-176    18-182    18-190    18-207    18-208    18-210    18-212    18-225    18-226
        18-227    18-228    18-231    19-18     19-19     19-27     19-28     19-30     19-32     19-34     19-49     20-110    21-1      21-1
        21-1      21-1      21-1      21-1      21-1      21-1      21-1      21-1      21-1      21-1      21-1      21-1      21-1      21-1
        21-1      21-1      22-1      22-1      23-1      23-1      23-1      23-1      23-1      23-1      23-1      24-1      25-1      26-1
        31-1#     35-114    35-121    37-20     37-23     37-29     37-32
TYPEST  12-29     16-91#    17-175    19-35
TYPOC   11-8      21-1      23-1      23-1      31-1#     37-22     37-31
TYPON   31-1#
TYPOS   11-140    12-49     13-46     13-68     13-92     16-122    17-202    18-49     18-96     18-122    18-209    18-230    19-29     19-33
        31-1#
TYPRI4  16-107    16-109    20-138#
UCPAR   13-30     13-59#
ULDFLG  32-124#   32-312*   32-420    32-435*   32-617*   32-680*   32-712*   32-878    32-961    32-963*   32-979    32-981*   32-:90*   32-;03*
UNIT     7-0#     13-3*     13-46     13-68     13-92     15-66     19-22*    19-29     19-33
UNLOAD   4-323#
UNS      4-167#
UNSAF   14-5      14-492#
UNTASN  17-72     35-9#
UNTMSG  12-48     18-208    35-5#
UNTNOT  17-159    17-177    35-8#
UNTOFF  11-156    17-132    35-6#
UNTON   11-160    35-7#
UPE      4-111#
US1      4-98#
US2      4-99#
US4      4-100#
USE     11-49     35-3#
UWR      4-295#
VU30     4-245#
VUF      4-294#
VV       4-140#
WAIT     8-0#     12-20     12-65     12-109    12-120    12-139
WAO      4-304#
WC      32-808    32-829#
WCE      4-112#
WCF      4-158#
WCFER   14-28     14-473#
WCKD     4-336#    8-0
WCKER   14-45     14-78     14-159#
WCKHD    4-337#    8-0
WCSEL    7-0#     14-<59    14-<78    35-54
WCU      4-233#    4-251#
WLE      4-164#
WLEER   14-34     14-402#
WRCHK   31-6#     32-807    32-826
WRL      4-145#
```

```
WRT.AD   32-<13*   32-<23*   32-<26#   34-57
WRT.R1   32-<24#   32-<43
WRT.R2   32-<10*   32-<42#
WRT.R3   32-<19    32-<30    32-<33    32-<39    32-<45#
WRT.R4   32-<35    32-<47#
WRT.R5   32-<46    32-<48#
WRT.RP   32-356    32-360    32-494    32-548    32-552    32-556    32-564    32-573    32-577    32-588    32-595    32-605    32-619    32-629
         32-650    32-665    32-676    32-902    32-931    32-968    32-<10#
WRT.WD   32-<11*   32-<15    32-<22*   32-<25#   34-57
WRTDAT    4-338#    8-0      14->27
WRTHD     4-339#    8-0
WRTPK    12-75     14-=88#
WRTPK1   14-=90    14-=92    14-=96#
WRTPK2   14-=95    14->21#
WRTPK3   14-=94    14->20    14->25#
WRTPK4   14->24    14->30#
WRU       4-241#    4-259#
WSU       4-235#    4-253#
ZROIND    7-0#     14-591*   14-642*   14-648*   14-652
```

```
$$CMRE    5-556#
$$CMTM    5-556#
$$ESCA    4-72#
$$NEWT    4-72#
$$SET    31-1      31-1      31-1      31-1      31-1      31-1      31-1      31-1      31-1      31-1      31-1      31-1#     31-2
$$SKIP    4-72#
.$ACT1    4-47#     5-9
.$CATC    4-48#     5-1
.$CMTA    4-48#     5-556
.$DB2D    4-49#    29-1
.$DB2O    4-49#    30-1
.$ERRO    4-48#    22-1
.$ERRT    4-48#    23-1
.$RAND    4-50#    27-1
.$READ    4-47#    21-1
.$SAVE    4-49#    28-1
.$SIZE    4-49#    36-1
.$TRAP    4-49#    31-1
.$TYPD    4-48#    26-1
.$TYPE    4-47#    24-1
.$TYPO    4-48#    25-1
.EQUAT    4-47#     4-72
.HEADE    4-47#     4-54
.SETUP    4-47#     4-343
.SWRHI    4-47#     4-55
.SWRLO    4-47#     4-55#     4-56      4-57      4-58      4-60      4-62      4-67      4-69      4-70
A        16-172#   16-187    16-188    16-189    16-190    16-191
AUTOM    21-20#    22-1
CKCHR     4-8#     20-244    20-252    38-21     38-30
CKDIG     4-18#    18-140    18-146    18-152    18-187
CKNUM     4-31#    37-27     37-36
COMMEN    4-72#
ENDCOM    4-72#
ERENTR   21-13#    22-1
ERRCAL   32-1#     32-856    32-906    32-908    32-;81    32-<40
ERROR     4-72#    32-856    32-906    32-908    32-;81    32-<40    37-49
ESCAPE    4-72#
GETPRI    4-72#    36-1
GETSWR    4-50#     4-72#    11-34     11-34#    11-58
MORETA    5-16#     6-0
MULT      4-72#
NEWTST    4-72#
POP       4-72#    14-689    14->70    15-91     18-38     18-164    26-1      27-1      28-1
PUSH      4-72#    14-682    14->41    15-27     18-7      18-115    26-1      27-1      28-1
REPORT    4-72#
SETPRI    4-72#    21-1
SETTRA   31-1      31-1      31-1      31-1      31-1      31-1      31-1      31-1      31-1      31-1      31-1      31-1#     31-2
SETUP     4-72#    11-29
SKIP      4-72#
SLASH     4-72#
STARS     4-72#     5-9       6-0       6-0      21-1      21-1      21-1      21-1      21-1      22-1      23-1      24-1      25-1      26-1
         27-1      28-1      29-1      30-1      31-1      36-1
SWRSU     4-72#    11-29     11-29#
TRMTRP   31-1#     31-3
TYPBIN    4-72#
TYPDEC    4-72#
```

```
TYPNAM   4-72#    11-34
TYPNUM   4-72#
TYPOCS   4-72#    11-140   12-49    13-46    13-68    13-92    16-122   17-202   18-49    18-209   18-230   19-29    19-33
TYPOCT   4-72#    21-1     23-1     23-1
TYPTXT   4-72#    11-6
```