RM05/3/2 RM05/3/2 EXT'D DR TS AH-S074A-MC
CZRMVA0 FICHE 1 OF 2

RM05/3/2 RM05/3/2 EXT'D DR TS AH-S074A-MC
CZRMVA0 FICHE 2 OF 2

```
.REM @
```

IDENTIFICATION

PRODUCT CODE:	AC-S072A-MC
PRODUCT NAME:	CZRMVA0 RM05/3/2 EXT'D DR TST
DATE:	APRIL 1980
MAINTAINER:	CX DIAGNOSTIC ENGINEERING
AUTHOR:	MIKE LEAVITT

CONTENTS

1. ABSTRACT

THIS PROGRAM CONTAINS A SERIES OF TESTS THAT WILL VERIFY THAT THE DISK IS CAPABLE OF PERFORMING SEEKS, THAT THE ACCESS TIMES ARE WITHIN TOLERANCE, THAT THE TRACK AND SECTOR ADDRESSING CIRCUITRY OPERATES PROPERLY, AND THAT THE DATA STORAGE AND RETRIEVAL CAPABILITIES ARE FUNCTIONING.

2. REQUIREMENTS

2.1 EQUIPMENT

PDP-11 PROCESSOR
16K MEMORY
TELETYPE
PROGRAM LOADING DEVICE
KW11-L OR KW11-P (THE KW11-P IS REQUIRED FOR THE TIMING TESTS)
1 TO 8 DISK DRIVES (ANY COMBINATION OF RM05'S, RM03'S OR RM02'S)

2.2 PRELIMINARY PROGRAMS

RM05/3/2 DISKLESS CONTROLLER TEST, PART 1 & 2

RM05/3/2 FUNCTIONAL CONTROLLER TEST, PART 1, 2 & 3

2.3 MEDIA

THE PROGRAM REQUIRES THAT EACH DRIVE TO BE TESTED HAS A FORMATTED DISK PACK. THE PACK MAY BE FORMATTED IN EITHER 16-BIT OR 18-BIT MODE, DEPENDING ON THE TESTING REQUIREMENTS. NOTE THAT THE PROGRAM WILL NOT TEST A MIXTURE OF DRIVES WITH BOTH 16 AND 18 BIT MODE PACKS.

3. LOADING PROCEDURE

THE PROGRAM MAY BE LOADED FROM PAPER TAPE USING THE ABSOLUTE LOADER OR IT MAY BE LOADED FROM THE APPROPRIATE 'XXDP' MEDIA USING THE ASSOCAITED 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.

4. STARTING PROCEDURE

```
                    ----------------

          4.1       STARTING ADDRESSES

                    200       NORMAL STARTING ADDRESS
                    204       SELECT OPERATING PARAMETERS
                    210       SELECT RH11-RH70 ADDRESSES
                    214       COMBINATION OF 204 AND 210

                    NOTE:  STARTING ADDRESSES 210 AND 214 ARE AVAILABLE WHEN THE
                    PROGRAM IS INITIALLY STARTED; THESE STARTING ADDRESSES ARE
                    TREATED AS ADDRESSES 200 OR 204 RESPECTIVELY ON RESTARTS.

          4.2       OPERATOR ACTION

                    1.  LOAD PROGRAM INTO MEMORY (SEE SECTION 3.)
                    2.  LOAD A FORMATTED PACK INTO DRIVE(S) TO BE TESTED
                    3.  BRING DRIVE(S) TO ONLINE STATE, WRITE ENABLED,
                        AND LOCKED ON PORT.
                    4.  LOAD ADDRESS 200.
                    5.  SET SWITCHES (SEE SECTION 5.)
                    6.  PRESS START.
                    7.  THE PROGRAM WILL TYPEOUT THE STATUS OF THE DRIVES ATTACHED TO
                        THE SELECTED MASSBUS SUBSYSTEM.  TO INHIBIT THIS TYPEOUT, DO NOT
                        RESTART THE PROGRAM FROM ANY OF THE STARTING ADDRESSES; INSTEAD
                        TYPE A 'CONTROL C' ON THE KEYBOARD TO RETURN THE PROGRAM TO
                        COMMAND ENTRY MODE.

          4.3       PROGRAM ACTION

                      IN AN EFFORT TO ALLOW CONVERSATION WITH A PROGRAM FOR THE
                    PURPOSE OF CONTROLLING ITS OPERATION AND PARAMETERS THE FOLLOWING
                    CONSTRUCTIONS HAVE BEEN ADOPTED.

                    NOTE1:  IN ALL EXAMPLES BRACKETS ARE USED FOR CLARITY AND ARE
                            NOT TYPED BY THE USER.

                    NOTE2:  THE CARRRIAGE RETURN TYPED BY THE USER IS INDICATED BY
                            <CR> AND WILL BE ECHOED AS A 'CARRIAGE RETURN-LINE FEED'.

                    <.><CR>          PERIOD

                                     A STATEMENT TERMINATOR: WHEN TYPED AT THE
                                     END OF A LINE (LEGAL ON ALL LINES) IT TELLS
                                     THE PARAMETER STRING INTERPRETER (PSI) THIS IS
                                     THE END OF CHANGES TO THE CURRENT PARAMETER
                                     STRING.

                    <..><CR>         PERIOD PERIOD

                                     THE 'PERIOD PERIOD' TERMINATOR IS TYPED TO INDICATE
                                     THE END OF TEST PARAMETER MODIFICATION AND TO SIGNAL
                                     THE START OF TEST EXECUTION.

                    <,><CR>          COMMA

                                     THE COMMA IS USED AS A SEPARATOR BETWEEN DRIVE NUMBERS
```

```
                                        AND TEST NUMBERS.

                                <\/>    SLASH

                                        A MODIFICATION INDICATOR: IF A SLASH FOLLOWS A TEST
                                        NUMBER, THE PROGRAM WILL OPEN THAT TEST FOR PARAMETER
                                        MODIFICATION.

                                <^U>    CONTROL-U

                                        DELETE THE PRESENT INPUT STRING AND START A NEW
                                        LINE. TYPED BY DEPRESSING THE "CONTROL KEY"
                                        (CTRL) AND THEN STRIKING THE "U".

                                <\>     RUBOUT

                                        DELETE THE LAST CHARACTER FROM THE INPUT STRING.
                                        TYPED BY STRIKING THE "RUBOUT" KEY. WHICH WILL
                                        BE ECHOED BY A BACKSLASH (\) FOLLOWED BY THE
                                        CHARACTER DELETED.


                    4.3.1   CONTROL SWITCH SELECTION

                            STARTING THE PROGRAM AT ANY OF THE POSSIBLE STARTING ADDRESSES
                            WITH SW<07>=1 WILL RESULT IN ENTERING THE "CONTROL SWITCH
                            SETTING" MODE. THUS, ALLOWING THE OPERATOR TO SPECIFY THE
                            DESIRED STATE OF "C.SWR".

                            CONTROL SWITCH SELECTION EXAMPLES:

                            EXAMPLE #1

                              SET SW<07>=0
                              C.SWR=000000 / 400..

                            EXAMPLE #2

                              SET SW<07>=0
                              C.SWR=000000 / 220.
                              C.SWR=000000 / 220..

                    4.3.2   RH11 - RH70 ADDRESS SELECTION

                            STARTING THE PROGRAM AT 200 WILL RESULT IN AUTOMATIC
                            SELECT OF THE DEFAULT VALUES OF BUS ADDRESS (RMCS1),
                            VECTOR ADDRESS, AND PRIORITY LEVEL OF THE RH11-RH70.
                            IF THE DEFAULT VAULE OF THE BUS ADDRESS DOES NOT RESPOND
                            (TIMES OUT) WHEN ADDRESSED, AN ERROR IS REPORTED.
                            AFTER THE ERROR IS REPORTED ONE OF TWO COURSES OF ACTION
                            WILL BE TAKEN:

                            1. IF THERE IS A MONITOR -- RETURN TO THE MONITOR

                            2. IF THERE ISN'T A MONITOR -- ASK FOR NEW ADDRESSES
```

```
                        STARTING THE PROGRAM AT 210 OR 214 ALLOWS THE OPERATOR
                        TO CHANGE THE ADDRESS OF THE RH11 OR RH70 AND THE VECTOR
                        ADDRESS.

                        ADDRESS SELECTION EXAMPLES

                        EXAMPLE #1

                          RMCS1=176700 / 177200.

                        EXAMPLE #2

                          RMCS1=176700 / 176300<CR>
                          RMVEC=254 / 260<CR>
                          RMPRIO=5 / 6.

                        EXAMPLE #3

                          RMCS1=176700<CR>
                          RMVEC=254 / 260.

                        EXAMPLE #4

                          RH/RM FAILED TO RESPOND TO ADDRESSING
                          RMCS1   ERR PC
                          176300  XXXXXX
                          RMCS1=176300 / 176700.

                        EXAMPLE #5

                          RMCS1=176700 / 1776\67\6300<CR>
                          RMVEC=254<CR>
                          RMPRIO=5<CR>
                          RMCS1=176300.

                4.3.3   DRIVE AND PARAMETER SELECTION

                        STARTING THE PROGRAM AT 200 OR 210 WILL RESULT IN AUTOMATIC
                        SELECTION OF THE DRIVES TO TEST AND THE TESTS TO RUN.

                        STARTING THE PROGRAM AT 204 OR 214 ALLOWS THE OPERATOR
                        TO SELECT THE DRIVE(S) TO BE TESTED, THE TESTS TO BE EXECUTED,
                        AND THE PARAMETERS TO USE.

                        EACH TEST CONTAINS TWO SETS OF TRACK LIMIT PARAMETERS.  PARAMETERS
                        "LT"  "FT" AND "IT" ARE USED BY RM03/2 DRIVES AND PARAMETERS "LT'",
                        "FT'" AND "IT'" ARE USED BY RM05 DRIVES. THE PROGRAM DETERMINES WHICH
                        DRIVE IS BEING TESTED AND SELECTS THE CORRECT SET OF TRACK LIMIT
                        VALUES.  IF THE PROGRAM IS BEING USED TO TEST A SUBSYSTEM WHICH
                        CONTAINS BOTH RM03/2 AND RM05 DRIVES, THE OPERATOR MUST CHANGE
                        BOTH SETS OF TRACK LIMITS IF THE TESTS ARE TO BE MODIFIED FOR
                        ALL DRIVES TESTED.

                4.3.3.1 DRIVE AND PARAMETER SELECTION DESCRIPTION

                        THE FOLLOWING IS A TABLE OF TERMS USED BY THE PSI.
```

```
                        ''R''     REPEATS (ITERATIONS)
                        ''FC''    FIRST CYLINDER ADDRESS
                        ''LC''    LAST CYLINDER ADDRESS
                        ''IC''    INCREMENT CYLINDER
                        ''FT''    FIRST TRACK ADDRESS
                        ''LT''    LAST TRACK ADDRESS
                        ''IT''    INCREMENT TRACK
                        ''FT'''   FIRST TRACK ADDRESS      ;RM05 PARAMETER
                        ''LT'''   LAST TRACK ADDRESS       ;RM05 PARAMETER
                        ''IT'''   INCREMENT TRACK          ;RM05 PARAMETER
                        ''FS''    FIRST SECTOR ADDRESS
                        ''LS''    LAST SECTOR ADDRESS
                        ''PAT''   PATTERN (USED FOR DATA TEST)
                        ''WDX''   WORD OF PATTERN 0 WHERE X IS 1 TO 16

                       *''S''     ALL SEEK TESTS (TESTS 0 - 10)
                       *''T''     ALL TIMING TESTS (TESTS 12 - 15)
                       *''A''     ALL ADDRESS TESTS (TESTS 16 - 17)
                       *''D''     THE DATA TEST (TEST 20)
                       *''E''     THE EXERCISER (TEST 21)

                       * USED BY THE OPERATOR TO SELECT TEST GROUPS

NOTE:    ALL NUMBERS WILL BE IN DECIMAL EXCEPT FOR THE PATTERN
         (PAT) AND WORDS (WDX) SELECTION.  ''PAT'' WILL BE SELECTED BY
         A BIT (I.E. 001000(8)=PATTERN 9) AND ''WDX'' WILL BE IN OCTAL.

SPECIAL CASES OF CONTROL CHARACTERS
         IF <..> IS TYPED WHILE A TEST IS OPEN FOR MODIFICATION
         (</>) AND OTHER TESTS IN THE ''TEST COMMAND'' STRING
         ARE TO BE MODIFIED, THE REMAINING TESTS WILL BE UNCHANGED.


WHEN THE PROGRAM IS STARTED FROM LOCATION 200 OR 210, TESTS 0-10,
12-20 WILL BE RUN USING ALL AVAILABLE, ONLINE DRIVES.  IF THE OPERATOR
WISHES TO SELECT THE DRIVES TO BE TESTED, THE TESTS TO BE PERFORMED,
OR THE PARAMETERS TO BE USED, THE CONVERSATION MODE MAY BE ENTERED
BY TYPING A 'CONTROL C' OR BY STARTING THE PROGRAM FROM EITHER
LOCATION 204 OR 214.


THE PROGRAM WILL THEN RESPOND WITH:

        DRIVE(S)=

THE FOLLOWING EXAMPLES ASSUME THAT THE OPERATOR IS TO TEST
DRIVE #3 USING TESTS 2 THRU 7 AND TEST 11 AND DOES NOT DESIRE TO CHANGE
THE PARAMETERS (INITIAL CYLINDER ADDRESS, FINAL CYLINDER ADDRESS, ETC.).
THE USER WOULD TYPE '3<CR>' WHICH SAYS 'THIS IS THE END OF DRIVE ENTRY'.
THE PROGRAM WILL THEN REQUEST TEST NUMBERS.

THE TRANSACTION APPEARS AS FOLLOWS:

        DRIVE(S)=3<CR>
        TEST=
```

```
                                THE OPERATOR MAY NOW ENTER DESIRED TEST NUMBERS.  IN THE EXAMPLE,
                                HE WANTS TESTS 2 THRU 7 AND TEST 11 SO HE TYPES 2-7<,> (THE 'COMMA'
                                SEPARATES ENTRIES), 11<.><CR> ('PERIOD' 'CARRIAGE RETURN' -
                                END OF CHANGES, START TEST EXECUTION.)

                                IT NOW LOOKS LIKE THIS

                                        DRIVE(S)=3<CR>
                                        TEST=2-7,11.<CR>

                                IN THE NEXT EXAMPLE, IT IS ASSUMED THAT THE OPERATOR WISHES TO TEST
                                DRIVE 4 AND TO RUN TESTS 1 AND 3 THRU 11, MODIFYING THE PARAMETERS
                                FOR TESTS 3 AND 10.

                                THE TRANSACTION WOULD BE AS FOLLOWS:

                                        DRIVE(S)=4<CR>
                                        TEST=

                                THE OPERATOR NOW ENTERS THE TEST NUMBERS.  THE TRANSACTION IS
                                GIVEN BELOW:

                                        DRIVE(S)=4<CR>
                                        TEST=1,3/4-7,10/11<CR>

                                NOTICE THIS SAYS SELECT TEST 1, CONTINUE<,>; SELECT TEST 3, OPEN</>;
                                SELECT TESTS 4-7, CONTINUE<,>; SELECT TEST 10, OPEN</>; SELECT TEST
                                11, END OF INPUT <.>.

                                THE PROGRAM SCANS THE TEST NUMBER INPUT AND DETERMINES THAT THE
                                PARAMETERS FOR TEST 3 AND TEST 10 ARE TO BE CHANGED.  THE OTHER
                                TESTS WILL NOT BE ALTERED.

                                (THE ENTIRE TRANSACTION IS REPEATED FOR CLARITY)

                                        DRIVE(S)=4<CR>
                                        TEST=1,3/4-7,10/11<CR>
                                          TEST 3
                                            R=X /                ;WHERE X IS ITERATION

                                THE NEW VALUE FOR 'R' MAY BE ENTERED. TERMINATING THE ENTRY WITH
                                A <.> (PERIOD) WILL TERMINATE THE CHANGES FOR THIS TEST; TYPING A
                                <CR> OR TERMINATING THE ENTRY WITH A <CR> WILL CAUSE THE PROGRAM
                                TO MOVE TO THE NEXT PARAMETER.

                                        DRIVE(S)=4<CR>
                                        TEST=1,3/4-7,10/11<CR>
                                          TEST 3
                                            R=1 / <CR>           ;DO NOT ALTER-BUT CONTINUE
                                            FC=N /               ;WHERE 'N' IS FIRST CYLINDER ADDRESS

                                IF THE OPERATOR DOES NOT WISH TO CHANGE 'FC', THE FOLLOWING OCCURS:

                                        DRIVE(S)=4<CR>
                                        TEST=1,3/4-7,10/11.<CR>
                                          TEST 3
                                            R=1 / <CR>           ;DO NOT ALTER THIS LINE BUT CONTINUE
```

```
                                        FC=0 / <CR>          ;DO NOT ALTER THIS LINE BUT CONTINUE
                                        LC=822 /

                                THE PROGRAM RESPONDS WITH THE PREVIOUSLY ASSIGNED PARAMETER FOR LAST
                                CYLINDER ADDRESS IN THIS CASE USING 822 AS THE EXAMPLE.  THIS IS
                                WHAT THE OPERATOR INTENDED TO MODIFY AND IS WHY TEST 3 WAS
                                OPENED.  TO CHANGE THE VALUE TO '20', THE NEW VALUE IS TYPED
                                FOLLOWED BY A 'PERIOD' TERMINATOR (<.><CR>).

                                THE TOTAL TRANSACTION AND RESPONSE:

                                        DRIVE(S)=4<CR>
                                        TEST=1,3/4-7,10/11<CR>
                                          TEST 3
                                            R=1 / <CR>
                                            FC=0 / <CR>
                                            LC= 822 / 20.<CR>
                                          TEST 10
                                            R=1 /

                                THE PROGRAM HAS LOADED TEST 3 WITH ITS NEW PARAMETERS AND THE
                                PROGRAM IS WAITING FOR CHANGES TO TEST 10'S PARAMETERS.

                                        DRIVE(S)=4<CR>
                                        TEST=1,3/4-7,10/11<CR>
                                          TEST 3
                                            R=1 / <CR>
                                            FC=0 / <CR>
                                            LC= 822 / 20.<CR>
                                          TEST 10
                                            R=1 / 10.<CR>

                                THE OPERATOR TYPES THE NEW VALUE (10) AND TERMINATES THE ENTRY
                                WITH A 'PERIOD' 'CARRIAGE RETURN'.

                                THE PROGRAM NOW LOADS TEST 10 WITH THE NEW PARAMETERS (TEST 11 RETAINS
                                THE PREVIOUSLY ASSIGNED PARAMETERS) AND RESPONDS WITH:

                                        DRIVE(S)=

                                SINCE THE USER DID NOT END THE CONVERSATION MODE WITH A 'PERIOD PERIOD',
                                THE PROGRAM HAS LOOPED BACK TO THE BEGINNING LOOKING FOR MORE
                                CHANGES. THAT IS TO SAY, AFTER THE ENTRY FOR DRIVE SELECTION, A
                                <,><CR> WILL CAUSE THE TEST MESSAGE TO BE REPEATED AND FURTHER
                                CHANGES CAN BE MADE. HOWEVER, AT SOME POINT IN ORDER TO EXECUTE
                                THE PROGRAM, A 'PERIOD PERIOD' MUST BE TYPED.

                                IF A SINGLE 'PERIOD' IS TYPED WHILE DRIVE OR TEST NUMBERS ARE BEING
                                ENTERED, THE PROGRAM WILL START EXECUTION IMMEDIATELY.  A 'PERIOD
                                PERIOD' MUST BE TYPED BEFORE THE PROGRAM WILL EXIT TEST PARAMETER
                                CHANGE MODE TO GO TO EXECUTION.

                        4.3.3.2 DRIVE AND PARAMETER SELECTION EXAMPLES


                                EXAMPLE #1
                                ----------
```

```
                                DRIVE=4.<CR>            ;SELECT DRIVE #4, TERMINATE AND
                                                        ;BEGIN EXECUTION USING PREVIOUSLY ASSIGNED
                                                        ;PARAMETERS

                                EXAMPLE #2
                                ----------

                                DRIVE=0<CR>             ;SELECT DRIVE #0 AND MAKE 'CHANGES '',''
                                TEST=1-5.<CR>           ;RUN TEST 1 THRU 5 ONLY, USE DEFAULT
                                                        ;PARAMETERS AND TERMINATE AND EXECUTE.''

                                EXAMPLE #3
                                ----------

                                DRIVE=2<CR>             ;SELECT DRIVE #2 AND MAKE CHANGES '',''
                                TEST=1-5,6/7/10/<CR>    ;RUN TEST 1-5 WITH DEFAULT PARAMETERS, OPEN
                                  TEST 6                ;TEST 6,7 AND 10 FOR CHANGES
                                    R=1 / <CR>          ;LEAVE 'R' AS IS AND MOVE TO NEXT PARAMETER
                                    FC=0 / 10.<CR>      ;SET 'FC' CYLINDER ADDRESS TO 10, END CHANGES
                                                        ;TO TEST 6.

                                  TEST 7
                                    R=1 / 50<CR>        ;50 ITERATIONS, MOVE TO NEXT PARAMETER
                                    FC=0 / <CR>         ;DO NOT CHANGE 'FC' CYLINDER ADDRESS BUT CONTINUE
                                    LC=822 / 50..<CR>   ;TEST 10 IS STILL PENDING AND WILL BE
                                                        ;RETAIN ITS PRESENT PARAMETERS.

                                EXAMPLE #4
                                ----------

                                DRIVE=0<CR>             ;SELECT DRIVE #0 AND MAKE CHANGES
                                TEST=S,E.<CR>           ;RUN ALL SEEK TESTS AND THE EXERCISER

                                EXAMPLE #5
                                ----------

                                DRIVE=1<CR>
                                TEST=S/D<CR>            ;RUN ALL SEEK TESTS (OPEN FOR CHANGES) AND
                                  TEST 0                ;THE DATA TEST (WITH DEFAULT PARAMETERS).
                                    R=10 / <CR>         ;RUN WITH 10 ITERATIONS
                                    FC=0 / 10..<CR>     ;CHANGE FIRST CYLINDER ADDRESS
                                                        ;AND START EXECUTION
                                                        ;TESTS 1 - 10 WILL RETAIN THEIR PREVIOUSLY
                                                        ;ASSIGNED PARAMETERS.

                                EXAMPLE #6
                                ----------

                                DRIVE=1<CR>
                                TEST=S/<CR>             ;OPEN THE SEEK TESTS (TESTS 0-10)
                                  TEST 0
                                    R=10 / 100.<CR>     ;CHANGE TO 100 ITERATIONS, TO TO THE NEXT TEST
                                  TEST 1
                                    R=100 / 1000.<CR>   ;CHANGE 'R' TO 1000 ITERATIONS, MOVE TO NEXT TEST
                                  TEST 2
```

```
                                    R=1 / 10<CR>          ;CHANGE 'R' TO 10 ITERATIONS, GO TO NEXT PARAMETER
                                    FC=0 / 50<CR>         ;CHANGE 'FC' TO 50, GO TO NEXT PARAMETER
                                    LC=822 / 51.<CR>      ;CHANGE 'LC' TO 51, GO TO THE NEXT TEST
                                  TEST 3
                                    R=1.<CR>              ;MOVE TO NEXT TEST
                                  TEST 4
                                    R=1..<CR>             ;USE TEST 4'S PARAMETERS AND START PROGRAM EXECUTION

                                EXAMPLE #7
                                ----------

                                DRIVE=1<CR>
                                TEST=D/<CR>               ;SELECT AND OPEN THE DATA TEST
                                  TEST 20
                                    .=1 / 1000<CR>        ;DO 1000 ITERATION OF TEST PATTERN
                                    FC=0 / 10<CR>         ;#8 ON CYLINDER 10, TRACK 2, SECTOR 4
                                    LC=822 / 10<CR>
                                    IC=64 / 0<CR>
                                    FT=0 / 2<CR>
                                    LT=4 / 2<CR>
                                    IT=1 / <CR>
                                    FT'=0 / 2<CR>         ;RM05 PARAMETER
                                    LT'=18 / 2<CR>        ;RM05 PARAMETER
                                    IT'=1 / <CR>          ;RM05 PARAMETER
                                    FS=0 / 4<CR>
                                    LS=22 / 4<CR>
                                    PAT=177777 / 400..<CR> ;RUN WITH PATTERN #8

                                EXAMPLE #8
                                ----------

                                DRIVE=1<CR>               ;USE THE SAME PARAMETERS AS IN EXAMPLE
                                TEST=D/<CR>               ;#7, BUT ALSO SPECIFY A DATA PATTERN (PAT #0).
                                  TEST 20
                                    R=1000 / <CR>
                                    FC=10 / <CR>
                                    LC=10 / <CR>
                                    IC=0 / <CR>
                                    FT=2 / <CR>
                                    LT=2 / <CR>
                                    IT=1 / <CR>
                                    FT'=0 / 2<CR>         ;RM05 PARAMETER
                                    LT'=18 / 2<CR>        ;RM05 PARAMETER
                                    IT'=1 / <CR>          ;RM05 PARAMETER
                                    FS=4 / <CR>
                                    LS=4 / <CR>
                                    PAT=000400 / 401<CR> ;RUN WITH PATTERNS #8 & #0 (0=OPERATOR INPUT)
                                      WD1=165555 / 125252<CR> ;FIRST WORD OF PATTERN 0
                                      WD2=133333 / 52525..<CR> ;SECOND WORD OF PATTERN 0
                                                          ;<..> START EXECUTION

                                EXAMPLE #9
                                ----------

                                DRIVE=0,1,4<CR>           ;TEST DRIVES 0,1, AND 4 IN SEQUENCE
                                TEST=0-5/<CR>             ;CHANGE TEST 5
                                  TEST 0
```

```
                                R=10 / <CR>
                                FC=0 / <CR>
                                LC=822 / 1..<CR>    ;CHANGE LAST CYLINDER FROM 822 TO 1
                                                    ;START PROGRAM EXECUTION.
```

5. SWITCH SETTINGS

5.1 OPERATIONAL SWITCH SETTINGS

WITH SW<15:0>=0 THE PROGRAM WILL PRINT OUT ON
ERRORS AND CONTINUE IN TEST.
THE SWITCH SETTINGS ARE:

SW<15>=1...HALT ON ERROR
SW<14>=1...LOOP ON TEST
SW<13>=1...INHIBIT ERROR TYPEOUTS
SW<12>=1...TYPE TEST NUMBER
SW<11>=1...INHIBIT ITERATIONS
SW<10>=1...RING BELL ON ERROR
SW<09>=1...LOOP ON ERROR
SW<08>=1...PRINT ERROR MESSAGE ON LINE PRINTER
SW<07>=1...READ 'C.SWR' SETTINGS FROM TTY
SW<06>=1...INHIBIT TIME REPORTS (TESTS 12-15)
SW<05>=1...REPORT ONE ERROR PER SECTOR (TESTS 16 & 17)
SW<04>=1...INHIBIT WRITES (TEST 20)
SW<03>=1...INHIBIT WRITE CHECKS (TEST 20)
SW<02>=1...INHIBIT READ AND SOFTWARE COMPARES (TEST 20)
SW<01>=1...INHIBIT SOFTWARE COMPARES (TEST 20)
SW<00>=1...PERFORM READ AFTER WRITE CHECK ERROR (TEST 20)

IF THE PROGRAM IS BEING RUN ON A SWITCHLESS PROCESSOR (I.E. AN 11/34)
THE PROGRAM WILL DETERMINE THAT THE HARDWARE SWITCH REGISTER IS
NOT PRESENT AND WILL USE A 'SOFTWARE' SWITCH REGISTER. THE
'SOFTWARE' SWITCH REGISTER IS LOCATED AT LOCATION 176 (8). THE
SETTINGS OF THE 'SOFTWARE' SWITCHES ARE CONTROLLED THROUGH A KEYBOARD
ROUTINE WHICH IS CALLED BY TYPING A 'CONTROL G'. THE PROGRAM WILL
RECOGNIZE THE 'CONTROL G' AT ANY TIME EXCEPT WHEN THE PROGRAM
IS AT A HIGHER PRIORITY PROCESSING AN RM05/3/2 INTERRUPT. THE
'SOFTWARE' SWITCH VALUES ARE ENTERED AS AN OCTAL NUMBER IN RESPONSE
TO THE PROMPT FROM THE SWITCH ENTRY ROUTINE:

        'SWR = NNNNNN   NEW ='

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

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

```
                                5.2     CONTROL SWITCH SETTINGS

                                        THE CONTROL SWITCH SETTINGS ARE ENTERED THROUGH THE KEYBOARD.

                                        TO ENTER THE CONTROL SWITCH SETTING MODE PLACE SW<07>=1
                                        BEFORE PRESSING START. THEN UPON STARTING THE PROGRAM IT
                                        WILL TYPE THE PRESENT CONTENTS OF THE CONTROL SWITCH REGISTER
                                        (C.SWR) AND WAIT FOR THE NEW SETTING TO BE INPUT. THE INPUT
                                        STRING MUST CONSIST OF 1 TO 6 OCTAL DIGITS, TWO PERIODS (..),
                                        AND A CARRIAGE RETURN.

                                        THE C.SWR SETTINGS ARE:

                                        C.SWR<15>=0...WRITE PACK BEFORE TESTING (TEST16)
                                                 =1...INHIBIT WRITE PACK BEFORE TESTING (TEST16)
                                        C.SWR<14>=0...NO STALL BETWEEN DRIVE FUNCTIONS
                                                 =1...STALL AFTER EVERY DRIVE FUNCTION
                                        C.SWR<13>=0...USE SPECIFIC STALL TIMES
                                                 =1...USE RANDOM STALL TIMES
                                        C.SWR<12>=0...NO INCREMENTING STALLS IN TEST 4
                                                 =1...PERFORM INCREMENTING STALLS IN TEST 4
                                        C.SWR<08>=0...DO IMPLIED SEEKS WITH DATA TRANSFERS
                                                 =1...DO EXPLICIT SEEKS BEFORE DATA TRANSFERS
                                        C.SWR<07>=0...DO READ HEADER AND  DATA COMMANDS IN TESTS 0-7
                                                 =1...DO EXPLICIT SEEK COMMANDS IN TESTS 0-7
                                        C.SWR<06>=0...60 HZ POWER SOURCE
                                                 =1...50 HZ POWER SOURCE
                                        C.SWR<05>=0...ALLOW SOFTWARE TIMEOUTS(ENABLE WATCHDOG TIMER)
                                                 =1...INHIBIT SOFTWARE TIMEOUTS(DISABLE WATCHDOG TIMER)
                                        C.SWR<00>=0...OPERATE IN 32. SECTOR (16 BIT) MODE
                                                 =1...OPERATE IN 30. SECTOR (18 BIT) MODE

                                        THE DEFAULT CONDITION OF C.SWR<15:00>=0.

                                        REFER TO 4.3.1 FOR C.SWR SELECTION



                        6.      ERRORS
                                ------

                                        THERE ARE ANUMBER OF ERRORS THAT CAN OCCUR IN THIS PROGRAM.
                                        WHEN AN ERROR IS ENCOUNTERED, THE CALL TO THE ERROR ROUTINE
                                        IS MADE AND IF SW<13> IS NOT SET, AN ERROR MESSAGE PERTAINING
                                        TO THE ERROR WILL BE TYPED.  EACH ERROR TYPEOUT WILL CONTAIN
                                        THE FOLLOWING:

                                        1.  AN ERROR MESSAGE
                                        2.  A DATA HEADER
                                        3.  A DATA STRING

                                        REFER TO THE FOLLOWING SECTION FOR THE DIFFERENT ERRORS
                                        THAT CAN OCCUR.

                        6.1     ERROR TYPES
```

THE ERRORS THAT OCCUR IN THIS PROGRAM FALL INTO THREE
(3) CATEGORIES DEFINED AND EXPLAINED AS FOLLOWS:

6.1.1 DRIVER ERROR

THESE ERRORS WILL BE DETECTED BY THE RH/RM DRIVER.
THERE ARE TWO CLASSES OF DRIVER ERRORS; THOSE THAT
CAN NOT BE IDENTIFIED IN A MANNER THAT ALLOWS THE
INFORMATION TO BE RETURNED TO A "DATA PARAMETER BLOCK"
(DPB) AND THOSE THAT CAN. THE FIRST CLASS WILL BE
REPORTED BY ERROR CALLS (EMT'S) 1-5 WITHIN THE DRIVER.
THE SECOND CLASS WILL PASS THE ERROR CODES TO THE
STATUS/ERROR WORD (DPB+16) OF THE PROPER DPB.

6.1.2 NON-FATAL ERRORS

THESE ERRORS WILL BE DUE TO "DISK" OR "DATA" FAILURES
WHICH WILL BE REPORTED AS THEY OCCUR. AFTER REPORTING
THE ERROR THE PROGRAM WILL CONTINUE TESTING.

6.1.3 FATAL ERRORS

THIS TYPE OF ERROR WILL BE THE RESULT OF ANY KIND
OF ERROR THAT INHIBITS THE PROGRAM FROM TESTING THE DISK.

THIS ERROR WILL BE REPORTED WHEN IT OCCURS, THEN THE PROGRAM
WILL ABORT THE TEST AND GO TO THE END OF PROGRAM.

6.2 ERROR RECOVERY

6.2.1 PRETEST ERROR

WHEN THIS TYPE OF ERROR OCCURS IT WILL BE REPORTED. THEN DEPENDING
ON HOW THE PROGRAM WAS STARTED IT WILL ASK FOR THE DRIVES AND
ADDRESSES FOR TESTING OR RETURN TO MONITOR.

6.2.2 NON-FATAL ERROR

WHEN THIS TYPE OF ERROR OCCURS IT WILL BE REPORTED AND
THE PROGRAM WILL CONTINUE IN TEST.

6.2.3 FATAL ERROR

WHEN THIS TYPE OF ERROR OCCURS IT WILL BE REPORTED. THE
PROGRAM WILL ABORT THE TEST AND GO TO THE END OF PROGRAM.

7. RESTRICTIONS

THE PROGRAM WILL TEST THE DRIVES IN EITHER 16 BIT MODE OR IN 18
BIT MODE DEPENDING ON THE SETTING OF 'S.SWR<00>'. IF 'C.SWR<00>'
IS 0, ALL OF THE DRIVES WILL BE TESTED IN 16 BIT MODE; IF 'C.SWR<00>
IS 1, ALL OF THE DRIVES WILL BE TESTED IN 18 BIT MODE. THE PROGRAM

HAS NO PROVISIONS FOR TESTING DRIVES WITH INTERMIXED PACKS OR TESTING
BOTH 16 BIT MODE AND 18 BIT MODE DRIVES ON THE SAME SYSTEM.
ACT11 AUTOMATIC MODE ASSUMES 16 BIT MODE.

BEFORE THE PROGRAM IS STARTED, PROPERLY FORMATTED PACKS MUST BE MOUNTED
ON THE DRIVES WHICH WILL BE TESTED. THE PROGRAM ASSUMES A
PROPERLY FORMATTED PACK. THE FORMAT OF THE PACK IS NOT ALTERED
BY THE PROGRAM.

8. MISCELLANEOUS

8.1 EXECUTION TIME

THE PROGRAM REQUIRES APPROXIMATELY 15 MINUTES TO MAKE ONE PASS WITH
RM03/2 DRIVES AND APPROXIMATELY 16.5 MINUTES TO A PASS WITH RM05
DRIVES. THIS ASSUMES THE DEFAULT TEST SEQUENCE (TESTS 0-10, 12-20)
AND DEFAULT TEST PARAMETERS.

8.2 STACK POINTER

THE STACK POINTER IS INITIALLY SET TO 1100.

8.3 TIMING TESTS (TESTS 12-15) PRINTOUTS

AT THE COMPLETION OF EACH OF THE TIMING TESTS THE TIME OF THE
MINIMUM SEEK, MAXIMUM SEEK, AND THE AVERAGE OF ALL OF THE
SEEKS PERFORMED ARE TYPED ON THE TTY. THE NUMBER OF SEEKS THAT
HAD TIMES BELOW THE MINIMUM TIME ALLOWED WILL BE TYPED ON
THE SAME LINE AS THE MINIMUM TIME. THE NUMBER ABOVE THE
MAXIMUM WILL BE TYPED ON THE SAME LINE AS THE MAXIMUM
TIME, AND THE TOTAL NUMBER OF SEEKS PERFORMED WILL BE ON THE
SAME LINE AS THE AVERAGE.

NOTE: THE PROGRAM STALLS FOR 2 MILLISECONDS BETWEEN SEEK ORDERS,
       THIS STALL TIME IS NOT INCLUDED IN THE CALCULATED SEEK TIMES.
       THE SEEK TIMES SPECIFIED ARE POSITIONER MOVEMENT TIMES ONLY
       AND ARE NOT A MEASUREMENT OF EFFECTIVE SEEK TIME.

8.3.1 TIMING TOLERANCES

1. TEST 12 -- ROTATIONAL SPEED TIMES

   --TIMES FOR RM05/3 DRIVES--

   60 HZ
   MINIMUM=16340 US
   MAXIMUM=17000 US
   NOMINAL=16670 US

   50 HZ
   MINIMUM=16250 US
   MAXIMUM=17090 US
   NOMINAL=16670 US

```
                                --TIMES FOR RM02 DRIVES--

                                60 HZ
                                MINIMUM=24500 US
                                MAXIMUM=25500 US
                                NOMINAL=25000 US

                                50 HZ
                                MINIMUM=24370 US
                                MAXIMUM=25630 US
                                NOMINAL=25000 US

                             2. TEST 13 -- ONE CYLINDER SEEK TIMES

                                MAXIMUM=7000 US
                                NOMINAL=6000 US

                             3. TEST 14 -- ACCESS TIME MEASUREMENT

                                MAXIMUM=32000 US
                                NOMINAL=30000 US

                             4. TEST 15 -- MAXIMUM SEEK TIMES

                                MAXIMUM=57000 US
                                NOMINAL=55000 US

                       8.3.2.  TIMING TESTS PRINTOUT EXAMPLES

                             EXAMPLE #1
                             ----------

                             ROTATIONAL SPEED TIMES
                             MIN=16670 US
                             MAX=16690 US
                             AVG=16680 US  10 SEEKS TIMED

                             ONE CYLINDER SEEK TIMES
                              * FORWARD
                             MIN=5350 US
                             MAX=6920 US
                             AVG=5550 US  821 SEEKS TIMED
                              * REVERSE
                             MIN=5140 US
                             MAX=5960 US
                             AVG=5430 US  822 SEEKS TIMED

                             ACCESS TIME MEASUREMENTS
                              * FORWARD
                             MIN=27770 US
                             MAX=28640 US
                             AVG=28230 US  128 SEEKS TIMED
                              * REVERSE
                             MIN=27990 US
                             MAX=28550 US
```

```
                                AVG=28220 US  128 SEEKS TIMED

                                MAXIMUM SEEK TIMES
                                 * FORWARD
                                MIN=49990 US
                                MAX=51980 US
                                AVG=51010 US  128 SEEKS TIMED
                                 * REVERSE
                                MIN=48120 US
                                MAX=50650 US
                                AVG=49340 US  128 SEEKS TIMED


                                EXAMPLE #2
                                ----------

                                ROTATIONAL SPEED TIMES
                                MIN=16670 US
                                MAX=16690 US
                                AVG=16680 US  10 SEEKS TIMED

                                ONE CYLINDER SEEK TIMES
                                 * FORWARD
                                MIN=5470 US
                                MAX=7940 US  3 ABOVE THE MAXIMUM OF 7000 US
                                AVG=5830 US  821 SEEKS TIMED
                                 * REVERSE
                                MIN=5040 US
                                MAX=5970 US
                                AVG=5330 US 822 SEEKS TIMED

                                ACCESS TIME MEASURMENTS
                                 * FORWARD
                                MIN=29730 US
                                MAX=32620 US  73 ABOVE THE MAXIMUM OF 32000 US
                                AVG=30320 US  128 SEEKS TIMED
                                 * REVERSE
                                MIN=28620
                                MAX=32230 US  128 ABOVE THE MAXIMUM OF 32000 US
                                AVG=32800 US  128 SEEKS TIMED

                                MAXIMUM SEEK TIMES
                                 * FORWARD
                                MIN=57510 US
                                MAX=57240 US  128 ABOVE THE MAXIMUM OF 57000 US
                                AVG=57020 US  128 SEEKS TIMED
                                 * REVERSE
                                MIN=57050 US
                                MAX=57550 US  128 ABOVE THE MAXIMUM OF 57000 US
                                AVG=57210 US  128 SEEKS TIMED


                        8.4     END OF TEST

                                WITH ALL SWITCHES ON A "0" AN "END OF PASS" MESSAGE WILL BE
                                TYPED AT THE COMPLETION OF TESTING A DRIVE AND THE "END OF TEST"
                                TYPEOUT WILL OCCUR WHEN ALL DRIVES HAVE BEEN TESTED.
```

9. PROGRAM DESCRIPTION

THIS PROGRAM CONTAINS NINTEEN TESTS NUMBERED 0-22 IN OCTAL.
TESTS 0-7 & 11 WILL READ THE CYLINDER, TRACK, AND SECTOR INFORMATION
FROM THE HEADER, USING A "READ HEADER AND DATA" COMMAND, AND
THEN CHECK THE INFORMATION FOR VALIDITY. THUS, INSURING
THE SEEK OPERATION FUNCTIONS PROPERLY. TESTS 12-15 WILL MEASURE
THE ROTATIONAL SPEED, THE ONE CYLINDER SEEK, THE ACCESS TIME,
AND THE MAXIMUM SEEK TIMES TO ENSURE THEY ARE ALL WITHIN THE
TOLERANCES ALLOWED. TEST 16 AND 17 ENSURES THE SECTOR AND
TRACK ADDRESSING CIRCUITRY WORKS PROPERLY. TEST 20 VERIFIES
THE DATA STORAGE AND RETRIEVAL CAPABILITIES ARE FUNCTIONAL.
AND TEST 21 WILL STRESS AND CHECK THE READ/WRITE AND SERVO
SYSTEMS.

THE PROGRAM WILL START BY IDENTIFING ITSELF AND DETERMINING ALL
DRIVES THAT ARE AVAILABLE FOR TESTING. THEN BEGINNING WITH
THE LOWEST NUMBERICAL DRIVE AND PROCEEDING IN SEQUENTIAL ORDER.
ALL OF THE DRIVES WILL BE TESTED. ONE PASS THROUGH THE TEST
SEQUENCE (TESTS 0-10,12-20) WILL BE PERFORMED ON EACH DRIVE BEFORE
MOVING TO THE NEXT DRIVE IN SEQUENCE. DRIVE TO BE
TESTED WILL BE TYPED AT THE BEGINNING OF EACH PASS, AN "END
OF PASS" MESSAGE WILL BE TYPED AT THE COMPLETION OF EACH PASS,
AND AN "END OF TEST" MESSAGE WILL BE TYPED AFTER TESTING ALL
DRIVES.

REFER TO THE FOLLOWING SECTIONS FOR DETAILED DESCRIPTIONS
OF EACH TEST.

9.1 TEST 0 - RECAL/SEEK TEST

THIS TEST WILL CAUSE THE DRIVE TO EXECUTE A RECALIBRATE
COMMAND CYCLE AND THEN SEEK FORWARD TO CYLINDER "LC". AT
THE COMPLETION OF BOTH COMMANDS, STATUS INDICATIONS ARE
CHECKED TO ENSURE NO ERRORS OCCURRED.

THE PARAMETERS USED BY THE TEST ARE GIVEN BELOW:

R	-	200
LC	-	822
FT	-	0
FT'	-	0
FS	-	0

9.2 TEST 1 - SEEK/SEEK TEST

THIS TEST WILL CAUSE THE DRIVE TO EXECUTE A FORWARD SEEK
CYCLE TO "LC", "LT", "LS" FOLLOWED BY A REVERSE SEEK CYCLE TO
"FC", "FT", "FS". AT THE COMPLETION OF EACH SEEK, THE PROPER
INDICATORS ARE EXAMINED TO ENSURE PROPER OPERATION.

THE PARAMETERS USED BY THE TEST ARE GIVEN BELOW:

R	-	100
FC	-	0
LC	-	256
IC	-	0
FT	-	0
LT	-	0
FT'	-	0
LT'	-	0
FS	-	0
LS	-	0

9.3 TEST 2 - INCREMENTAL SEEK TEST

THIS TEST WILL COMMAND FORWARD SEEK CYCLES TO ADVANCE THE CYLINDER ADDRESS FROM "FC" TO "LC" BY THE INCREMENT "IC". WHEN THE RESULTANT CYLINDER ADDRESS (NC) EXCEEDS "LC" REVERSE SEEK CYCLES ARE INITIATED; STARTING AT THE LAST LEGAL "NC" AND DECREMENTING BY "IC" UNTIL "NC" IS LESS THAN "FC". AT THE COMPLETION OF EACH SEEK COMMAND THE PROPER INDICATORS ARE EXAMINED TO ENSURE PROPER OPERATION.

THE PARAMETERS USED BY THE TEST ARE GIVEN BELOW:

R	-	1
FC	-	0
LC	-	822
IC	-	1
FT	-	0
FT'	-	0
FS	-	0

9.4 TEST 3 - STEPPING SEEK TEST

THIS TEST WILL COMMAND SEEK CYCLES TO CYLINDER 0,1,2,4, 8,16,32,64,128,256 AND 512. AT THE COMPLETION OF EACH SEEK COMMAND THE PROPER INDICATORS ARE EXAMINED TO VERIFY PROPER OPERATION.

THE PARAMETERS USED BY THE TEST ARE GIVEN BELOW:

R	-	8
FC	-	0
LC	-	512
IC	-	1
FT	-	0
FT'	-	0
FS	-	0

9.5 TEST 4 - OSCILLATING SEEK TEST

THIS TEST WILL COMMAND SEEK CYCLES FROM "FC" TO "NC" AND BACK TO "FC". "NC" STARTS AT "FC" AND INCREMENTS BY "IC" UP TO CYLINDER "LC", THEN IS DECREMENTED BY "IC" BACK TO CYLINDER "FC". AT THE COMPLETION OF EVERY SEEK COMMAND THE PROPER INDICATORS ARE

EXAMINED TO ENSURE PROPER OPERATION.

THE FOLLOWING PARAMETERS ARE USED BY THE TEST:

R	-	1
FC	-	0
LC	-	822
IC	-	1
FT	-	0
FT'	-	0
FS	-	0

9.6 TEST 5 - CONVERGING/DIVERGING SEEK TEST

THIS TEST WILL CAUSE THE DRIVE TO EXECUTE FORWARD AND REVERSE SEEKS FROM "NC1" AND "NC2" RESPECTIVELY, "NC1" WILL BE INCREMENTED BY "IC" AND "NC2" WILL BE DECREMENTED BY "IC" UNTIL "NC1" IS GREATER THAN THE INITIAL VALUE OF "NC2" AND "NC2" IS LESS THAN THE INITIAL VALUE OF "NC1". AT THE COMPLETION OF EACH SEEK COMMAND THE PROPER INDICATORS ARE EXAMINED TO ENSURE PROPER OPERATION. "NC1" AND "NC2" DEFAULT TO "FC" AND "LC" RESPECTIVELY.
THE FOLLOWING PARAMETERS ARE USED BY THE TEST:

R	-	1
FC	-	0
LC	-	822
IC	-	1
FT	-	0
FT'	-	0
FS	-	0

9.7 TEST 6 - SERVO ADDRESSING LOGIC NOISE GENERATOR TEST

IN THIS TEST A SEEK IS DONE TO CYL "NC" THEN A SEEK TO NC+4 THEN NC+1 THEN NC+3 THEN NC+2 THEN NC+5. NOW "NC" IS UPDATED BY "IC" AND THE ABOVE SEQUENCE IS REPEATED UNITL "LC" IS EXCEEDED BY ANY OF THE ABOVE VALUES. THE INITIAL VALUE OF "NC" IS "FC". AT THE COMPLETION OF EACH SEEK COMMAND THE PROPER INDICATORS ARE EXAMINED TO ENSURE PROPER OPERATION.

THE FOLLOWING PARAMETERS ARE USED BY THE TEST:

R	-	1
FC	-	0
LC	-	822
IC	-	1
FT	-	0
FT'	-	0
FS	-	0

9.8 TEST 7 - RANDOM SEEK TEST

THIS TEST PERFORMS RANDOM SEEK OPERATIONS BETWEEN CYLINDERS 'FC' 'LC'. AFTER EACH SEEK, THE POSITION OF THE DRIVE IS VERIFIED BY READING A SECTOR FROM THE CURRENTLY ADDRESSED CYLINDER AND TRACK. THE TRACK ADDRESS IS INCREMENTED FOR EACH SEEK SO THAT VERIFICATION

OF POSITIONING OCCURS USING EACH HEAD. TRACK ADDRESSES ARE INCREMENTED BETWEEN PARAMTERS 'FT' AND 'LT'.

THE FOLLOWING PARAMETERS ARE USED BY THE TEST:

R	-	5000
FC	-	0
LC	-	822
FT	-	0
LT	-	4
FT'	-	0
LT'	-	18

9.9 TEST 10 - SERVO SETTLE DOWN TEST

THIS TEST VERIFIES THAT THE SERVO HAS SETTLED DOWN AND THAT THE DRIVE IS ON CYLINDER WHEN THE DRIVE INDICATES SEEK COMPLETE. RANDOM SEEKS ARE ISSUED BETWEEN CYLINDERS 'NC1' AND 'NC1+IC' ('NC1' STARTS AT VALUE 'FC'). AT THE COMPLETION OF 1000 (10) SEEKS, 'NC1' IS INCREMENTED BY VALUE 'IC' AND THE SEQUENCE IS REPEATED. THE TEST IS COMPLETED WHEN 'NC1' HAS BEEN INCREMENTED BEYOND 'LC'.

WHEN THE SEEK COMPLETES, THE PROGRAM READS THE DRIVE'S LOOK-AHEAD REGISTER (RMLA) TO DETERMINE THE ADDRESS OF THE SECTOR ROTATING INTO POSITION. THE PROGRAM THEN ISSUES A WRITE HEADER AND DATA COMMAND FOR THAT SECTOR. ERRORS IN THIS TEST INDICATE THAT THE SERVO SYSTEM MAY NOT BE ADJUSTED CORRECTLY, THAT THE DRIVE IS MALFUNCTIONING, OR THAT A PACK WITH MARGINAL SERVO TRACKS IS MOUNTED ON THE DRIVE.

THIS TEST IS VALID ONLY IF THE OPERATION IS STARTED WITHIN A FEW HUNDRED MICRO-SECONDS AFTER SEEK DONE OCCURS. THE NECESSARY TIME DEPENDENT PARAMETERS OCCUR WITHIN THE REQUIRED TIME RANGE FREQUENTLY ENOUGH TO PERMIT THIS TEST TO BE EFFECTIVE.

THE FOLLOWING PARAMETERS ARE USED BY THE TEST:

R	-	1
FC	-	0
LC	-	822
IC	-	100
FT	-	0
FT'	-	0

9.10 TEST 11 - ALL SEEKS TEST

THIS TEST VERIFIES THAT THE DISK DRIVE CAN SEEK FROM EACH CYLINDER TO ALL OTHER CYLINDERS.

BEGINNING WITH CYLINDER 'FC', THE TEST SEEKS TO EACH CYLINDER BETWEEN 'FC' AND 'LC' FROM CYINDER 'FC'. THE BEGINNING CYLINDER ADDRESS IS INCREMENTED AND THE TEST SEEKS BETWEEN THE NEW CYLINDER ADDRESS AND ALL CYLINDERS BETWEEN 'FC' AND 'LC'. THE SEQUENCE CONTINUES UNTIL ALL CYLINDERS HAVE BEEN CHECKED.

THE FOLLOWING PARAMETERS ARE USED BY THIS TEST:

R	-	1

```
1084                                        FC      -       0
1085                                        LC      -     822
1086                                        IC      -       1
1087                                        FT      -       0
1088                                        FT'     -       0
1089                                        FS      -       0
1090
1091                    9.11    TEST 12 - ROTATIONAL SPEED TIMING TEST
1092
1093                            THIS TEST WILL START A SEARCH TO CYLINDER 'FC', TRACK 'FT',
1094                            SECTOR 'FS'. AS SOON AS THE INTERRUPT OCCURS, THE GO BIT IS
1095                            SET AGAIN AND THE OPERATION IS TIMED. THIS PROCEDURE IS
1096                            REPEATED 10 TIMES THEN THE AVERAGE TIME IS CALCULATED AND
1097                            CHECKED TO ENSURE IT IS WITHIN TOLERANCE:
1098
1099                            RM05/3:
1100                                    16.67 MS/REV + OR - 2% IF 60HZ
1101                                    16.67 MS/REV + OR - 2.5% IF 50HZ.
1102
1103                            RM02:
1104                                    25.00 MS/REV + OR - 2% IF 60HZ
1105                                    25.00 MS/REV + OR - 2.5% IF 50HZ.
1106
1107                            THE FOLLOWING PARAMETERS ARE USED BY THE TEST:
1108
1109                                    R       -       1
1110                                    FC      -       0
1111                                    FT      -       0
1112                                    FT'     -       0
1113                                    FS      -       0
1114
1115                    9.12    TEST 13 - ONE CYLINDER SEEK TIMING TEST
1116
1117                            THIS TEST WILL COMMAND FORWARD SEEK CYCLES TO ADVANCE THE
1118                            CYLINDER BY ONE UNTIL THE INCREMENT IS GREATER THAN THE
1119                            CYLINDER 'LC', THEN REVERSE SEEK TO CYLINDER 'FC'.  THE
1120                            TIME TO PERFORM EACH SEEK IS CHECKED TO ENSURE IT DOES NOT
1121                            EXCEED THE MAXIMUM TIME PERMITTED FOR A ONE CYLINDER SEEK.
1122                            THE TIME MUST BE LESS THAN 10MS.
1123
1124                            THE TEST USES THE FOLLOWING PARAMETERS:
1125
1126                                    R       -       1
1127                                    FC      -       0
1128                                    LC      -     822
1129
1130                    9.13    TEST 14 - ACCESS TIME MEASUREMENT
1131
1132                            THIS TEST WILL COMMAND A FORWARD SEEK FROM CYLINDER 'FC' TO
1133                            CYLINDER 'LC', THEN A REVERSEK FROM CYLINDER 'LC' TO
1134                            CYLINDER 'FC'.  BOTH SEEKS ARE TIMED AND CHECKED TO ENSURE THEY
1135                            ARE WITHIN THE TOLERANCE ALLOWED FOR THE ACCESS TIME MEASURMENT.
1136                            THIS SEQUENCE IS REPEATED 128 TIMES (FOR A TOTAL
1137                            OF 256 SEEKS).  THE AVERAGE ACCESS TIME MUST BE LESS THAN 30 MS.
1138
1139                            THE TEST USES THE FOLLOWING PARAMETERS:
1140
```

```
                                        R       -       1
                                        FC      -       0
                                        LC      -       255

                9.14    TEST 15 - MAXIMUM SEEK TIMING TEST

                        THIS TEST WILL COMMAND A FORWARD SEEK FROM CYLINDER 'FC' TO
                        CYLINDER 'LC', THEN A REVERSE SEEK FROM CYLINDER 'LC' TO
                        CYLINDER 'FC'.  BOTH SEEKS ARE TIMED AND CHECKED TO ENSURE
                        THEY ARE WITHIN THE TOLERANCE ALLOWED FOR THE MAXIMUM SEEK
                        TIME.  THIS SEQUENCE IS REPEATED 128 TIMES (FOR
                        A TOTAL OF 256 SEEKS).  THE MAXIMUM SEEK TIME MUST BE LESS THAN
                        54 MS.

                        THE TEST USES THE FOLLOWING PARAMETERS:

                                        R       -       1
                                        FC      -       0
                                        LC      -       822

                9.15    TEST 16 - SECTOR ADDRESSING TEST

                        THIS TEST WRITES DATA INTO ALL SECTORS OF TRACK "FT".  THE
                        DATA WILL BE 256 WORDS OF THE SECTOR ADDRESS OF THE SECTOR
                        BEING WRITTEN.  A WRITE CHECK IS PERFORMED, THE BUFFER IS
                        CLEARED (TO 177400) AND THE DATA IS READ AND COMPARED.  THEN SECTOR 0
                        IS REWRITTEN AND SECTORS 0 - 31 ARE WRITE CHECKED.  THEN
                        SECTOR 1 IS REWRITTEN AND SECTORS 0 - 31 ARE WRITE CHECKED.
                        THIS REWRITE AND WRITE CHECK PROCEDURE IS CONTINUED UP THROUGH
                        REWRITE SECTOR 31 AND WRITE CHECK SECTORS 0-31.

                        THE TEST USES THE FOLLOWING PARAMETERS:

                                        R       -       1
                                        FC      -       0
                                        FT      -       0
                                        FT'     -       0

                9.16    TEST 17 - TRACK ADDRESSING TEST

                        THIS TEST WILL WRITE DATA IN THE FORM OF TRACK ADDRESSES
                        IN CYLINDER "FC" SECTOR "FS" OF EVERY TRACK WITH EACH TRACK
                        GETTING ITS OWN TRACK ADDRESS.
                        A WRITE CHECK IS THEN PERFORMED ON EACH TRACK TO INSURE
                        THE DATA IS VALID.  THEN TRACK 0 IS REWRITTEN AND TRACK 1
                        THROUGH LAST TRACK IS WRITE CHECKED.  THEN TRACK 1 IS
                        REWRITTEN AND TRACK 2 THROUGH LAST TRACK IS WRITE CHECKED.
                        THIS PROCEDURE IS CONTINUED UP THROUGH REWRITING NEXT TO
                        LAST TARCK AND WRITE CHECKING LAST TRACK.

                        THE TEST USES THE FOLLOWING PARAMETERS:

                                        R       -       1
                                        FC      -       0
                                        FS      -       0

                9.17    TEST 20 - DATA TEST
```

```
1198
1199                    THIS TEST PERFORMS DATA STORAGE AND RETRIEVAL ON CYLINDERS
1200                    "FC" THROUGH "LC" BY THE INCREMENT "IC" USING THE DATA PATTERNS
1201                    SPECIFIED. THE FOLLOWING SEQUENCE OCCURS FOR EACH CYLINDER:
1202                    1.      SET "NT" TO "FT" THEN REPEAT 2-4 UNTIL "NT" > "LT"
1203                    2.      WRITE THEN WRITE CHECK "FS" THROUGH "LS" OF TRACK "NT"
1204                    3.      READ THEN SOFTWARE COMPARE "FS" THROUGH "LS" OF TRACK "NT"
1205                    4.      INCREMENT "NT" BY "IT"
1206                    5.      REPEAT STEPS 1-4 FOR EACH DATA PATTERN
1207                    6.      REPEAT STEPS 1-5 FOR "FC" THROUGH "LC" ADVANCING BY "IC"
1208
1209                    IF A WRITE CHECK ERROR OCCURS THE ERROR IS REPORTED AND
1210                    THE TRACK IN ERROR IS REWRITTEN AND CHECKED. THIS CHECK IS
1211                    ACCOMPLISHED BY PERFORMING TWO(2) SUCCESSIVE ERROR FREE
1212                    WRITE CHECKS. IF THE CHECK FAILS THE ERROR IS REPORTED AS
1213                    FATAL AND NO READ OCCURS.
1214                    FS DEFAULTS TO 1 AND LS DEFAULTS TO 0
1215                    PAT DEFAULTS TO 177777 (ALL POSSIBLE PATTERNS)
1216                    THE POSSIBLE PATTERNS ARE:
1217
1218                    PAT 0   PAT 1   PAT 2   PAT 3   PAT 4   PAT 5   PAT 6   PAT 7
1219                    ------  ------  ------  ------  ------  ------  ------  ------
1220                    165555  000001  177776  000000  000000  052525  007417  026455
1221                    133333  000003  177774  000000  010421  052525  007417  026455
1222                    165555  000007  177770  000000  021042  052525  007417  026455
1223                    133333  000017  177760  177777  031463  125252  170360  151322
1224                    165555  000037  177740  177777  042104  125252  170360  151322
1225                    133333  000077  177700  177777  052525  125252  170360  151322
1226                    165555  000177  177600  000000  063146  052525  007417  026455
1227                    133333  000377  177400  000000  073567  052525  007417  026455
1228                    165555  000777  177000  177777  104210  125252  170360  151322
1229                    133333  001777  176000  177777  114631  125252  170360  151322
1230                    165555  003777  174000  000000  125252  052525  007417  026455
1231                    133333  007777  170000  177777  135673  125252  170360  151322
1232                    165555  017777  160000  000000  146314  052525  007417  026455
1233                    133333  037777  140000  177777  156735  125252  170360  151322
1234                    165555  077777  100000  000000  167356  052525  007417  026455
1235                    133333  177777  000000  177777  177777  125252  170360  151322
1236
1237                    PAT 8   PAT 9   PAT 10  PAT 11  PAT 12  PAT 13  PAT 14  PAT 15
1238                    ------  ------  ------  ------  ------  ------  ------  ------
1239                    165555  000001  177776  172666  077777  153333  000000  177777
1240                    133333  000002  177775  155555  137777  066667  177777  000000
1241                    165555  000004  177773  172666  157777  153333  177777  000000
1242                    133333  000010  177767  155555  167777  066667  177777  000000
1243                    165555  000020  177757  172666  173777  153333  177777  000000
1244                    133333  000040  177737  155555  175777  066667  177777  000000
1245                    165555  000100  177677  172666  176777  153333  177777  000000
1246                    133333  000200  177577  155555  177377  066667  177777  000000
1247                    165555  000400  177377  172666  177577  153333  177777  000000
1248                    133333  001000  176777  155555  177677  066667  177777  000000
1249                    165555  002000  175777  172666  177737  153333  177777  000000
1250                    133333  004000  173777  155555  177757  066667  177777  000000
1251                    165555  010000  167777  172666  177767  153333  177777  000000
1252                    133333  020000  157777  155555  177773  066667  177777  000000
1253                    165555  040000  137777  172666  177775  153333  177777  000000
1254                    133333  100000  077777  155555  177776  066667  177777  000000
```

THE TEST USES THE FOLLOWING PARAMETERS:

R	-	1
FC	-	0
LC	-	822
IC	-	64
FT	-	0
LT	-	4
IT	-	1
FT'	-	0
LT'	-	18
IT'	-	1
FS	-	1
LS	-	0
PAT	-	177777

9.18 TEST 21 - RANDOM ADDRESS AND RANDOM PATTERN TEST

STARTING AT "FC" AND GOING THROUGH "LC" THE DISK PACK
IS WRITTEN WITH A RANDOM PATTERN. THE FIRST TWO WORDS
OF EACH SECTOR WILL BE THE BASE OF THE RANDOM GENERATOR
FOR THAT SECTOR.
THE TEST THEN PERFORMS THE FOLLOWING SEQUENCE "R" TIMES
"R" DEFAULTS TO 20,000.

1) GENERATE A RANDOM ADDRESS
2) WRITE A RANDOM PATTERN AT THE ADDRESS
   GENERATED IN 1.
3) GENERATE A RANDOM ADDRESS
4) READ THE SECTOR AT THE ADDRESS
   GENERATED IN 3.
5) DO A SOFTWARE CHECK OF THE DATA READ IN 4.
6) DO A WRITE CHECK OF THE DATA WRITTEN IN 2
7) GENERATE A RANDOM ADDRRESS
8) READ THE SECTOR AT THE ADDRESS
   GENERATED IN 7.
9) DO A SOFTWARE CHECK OF THE DATA READ IN 8
10) DO A WRITE CHECK OF THE DATA WRITTEN IN 2

THE TEST USES THE FOLLOWING PARAMETERS:

R	-	20000
FC	-	0
LC	-	822

9.19 TEST 22 - ACCESS TIME ADJUSTMENT TEST

THIS TEST PERFORMS SEEKS BETWEEN CYLINDERS 0 & 255 TO ALLOW THE
OPERATOR TO ADJUST THE ACCESS TIME ON AN RM05/3/2 USING THE
DDU. THE PROGRAM STALLS APPROXIMATELY 5 SECONDS BETWEEN SEEKS
SO THAT THE ACCESS TIME INDICATORS ON THE DDU MAY BE OBSERVED.

THE TEST USES THE FOLLOWING PARAMETERS:

R	-	5000
FC	-	0

```
1312                                        LC      -       255
1313
1314
1315
1316                            10.     PROGRAM LISTING
1317                                    ---------------
1318                            @
```

```
                        ;PROGRAM REVISION #001

                        .TITLE  CZRMVA0 RM05/3/2 EXT'D DR TST
                        ;*COPYRIGHT (C) 1980
                        ;*DIGITAL EQUIPMENT CORP.
                        ;*MAYNARD, MASS. 01754
                        ;*
                        ;*PROGRAM BY MIKE LEAVITT
                        ;*
                        ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                        ;*PACKAGE (MAINDEC-11-DZQAC-C4), 1980.
                        ;*
                        .SBTTL  OPERATIONAL SWITCH SETTINGS
                        ;*
                        ;*      SWITCH                  USE
                        ;*      ------          --------------------
                        ;*        15            HALT ON ERROR
                        ;*        14            LOOP ON TEST
                        ;*        13            INHIBIT ERROR TYPEOUTS
                        ;*        12            TYPE TEST NUMBER
                        ;*        10            BELL ON ERROR
                        ;*         9            LOOP ON ERROR
                        .SBTTL  OPERATIONAL SWITCH SETTINGS
                        ;*
                        ;*      SWITCH                  USE
                        ;*      ------          --------------------
                        ;*        15            HALT ON ERROR
                        ;*        14            LOOP ON TEST
                        ;*        13            INHIBIT ERROR TYPEOUTS
                        ;*        10            BELL ON ERROR
                        ;*         9            LOOP ON ERROR
                        ;*         8            PRINT ERROR MESSAGE ON LINE PRINTER
                        ;*         7            READ 'C.SWR' SETTINGS FROM TTY
                        ;*         6            INHIBIT TIME REPORTS (TESTS 12-15)
                        ;*         5            REPORT ONE ERROR PER SECTOR (TESTS 16 & 17)
                        ;*         4            INHIBIT WRITES (TEST 20)
                        ;*         3            INHIBIT WRITE CHECKS (TEST 20)
                        ;*         2            INHIBIT READ AND SOFTWARE COMPARES (TEST 20)
                        ;*         1            INHIBIT SOFTWARE COMPARES (TEST 20)
                        ;*         0            PERFORM READ AFTER WRITE CHECK ERROR (TEST 20)

                        .SBTTL  CONTROL SWITCH SETTINGS
                        ;*
                        ;*    SWITCH  STATE             USE
                        ;*    ------  -----     ------------------------------------
                        ;*      15      0       WRITE PACK BEFORE TESTING (TEST 21)
                        ;*              1       INHIBIT WRITING PACK BEFORE TESTING (TEST 21)
                        ;*      14      0       NO STALL BETWEEN DRIVE FUNCTIONS
                        ;*              1       STALL AFTER EVERY DRIVE FUNCTION
                        ;*      13      0       USE SPECIFIC STALL TIME
                        ;*              1       USE RANDOM STALL
                        ;*      12      0       NO INCREMENTING STALL IN TEST 4
                        ;*              1       DO INCREMENTING STALL IN TEST 4
                        ;*       8      0       DO IMPLIED SEEKS WITH DATA TRANSFERS
                        ;*              1       DO EXPLICIT SEEKS BEFORE DATA TRANSFERS
                        ;*       7      0       DO "READ HEADER AND DATA" IN TESTS 0-11
                        ;*              1       DO EXPLICIT SEEKS IN TESTS 0-11
```

```
                        ;*       6      0       60 HZ
                        ;*              1       50 HZ
                        ;*       5      0       RUN WATCHDOG TIMER
                        ;*              1       INHIBIT WATCHDOG TIMER
                        ;*       0      0       TEST DRIVE(S) IN 32. SECTOR (16 BIT) MODE
                        ;*              1       TEST DRIVE(S) IN 30. SECTOR (18 BIT) 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
000004                  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/RH70 REGISTERS

                        ;CONTROL AND STATUS REGISTER 1 (RMCS1)

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

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

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

                        ;CONTROL AND STATUS REGISTER 2 (RMCS2)

000001                  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)
                        ;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 (RMDB)
                        ;(EACH BIT IS CALLED BY BIT NUMBER)

                        .SBTTL  RM 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 (RMDS) (#01)

000001          OM      = 1                     ;OFFSET MODE
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 (RMER1) (#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 #01 (RMMR1)(#03) - READ ONLY BITS

000001          DMD     = 1                     ;DIAGNOSTIC MODE
000002          LSIT    = 2
                ;LS     = 4
000010          WD      = 10
000020          EECC    = 20
000040          WC      = 40
000100          CONT    = 100
000200          PHA     = 200
000400          PDA     = 400
001000          ECRC    = 1000
002000          PLFS    = 2000
004000          ESRC    = 4000
010000          REX     = 10000
020000          EBL     = 20000
                ;R/G    = 40000
100000          OCC     = 100000

                        ;MAINTAINABILITY REGISTER #01 (RMMR1) (#03) - WRITE ONLY BITS

000001          DMD     = 1                     ;DIAGNOSTIC MODE BIT
000002          MSC     = 2
000004          MI      = 4
```

```
000010          MWP     = 10
000020          DTG     = 20
000040          MS      = 40
000100          MDF     = 100
000200          MSER    = 200
000400          MOC     = 400
001000          MUR     = 1000
002000          MRD     = 2000
004000          MCLK    = 4000
010000          MSEN    = 10000
020000          DTO     = 20000
040000          OBEN    = 40000
100000          OBCK    = 100000

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

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

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

                ;DRIVE TYPE REGISTER (RMDT) (#06)

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

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


000100          SC0     = 100           ;SECTOR COUNT FIELD 0 (BIT #6)
000200          SC1     = 200           ;SECTOR COUNT FIELD 1 (BIT #7)
000400          SC2     = 400           ;SECTOR COUNT FIELD 2 (BIT #8)
                ;SC3    = 1000          ;SECTOR COUNT FIELD 3 (BIT #9)
                ;SC4    = 2000          ;SECTOR COUNT FIELD 4 (BIT #10)


                ;RM MAINTAINABILITY REGISTER #2 (RMMR2) (#10)

                ;RM ERROR REGISTER #02 (RMMR2) (#10)
```

```
                                                    H 3
                        ;OFFSET REGISTER (RMOF) (#11)

000200                  OFD     = 200                   ;OFFSET DIRECTION (BIT #07)
002000                  HCI     = 2000                  ;HEADER COMPARE INHIBIT (BIT #10)
004000                  ECI     = 4000                  ;ERROR CORRECTION CODE INHIBIT (BIT #11)
010000                  FMT16   = 10000                 ;FORMAT BIT (BIT #12)

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

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

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


                        ;RM ERROR REGISTER #02 (RMER2) (#15)

020000                  OPE     = 20000                 ;OPERATOR PLUG ERROR (BIT #13)
040000                  SKI     = 40000                 ;SEEK INCOMPLETE (BIT #15)

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

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

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

                        ;OP CODE DEFINITIONS
000101                  NOOP    = 101
000103                  UNLOAD  = 103
000105                  SEEK    = 105
000107                  RECAL   = 107
000111                  DRVCLR  = 111
000113                  RELEASE = 113
000115                  OFFSET  = 115
000117                  RTC     = 117
000121                  READIN  = 121
000123                  PACK    = 123
000131                  SEARCH  = 131
000151                  WRCKD   = 151
000153                  WRCKHD  = 153
000161                  WRITE   = 161
000163                  WRTHD   = 163
000171                  READ    = 171
000173                  READHD  = 173
000141                  GETREG  = 141
000143                  SETFORM = 143
000145                  SELDRV  = 145

                        ;OTHER EQUATES

177400                  SCTRWC  = -256.                 ;WORD COUNT FOR SECTOR
```

```
                                .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  004570              JMP     @#START1               ;;JUMP TO STARTING ADDRESS OF PROGRAM
309
310 000204  000137  004612              JMP     @#START2       ;SELECT OPERATING PARAMETERS
311
312 000210  000137  004560              JMP     @#START3       ;SELECT RH/RM ADDRESSES
313
314 000214  000137  004602              JMP     @#START4       ;COMBINATION OF 204 AND 210
315
316                             .SBTTL  ACT11 HOOKS

                                ;;**********************************************************************
                                ;HOOKS REQUIRED BY ACT11
            000220                      $SVPC=.                ;SAVE PC
            000046                      .=46
    000046  021244                      $ENDAD                 ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
            000052                      .=52
    000052  020000                      .WORD   20000          ;;2)SET LOC.52 TO 20000
            000220                      .=$SVPC                ;; RESTORE PC
317
318         001100              .=1100
319                             .SBTTL  APT PARAMETER BLOCK

                                ;;**********************************************************************
                                ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
                                ;;**********************************************************************
            001100                      .$X=.   ;;SAVE CURRENT LOCATION
            000024                      .=24    ;;SET POWER FAIL TO POINT TO START OF PROGRAM
    000024  000200                      200     ;;FOR APT START UP
            000044                      .=44    ;;POINT TO APT INDIRECT ADDRESS PNTR.
    000044  001100                      $APTHDR ;;POINT TO APT HEADER BLOCK
            001100                      .=.$X   ;;RESET LOCATION COUNTER
                                ;;**********************************************************************
                                ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
                                ;INTERFACE SPEC.

    001100                      $APTHD:
    001100  000000              $HIBTS: .WORD   0       ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
    001102  001234              $MBADR: .WORD   $MAIL   ;;ADDRESS OF APT MAILBOX (BITS 0-15)
    001104  001604              $TSTM:  .WORD   900.    ;;RUN TIM OF LONGEST TEST
    001106  001604              $PASTM: .WORD   900.    ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
    001110  001604              $UNITM: .WORD   900.    ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDED UNIT
    001112  000030                      .WORD   $ETEND-$MAIL/2  ;;LENGTH MAILBOX-ETABLE(WORDS)
320         001114              TAB.XY=.        ;COMMON TAG STARTING ADDRESS
321
```

```
    520
    524
```

```
      0                         .SBTTL  COMMON TAGS

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

            001114                      .=TAB.XY
      001114                    $CMTAG:                     ;;START OF COMMON TAGS
      001114  000000                    .WORD   0
      001116     000            $TSTNM: .BYTE   0           ;;CONTAINS THE TEST NUMBER
      001117     000            $ERFLG: .BYTE   0           ;;CONTAINS ERROR FLAG
      001120  000000            $ICNT:  .WORD   0           ;;CONTAINS SUBTEST ITERATION COUNT
      001122  000000            $LPADR: .WORD   0           ;;CONTAINS SCOPE LOOP ADDRESS
      001124  000000            $LPERR: .WORD   0           ;;CONTAINS SCOPE RETURN FOR ERRORS
      001126  000000            $ERTTL: .WORD   0           ;;CONTAINS TOTAL ERRORS DETECTED
      001130     000            $ITEMB: .BYTE   0           ;;CONTAINS ITEM CONTROL BYTE
      001131     001            $ERMAX: .BYTE   1           ;;CONTAINS MAX. ERRORS PER TEST
      001132  000000            $ERRPC: .WORD   0           ;;CONTAINS PC OF LAST ERROR INSTRUCTION
      001134  000000            $GDADR: .WORD   0           ;;CONTAINS ADDRESS OF 'GOOD' DATA
      001136  000000            $BDADR: .WORD   0           ;;CONTAINS ADDRESS OF 'BAD' DATA
      001140  000000            $GDDAT: .WORD   0           ;;CONTAINS 'GOOD' DATA
      001142  000000            $BDDAT: .WORD   0           ;;CONTAINS 'BAD' DATA
      001144  000000                    .WORD   0           ;;RESERVED--NOT TO BE USED
      001146  000000                    .WORD   0
      001150     000            $AUTOB: .BYTE   0           ;;AUTOMATIC MODE INDICATOR
      001151     000            $INTAG: .BYTE   0           ;;INTERRUPT MODE INDICATOR
      001152  000000                    .WORD   0
      001154  177570            SWR:    .WORD   DSWR        ;;ADDRESS OF SWITCH REGISTER
      001156  177570            DISPLAY: .WORD  DDISP       ;;ADDRESS OF DISPLAY REGISTER
      001160  177560            $TKS:   177560              ;;TTY KBD STATUS
      001162  177562            $TKB:   177562              ;;TTY KBD BUFFER
      001164  177564            $TPS:   177564              ;;TTY PRINTER STATUS REG. ADDRESS
      001166  177566            $TPB:   177566              ;;TTY PRINTER BUFFER REG. ADDRESS
      001170     000            $NULL:  .BYTE   0           ;;CONTAINS NULL CHARACTER FOR FILLS
      001171     002            $FILLS: .BYTE   2           ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
      001172     012            $FILLC: .BYTE   12          ;;INSERT FILL CHARS. AFTER A ''LINE FEED''
      001173     000            $TPFLG: .BYTE   0           ;;''TERMINAL AVAILABLE'' FLAG (BIT<07>=0=YES)
      001174  000000            $REGAD: .WORD   0           ;;CONTAINS THE ADDRESS FROM
                                                           ;;WHICH  ($REG0) WAS OBTAINED
      001176  000000            $REG0:  .WORD   0           ;;CONTAINS (($REGAD)+0)
      001200  000000            $REG1:  .WORD   0           ;;CONTAINS (($REGAD)+2)
      001202  000000            $REG2:  .WORD   0           ;;CONTAINS (($REGAD)+4)
      001204  000000            $REG3:  .WORD   0           ;;CONTAINS (($REGAD)+6)
      001206  000000            $REG4:  .WORD   0           ;;CONTAINS (($REGAD)+10)
      001210  000000            $REG5:  .WORD   0           ;;CONTAINS (($REGAD)+12)
      001212  000000            $TMP0:  .WORD   0           ;;USER DEFINED
      001214  000000            $TMP1:  .WORD   0           ;;USER DEFINED
      001216  000000            $TMP2:  .WORD   0           ;;USER DEFINED
      001220  000000            $TIMES: 0                   ;;MAX. NUMBER OF ITERATIONS
      001222  000000            $ESCAPE:0                   ;;ESCAPE ON ERROR ADDRESS
      001224     207     377 377 $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
      001230     077            $QUES:  .ASCII  /?/         ;;QUESTION MARK
      001231     015            $CRLF:  .ASCII  <15>        ;;CARRIAGE RETURN
      001232     012     000    $LF:    .ASCIZ  <12>        ;;LINE FEED
                                ;;**********************************************************************
                                .SBTTL  APT MAILBOX-ETABLE
```

```
                                ;;*****************************************************************
                                .EVEN
001234                          $MAIL:                          ;;APT MAILBOX
001234  000000                  $MSGTY: .WORD   AMSGTY  ;;MESSAGE TYPE CODE
001236  000000                  $FATAL: .WORD   AFATAL  ;;FATAL ERROR NUMBER
001240  000000                  $TESTN: .WORD   ATESTN  ;;TEST NUMBER
001242  000000                  $PASS:  .WORD   APASS   ;;PASS COUNT
001244  000000                  $DEVCT: .WORD   ADEVCT  ;;DEVICE COUNT
001246  000000                  $UNIT:  .WORD   AUNIT   ;;I/O UNIT NUMBER
001250  000000                  $MSGAD: .WORD   AMSGAD  ;;MESSAGE ADDRESS
001252  000000                  $MSGLG: .WORD   AMSGLG  ;;MESSAGE LENGTH
001254                          $ETABLE:                        ;;APT ENVIRONMENT TABLE
001254     000                  $ENV:   .BYTE   AENV    ;;ENVIRONMENT BYTE
001255     000                  $ENVM:  .BYTE   AENVM   ;;ENVIRONMENT MODE BITS
001256  000000                  $SWREG: .WORD   ASWREG  ;;APT SWITCH REGISTER
001260  000000                  $USWR:  .WORD   AUSWR   ;;USER SWITCHES
001262  000000                  $CPUOP: .WORD   ACPUOP  ;;CPU TYPE,OPTIONS
                                ;*                      BITS 15-11=CPU TYPE
                                ;*                              11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
                                ;*                              11/70=06,PDQ=07,Q=10
                                ;*                      BIT 10=REAL TIME CLOCK
                                ;*                      BIT  9=FLOATING POINT PROCESSOR
                                ;*                      BIT  8=MEMORY MANAGEMENT
001264     000                  $MAMS1: .BYTE   AMAMS1  ;;HIGH ADDRESS,M.S. BYTE
001265     000                  $MTYP1: .BYTE   AMTYP1  ;;MEM. TYPE,BLK#1
                                ;*                      MEM.TYPE BYTE   --  (HIGH BYTE)
                                ;*                              900 NSEC CORE=001
                                ;*                              300 NSEC BIPOLAR=002
                                ;*                              500 NSEC MOS=003
001266  000000                  $MADR1: .WORD   AMADR1  ;;HIGH ADDRESS,BLK#1
                                ;*                        MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF ''TYPE'' ABOVE
001270     000                  $MAMS2: .BYTE   AMAMS2  ;;HIGH ADDRESS,M.S. BYTE
001271     000                  $MTYP2: .BYTE   AMTYP2  ;;MEM.TYPE,BLK#2
001272  000000                  $MADR2: .WORD   AMADR2  ;;MEM.LAST ADDRESS,BLK#2
001274     000                  $MAMS3: .BYTE   AMAMS3  ;;HIGH ADDRESS,M.S.BYTE
001275     000                  $MTYP3: .BYTE   AMTYP3  ;;MEM.TYPE,BLK#3
001276  000000                  $MADR3: .WORD   AMADR3  ;;MEM.LAST ADDRESS,BLK#3
001300     000                  $MAMS4: .BYTE   AMAMS4  ;;HIGH ADDRESS,M.S.BYTE
001301     000                  $MTYP4: .BYTE   AMTYP4  ;;MEM.TYPE,BLK#4
001302  000000                  $MADR4: .WORD   AMADR4  ;;MEM.LAST ADDRESS,BLK#4
001304  000000                  $VECT1: .WORD   AVECT1  ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
001306  000000                  $VECT2: .WORD   AVECT2  ;;INTERRUPT VECTOR#2BUS PRIORITY#2
001310  000000                  $BASE:  .WORD   ABASE   ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
001312  000000                  $DEVM:  .WORD   ADEVM   ;;DEVICE MAP
001314                          $ETEND:
                                .MEXIT
```

```
 0                              .SBTTL  USER DEFINED TAGS

001314  000000                 C.SWR:  .WORD   0               ;CONTROL SWITCHES
001316  000000                 SAVCSW: .WORD   0               ;PREVIOUS CONTENTS OF 'C.SWR'
001320  000000                 CNTRLC: .WORD   0               ;CONTROL "C" FLAG
001322  000000                 BUSADR: .WORD   0               ;GET ADDRESSES FROM THE TTY FLAG (0=NO, -1=YES)
001324  000000                 LPTAVL: .WORD   0               ;LPT AVAILABLE STATUS (0=NO,1=YES)
001326  000000                 DRVSEL: .WORD   0               ;DRIVES SELECTED FOR TESTING
001330  176777  000003         TSTNMS: .WORD   176777,3        ;DEFAULT IS RUN TESTS 0-10 & 12-21
001334  000000  000000         OPNFLG: .WORD   0,0             ;MODIFY TEST PARAMETER FLAGS
001340  000000                 CLKSTA: .WORD   0               ;CLOCK STATUS (0=NO CLOCK,+1=KW11-P,
                                                               ;AND -1=KW11-L)
001342  000020                 TICKMS: .WORD   16.             ;16 MILLISECONDS PER CLOCK TICK
001344  040432                 TICKUS: .WORD   16666.          ;16666 MIRCOSECONDS PER CLOCK TICK
001346  000000                 BYPASS: .WORD   0
001350  000000                 CHKDRV: .WORD   0               ;DRIVE UNDER TEST
001352  000000                 DRVMSK: .WORD   0               ;DRIVE MASK BIT
001354  000000                 SVSTAT: .WORD   0               ;STATUS/ERROR INDICATOR IS
                                                               ;SAVED HERE ON AN ERROR
001356  000000                 CYL.RD: .WORD   0               ;CYLINDER READ
001360  000000                 TRK.RD: .WORD   0               ;TRACK READ
001362  000000                 SEC.RD: .WORD   0               ;SECTOR READ
001364  000000                 CYL.DS: .WORD   0               ;CYLINDER DESIRED
001366  000000                 SEC.DS: .WORD   0               ;SECTOR DESIRED
001370  000000                 TRK.DS: .WORD   0               ;TRACK DESIRED
001372  000000                 LSTRK:  .WORD   0               ;CONTAINS THE LAST TRACK OF THE UNIT UNDER
                                                               ;TEST. RM02/3 = 4., RM05 = 18.
001374  000000                 TIM.UP: .WORD   0               ;MINIMUM TIME
001376  000000                         .WORD   0               ;NUMBER OF COUNTS BELOW MIN. LIMIT
001400  000000                         .WORD   0               ;MAXIMUM TIME
001402  000000                         .WORD   0               ;NUMBER OF COUNTS ABOVE MAX. LIMIT
001404  000000  000000                 .WORD   0,0             ;TOTAL TIME OF ALL SEEKS
001410  000000                         .WORD   0               ;NUMBER OF SEEKS PERFORMED
001412  000000                 TIM.DN: .WORD   0               ;MINIMUM TIME
001414  000000                         .WORD   0               ;NUMBER OF COUNTS BELOW MIN. LIMIT
001416  000000                         .WORD   0               ;MAXIMUM TIME
001420  000000                         .WORD   0               ;NUMBER OF COUNTS ABOVE MAX. LIMIT
001422  000000  000000                 .WORD   0,0             ;TOTAL TIME OF ALL SEEKS
001426  000000                         .WORD   0               ;NUMBER OF SEEKS PERFORMED
001430  000000                 TIM.PT: .WORD   0               ;POINTS TO TABLE OF TIMES
001432  000000                 WCEFLG: .WORD   0               ;FATAL WRITE CHECK ERROR FLAG
001434  000000                 STALL0: .WORD   0               ;VARIABLE STALL
001436  000000  000000         SVADR:  .WORD   0,0             ;SAVE DISK ADDRESS
001442  000000                 SEKTMR: .WORD   0               ;SEEK TIMER
001444  000000                 SEKCNT: .WORD   0               ;SEEK COUNTER
001446  000000                 DELTA:  .WORD   0               ;TESTING RANGE FOR SERVO SETTLE DOWN TEST
001450  160000                 TRCKWC: .WORD   -<256.*32.>     ;WORD COUNT FOR A FULL TRACK IN 16 BIT MODE
001452  000012                 STALL1: .WORD   10.             ;10 MILLISECONDS STALL
001454  000012                 STALL2: .WORD   10.             ;10 MILLISECONDS STALL
001456  011610                 STALL3: .WORD   5000.           ;5 SEC STALL
001460  000031                 MXSTAL: .WORD   25.             ;MAX. INCREMENTING STALL ALLOWED IN TEST 4
001462     144                 ERR.CT: .BYTE   100.            ;NUMBER OF ERRORS ALLOWED IN TESTS 16 - 21
                                                               ;BEFORE GOING TO THE NEXT TEST
001463     000                         .BYTE   0               ;RESERVED
001464  000000                 BASFLG: .WORD   0               ;FLAG FOR DETECTING BAD SECTOR
001466  000000                 XXDP:   .WORD   0               ;THE LOW BYTE CONTAINS THE DRIVE NUMBER FROM WHICH
                                                               ;THE PROGRAM WAS LOADED. THE HIGH BYTE CONTAINS THE
```

```
                                                    ;'XXDP' DEVICE CODE FOR THE RM05/3/2.

                                    ;ADDRESSES AND VECTORS
001470  176700                      RH.ADR: .WORD   176700          ;RH/RM UNIBUS ADDRESS
001472  000254  000240              RHVEC:  .WORD   254,240         ;VECTOR ADDRESS AND PRIORITY
001476  000104  000106              PKV:    .WORD   104,106         ;KW11-P VECTOR ADDRESS
001502  172540                      PKCS:   .WORD   172540          ;KW11-P CONTROL AND STATUS REG.
001504  172542                      PKB:    .WORD   172542          ;KW11-P COUNT SET BUFFER
001506  172544                      PKC:    .WORD   172544          ;KW11-P COUNTER
001510  000100  000102              LKV:    .WORD   100,102         ;KW11-L VECTOR ADDRESS
001514  177546                      LKS:    .WORD   177546          ;KW11-L STATUS REGISTER
001516  177564                      TPS:    .WORD   177564          ;TTY PRINTER STATUS
001520  177566                      TPB:    .WORD   177566          ;TTY PRINTER BUFFER
001522  177514                      LPS:    .WORD   177514          ;LINE PRINTER STATUS
001524  177516                      LPB:    .WORD   177516          ;LINE PRINTER BUFFER

                                    ;BIT TABLE
001526  000001                      BITS:   .WORD   BIT00
001530  000002                              .WORD   BIT01
001532  000004                              .WORD   BIT02
001534  000010                              .WORD   BIT03
001536  000020                              .WORD   BIT04
001540  000040                              .WORD   BIT05
001542  000100                              .WORD   BIT06
001544  000200                              .WORD   BIT07
001546  000400                              .WORD   BIT08
001550  001000                              .WORD   BIT09
001552  002000                              .WORD   BIT10
001554  004000                              .WORD   BIT11
001556  010000                              .WORD   BIT12
001560  020000                              .WORD   BIT13
001562  040000                              .WORD   BIT14
001564  100000                              .WORD   BIT15
001566  000001                              .WORD   BIT00
001570  000002                              .WORD   BIT01
001572  000004                              .WORD   BIT02
001574  000010                              .WORD   BIT03
001576  000020                              .WORD   BIT04
001600  000040                              .WORD   BIT05
001602  000100                              .WORD   BIT06
001604  000200                              .WORD   BIT07

                                    .SBTTL  TIMING LIMITS

                                    ;ROTATIONAL TEST TABLES FOR RM05/3 DRIVES
                                    ;60HZ TABLE
001606  046261                      T7A:    .WORD   ROTATE
001610  000000                              .WORD   0
001612  003142                              .WORD   1634.           ;LO LIMIT (16.67MS - 2%)
001614  003244                              .WORD   1700.           ;HI LIMIT (16.67MS + 2%)

                                    ;50HZ TABLE
001616  046261                      T7B:    .WORD   ROTATE
001620  000000                              .WORD   0
001622  003131                              .WORD   1625.           ;LO LIMIT (16.67MS - 2.5%)
001624  003255                              .WORD   1709.           ;HI LIMIT (16.67MS + 2.5%)
```

```
                                ;ROTATIONAL TEST TABLES FOR RM02 DRIVES
                                ;60HZ TABLE
001626  046261                  T7A1:   .WORD   ROTATE
001630  000000                          .WORD   0
001632  004622                          .WORD   2450.           ;LO LIMIT (25.00MS - 2%)
001634  004766                          .WORD   2550.           ;HI LIMIT (25.00MS + 2%)

                                ;50HZ TABLE
001636  046261                  T7B1:   .WORD   ROTATE
001640  000000                          .WORD   0
001642  004605                          .WORD   2437.           ;LO LIMIT (25.00MS - 2.5%)
001644  005003                          .WORD   2563.           ;HI LIMIT (25.00MS + 2.5%)

                                ;SEEK TEST TABLES
001646  046312                  T10:    .WORD   ONECYL          ;FORWARD
001650  046464                          .WORD   REV             ;REVERSE
001652  000000                          .WORD   0               ;NO LO LIMIT
001654  001274                          .WORD   700.            ;HI LIMIT (7MS)

001656  046357                  T11:    .WORD   ACCESS          ;FORWARD
001660  046464                          .WORD   REV             ;REVERSE
001662  000000                          .WORD   0               ;NO LO LIMIT
001664  006200                          .WORD   3200.           ;HI LIMIT (32MS)

001666  046424                  T12:    .WORD   MXSEEK          ;FORWARD
001670  046464                          .WORD   REV             ;REVERSE
001672  000000                          .WORD   0               ;NO LO LIMIT
001674  013104                          .WORD   5700.           ;HI LIMIT (57MS)

                                ;SPECS. MESSAGE TABLES FOR ROTATIONAL AND TIMING TESTS

                                ;ROTATIONAL MESSAGE AND LO/HI LIMITS FOR RM05/3 DRIVES
                                ;60HZ TABLE
001676  046662                  SP7A:   .WORD   MSG7XA
001700  003142                          .WORD   1634.           ;LO LIMIT (16.67MS - 2%)
001702  003244                          .WORD   1700.           ;HI LIMIT (16.67MS + 2%)

                                ;50HZ TABLE
001704  046662                  SP7B:   .WORD   MSG7XA
001706  003131                          .WORD   1625.           ;LO LIMIT (16.67MS - 2.5%)
001710  003255                          .WORD   1709.           ;HI LIMIT (16.67MS + 2.5%)

                                ;ROTATIONAL MESSAGE AND LO/HI LIMITS FOR RM02 DRIVES
                                ;60HZ TABLE
001712  046737                  SP7A1:  .WORD   MSG7XB
001714  004622                          .WORD   2450.           ;LO LIMIT (25.00MS - 2%)
001716  004766                          .WORD   2550.           ;HI LIMIT (25.00MS + 2%)

                                ;50HZ TABLE
001720  046737                  SP7B1:  .WORD   MSG7XB
001722  004605                          .WORD   2437.           ;LO LIMIT (25.00MS - 2.5%)
001724  005003                          .WORD   2563.           ;HI LIMIT (25.00MS + 2.5%)

                                ;TIMING TESTS MESSAGES AND LO/HI LIMITS
001726  047012                  SP10:   .WORD   MSG10X
001730  000000                          .WORD   0               ;NO LO LIMIT
```

```
001732  001274                          .WORD   700.            ;HI LIMIT (7MS)

001734  047054          SP11:  .WORD   MSG11X
001736  000000                 .WORD   0               ;NO LO LIMIT
001740  006200                 .WORD   3200.           ;HI LIMIT (32MS)

001742  047110          SP12:  .WORD   MSG12X
001744  000000                 .WORD   0               ;NO LO LIMIT
001746  013104                 .WORD   5700.           ;HI LIMIT (57MS)

                        ;STATUS/ERROR MESSAGE POINTER TABLE
001750  050012          STATBL: .WORD   MSGB14          ;OFFLINE OR UNSAFE DRIVE REQUESTED
001752  050054                  .WORD   MSGB13          ;UNLOAD DRIVE REQUESTED
001754  050105                  .WORD   MSGB12          ;PERSISTENT UNSAFE
001756  050127                  .WORD   MSGB11          ;PARITY ERROR OCCURRED
001760  050155                  .WORD   MSGB10          ;FATAL PARITY ERROR
001762  050200                  .WORD   MSGB09          ;SOFTWARE TIMEOUT ON THIS DRIVE
001764  050237                  .WORD   MSGB08          ;SOFTWARE TIMEOUT ON ANOTHER DRIVE
001766  050301                  .WORD   MSGB06          ;ERROR OCCURRED DURING I/O OPERATION
001770  050345                  .WORD   MSGB05          ;ERROR OCCURRED DURING NON-I/O OPERATION
001772  050415                  .WORD   MSGB04          ;UNSAFE OCCURRED
001774  050435                  .WORD   MSGB03          ;AUTOMATIC RECALIBRATE SEQUENCE OCCURRED
001776  050505                  .WORD   MSGB02          ;DRIVE HAS NOT RESPONDED TO PORT REQUEST
002000  050555                  .WORD   MSGB01          ;DRIVE HAS BECOME NONEXISTENT
```

```
      0                                 .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


    002002                              $ERRTB:
  1
  5                                     ;*ERROR  1
  6
  7                                     ;*    RH/RM INTERRUPT OCCURED (RMAS = 0)
  8                                     ;*    ERR PC  RMAS
  9                                     ;*    $ERRPC  $REG3
 10
 11 002002  047152                        EM1
 12 002004  050613                        DH1
 13 002006  052170                        DT1
 14 002010  052622                        DF1
 15
 16                                     ;*ERROR  2
 17
 18                                     ;*    UNEXPECTED ATTENTION OCCURRED
 19                                     ;*    ERR PC  DRIVE   RMAS    RMDS   RMER1     RMMR2     RMER2
 20                                     ;*    $ERRPC  $REG1   $REG3   RMERRS  RMERRS+2  RMERRS+4  RMERRS+6
 21
 22 002012  047216                        EM2
 23 002014  050630                        DH2
 24 002016  052174                        DT2
 25 002020  052626                        DF2
 26
 27                                     ;*ERROR  3
 28
 29                                     ;*    MASSBUS PARITY ERROR (MCPE=1)
 30                                     ;*    TEST    ERR PC  ADDRESS DATA
 31                                     ;*    $TMP0   $ERRPC  RD.ADR  RD.WRD
 32
 33 002022  047254                        EM3
 34 002024  050715                        DH3
 35 002026  052212                        DT3
 36 002030  052632                        DF3
 37
 38                                     ;*ERROR  4
 39
 40                                     ;*    MASSBUS PARITY ERROR (PAR=1)
 41                                     ;*    TEST    ERR PC  ADDRESS GDDATA  BDDATA
 42                                     ;*    $TMP0   $ERRPC  WRT.ADR WRT.WD  RD.WRD
 43
 44 002032  047311                        EM4
 45 002034  050752                        DH4
```

```
     46 002036  052222                        DT4
     47 002040  052636                        DF4
     48
     49                                ;*ERROR  5
     50
     51                                ;*    ADDRESS PLUG CHANGE BIT SET
     52                                ;*    ERR PC  DRIVE   RMAS    RMDS   RMER1     RMMR2    RMER2
     53                                ;*    $ERRPC  $REG1   $REG3   RMERRS  RMERRS+2  RMERRS+4 RMERRS+6
     54
     55 002042  047345                        EM5
     56 002044  050630                        DH2
     57 002046  052174                        DT2
     58 002050  052626                        DF2
     59
     60                                ;*ERROR  6 -- NOT USED
     61
     62 002052  000000                        0
     63 002054  000000                        0
     64 002056  000000                        0
     65 002060  000000                        0
     66
     67                                ;*ERROR  7 -- NOT USED
     68
     69 002062  000000                        0
     70 002064  000000                        0
     71 002066  000000                        0
     72 002070  000000                        0
     73
     74                                ;*ERROR  10
     75
     76                                ;*    RH/RM FAILED TO RESPOND TO ADDRESSING
     77                                ;*    RMCS1   ERR PC
     78                                ;*    RH.ADR  $ERRPC
     79
     80 002072  047401                          EM10
     81 002074  051021                          DH10
     82 002076  052254                          DT10
     83 002100  052642                          DF10
     84
     85                                ;*ERROR  11
     86
     87                                ;*    DRIVE SELECTED IS NOT ONLINE
     88                                ;*    DRIVE   ERR PC
     89                                ;*    $REG2   $ERRPC
     90
     91 002102  047447                          EM11
     92 002104  051040                          DH11
     93 002106  052260                          DT11
     94 002110  052646                          DF11
     95
     96                                ;*ERROR  12
     97
     98                                ;*    IMPROPER HEADER DATA
     99                                ;*    TEST    ERR PC  TST PC  DRIVE   CYLNDR  TRACK   SECTOR
    100                                ;*    $TMP0   $ERRPC  $REG0   CHKDRV  CYL.DS  TRK.DS  SEC.DS
    101                                ;*        GDCYL   GDTRK   GDSCTR  BDCYL   BDTRK   BDSCTR
    102                                ;*        CYL.DS  TRK.DS  SEC.DS  CYL.RD  TRK.RD  SEC.RD
```

```
                                ;*    CYLNDR, TRACK, AND SECTOR ARE DECIMAL

002112  047504                    EM12
002114  051057                    DH12
002116  052264                    DT12
002120  052652                    DF12

                                ;*ERROR  13

                                ;*    DATA COMPARE FAILURE
                                ;*    TEST    ERR PC  TST PC  DRIVE   CYLNDR  TRACK   SECTOR
                                ;*    $TMP0   $ERRPC  $REG0   CHKDRV  CYL.DS  TRK.DS  SEC.DS
                                ;*        GDDAT   BDDAT   WRDCNT  GDADR   BDADR
                                ;*        $GDDAT  $BDDAT  $REG4   $GDADR  $BDADR
                                ;*    CYLNDR, TRACK, SECTOR, AND WRDCNT  ARE DECIMAL

002122  047531                    EM13
002124  051057                    DH12
002126  052316                    DT13
002130  052662                    DF13

                                ;*ERROR  14 -- FOLLOWS #13

                                ;*        $GDDAT  $BDDAT  $REG4   $GDADR  $BDADR

002132  000000                      0
002134  000000                      0
002136  052334                    DT13A
002140  052672                    DF14

                                ;*ERROR  15

                                ;*    DATA COMPARE FAILURE
                                ;*    TEST    ERR PC  TST PC  DRIVE   CYLNDR  TRACK   SECTOR
                                ;*    $TMP0   $ERRPC  $REG0   CHKDRV  CYL.DS  TRK.DS  SEC.DS
                                ;*        GDDAT   BDDAT   WRDCNT  GDADR   BDADR
                                ;*        $GDDAT  $BDDAT  $REG4   $GDADR  $BDADR
                                ;*    CYLNDR, TRACK, SECTOR, AND WRDCNT ARE DECIMAL

002142  047531                    EM13
002144  051057                    DH12
002146  052316                    DT13
002150  052662                    DF13

                                ;*ERROR  16 -- FOLLOWS #15

                                ;*  $GDDAT  $BDDAT  $REG4   $GDADR  $BDADR

002152  000000                      0
002154  000000                      0
002156  052334                    DT13A
002160  052672                    DF14

                                ;*ERROR  17

                                ;*    DISK ERROR IN TIMING TEST
                                ;*    TEST    ERR PC  DRIVE   RMCS1   RMDS    RMER1    RMMR2    RMER2
```

```
                                ;*    $TMP0   $ERRPC  CHKDRV  RM.REG  RM.REG+12 RM.REG+14 RM.REG+40 RM.REG+42

002162  047556                        EM17
002164  051273                        DH17
002166  052346                        DT17
002170  052676                        DF17

                                ;*ERROR  20

                                ;*    CLOCK (KW11-P) OVERFLOW IN TIMING TEST
                                ;*    TEST    ERR PC  DRIVE   RMCS1   RMDS     RMER1     RMMR2     RMER2
                                ;*    $TMP0   $ERRPC  CHKDRV  RM.REG  RM.REG+12 RM.REG+14 RM.REG+40 RM.REG+42

002172  047610                        EM20
002174  051273                        DH17
002176  052346                        DT17
002200  052676                        DF17

                                ;*ERROR  21

                                ;*    DATA COMPARE FAILURE
                                ;*    TEST      ERR PC  TST PC  DRIVE   CYLNDR  TRACK
                                ;*    $TMP0     $ERRPC  $REG0   CHKDRV  CYL.DS  TRK.DS
                                ;*  GDDAT         BDDAT   WRDCNT  SECTOR
                                ;*  $REG1   $BDDAT  $REG4   $REG1
                                ;*  CYLINDR, TRACK, WRDCNT, AND SECTOR ARE DECIMAL

002202  047531                        EM13
002204  051370                        DH21
002206  052366                        DT21
002210  052702                        DF21

                                ;*ERROR  22--FOLLOWS #21

                                ;*        $REG1   $BDDAT   $REG4   $REG1

002212  000000                        0
002214  000000                        0
002216  052402                        DT21A
002220  052712                        DF22

                                ;*ERROR  23

                                ;*    DISK ERROR DURING SEEK
                                ;*    TEST      ERR PC  DRIVE   CYLNDR  RMCS1   RMCS2     RMDS
                                ;*    $TMP0     $ERRPC  CHKDRV  CYL.DS  RM.REG RM.REG+10 RM.REG+12
                                ;*       RMER1     RMMR2     RMER2     RMDC      RMHR
                                ;*       RM.REG+14 RM.REG+40 RM.REG+42 RM.REG+34 RM.REG+36

002222  047657                        EM23
002224  051505                        DH23
002226  052412                        DT23
002230  052716                        DF23

                                ;*ERROR  24

                                ;*    SEEK NOT COMPLETE WITHIN 120 MS
```

```
                                ;*    TEST      ERR PC  DRIVE   CYLNDR  RMCS1   RMCS2     RMDS
                                ;*    $TMP0     $ERRPC  CHKDRV  CYL.DS  RM.REG RM.REG+10 RM.REG+12
                                ;*       RMER1     RMMR2     RMER2     RMDC      RMHR
                                ;*       RM.REG+14 RM.REG+40 RM.REG+42 RM.REG+34 RM.REG+36

002232  047706                          EM24
002234  051505                          DH23
002236  052412                          DT23
002240  052716                          DF23

                                ;;*****************************************************************
                                ;*  ERRORS  23-40 NOT USED
                                ;*  ERRORS  41-46 WILL HAVE AN EM THAT

                                ;*  VARIES DEPENDING ON THE ERROR, IT WILL BE IN THE FORM:
                                ;*  RH/RM ERROR (MESSAGE)
                                ;*  WHERE MESSAGE WILL BE ONE OR MORE OF THE FOLLOWING:
                                ;*  1)  OFFLINE OR UNSAFE DRIVE REQUESTED
                                ;*  2)  UNLOADED DRIVE REQUESTED
                                ;*  3)  PERSISTENT UNSAFE
                                ;*  4)  PARITY ERROR OCCURRED
                                ;*  5)  FATAL PARITY ERROR
                                ;*  6)  SOFTWARE TIMEOUT ON THIS DRIVE
                                ;*  7)  SOFTWARE TIMEOUT ON ANOTHER DRIVE
                                ;*  8)  ERROR OCCURRED DURING I/O OPERATION
                                ;*  9)  ERROR OCCURRED DURING NON-I/O OPERATION
                                ;*  10) UNSAFE OCCURRED
                                ;*  11) AUTOMATIC RECALIBRATE SEQUENCE OCCURRED
                                ;;*****************************************************************

002242                          ITEM41:

                                ;*ERROR  41

                                ;*    RH/RM ERROR (MESSAGE)
                                ;*    TEST    ERR PC  TST PC  DRIVE
                                ;*    $TMP0   $ERRPC  $REG0   CHKDRV

002242  047746                          EM41
002244  051637                          DH41
002246  052442                          DT41
002250  052726                          DF41

                                ;*ERROR  42

                                ;*    RH/RM ERROR (MESSAGE)
                                ;*    TEST    ERR PC  TST PC  DRIVE   RMCS1   RMCS2   RMDS
                                ;*    $TMP0   $ERRPC  $REG0   CHKDRV  RM.REG  RM.REG+10 RM.REG+12

002252  047746                          EM41
002254  051675                          DH42
002256  052452                          DT42
002260  052732                          DF42

                                ;*ERROR  43

                                ;*    RH/RM ERROR (MESSAGE)
```

```
                                ;*      TEST    ERR PC  TST PC  DRIVE   RMCS1   RMCS2   RMDS
                                ;*      $TMPO   $ERRPC  $REGO   CHKDRV  RM.REG  RM.REG+10 RM.REG+12
                                ;*          RMER1       RMMR2       RMER2
                                ;*          RM.REG+14 RM.REG+40 RM.REG+42

002262  047746                          EM41
002264  051675                          DH42
002266  052470                          DT43
002270  052736                          DF43

                                ;*ERROR  44

                                ;*      RH/RM ERROR (MESSAGE)
                                ;*      TEST    ERR PC  TST PC  DRIVE   CYLNDR  TRACK     SECTOR
                                ;*      $TMPO   $ERRPC  $REGO   CHKDRV  CYL.DS  TRK.DS    SEC.DS
                                ;*          RMCS1       RMCS2     RMDS      RMHR      RMDC      RMDA
                                ;*          RM.REG      RM.REG+10 RM.REG+12 RM.REG+36 RM.REG+34 RM.REG+06
                                ;*      RMER1           RMMR2           RMER2
                                ;*      RM.REG+14    RM.REG+40     RM.REG+42
                                ;*      CYLNDR,TRACK, AND SECTOR ARE DECIMAL

002272  047746                          EM41
002274  051057                          DH12
002276  052514                          DT44
002300  052746                          DF44

                                ;*ERROR  45

                                ;*      RH/RM ERROR (MESSAGE)
                                ;*      TEST    ERR PC  TST PC  DRIVE   CYLNDR  TRACK     SECTOR
                                ;*      $TMPO   $ERRPC  $REGO   CHKDRV  CYL.DS  TRK.DS    SEC.DS
                                ;*          RMCS1       RMCS2     RMDS      RMHR      RMDC      RMDA
                                ;*          RM.REG      RM.REG+10 RM.REG+12 RM.REG+36 RM.REG+34 RM.REG+06
                                ;*      RMER1           RMMR2         RMER2       RMWC        RMBA      RMDB
                                ;*      RM.REG+14   RM.REG+40   RM.REG+42   RM.REG+2    RM.REG+4 RM.REG+22
                                ;*      CYLNDR,TRACK, AND SECTOR ARE DECIMAL

002302  047746                          EM41
002304  051057                          DH12
002306  052554                          DT45
002310  052762                          DF45

                                ;*ERROR  46

                                ;*      FATAL WRITE CHECK ERROR (MESSAGE)
                                ;*      TEST    ERR PC  TST PC  DRIVE   CYLNDR  TRACK     SECTOR
                                ;*      $TMPO   $ERRPC  $REGO   CHKDRV  CYL.DS  TRK.DS    SEC.DS
                                ;*          RMCS1       RMCS2     RMDS      RMHR      RMDC      RMDA
                                ;*          RM.REG      RM.REG+10 RM.REG+12 RM.REG+36 RM.REG+34 RM.REG+06
                                ;*      RMER1           RMMR2         RMER2       RMWC        RMBA      RMDB
                                ;*      RM.REG+14   RM.REG+40   RM.REG+42   RM.REG+2    RM.REG+4 RM.REG+22
                                ;*      CYLNDR,TRACK, AND SECTOR ARE DECIMAL

002312  047762                          EM46
002314  051057                          DH12
002316  052554                          DT45
002320  052762                          DF45
```

331

```
                                        .SBTTL  TEST PARAMETER POINTERS AND TABLES

                                        ;COMMON STORAGE FOR TEST PARAMETERS
002322  000000                  PRM:    .WORD   0               ;THIS WORD TELLS WHICH OF THE
                                                                ;FOLLOWING PARAMETERS ARE TO BE USED
002324  000000                  RPT:    .WORD   0               ;REPEAT COUNTS FOR ALL TESTS
002326  000000                  FC:     .WORD   0               ;FIRST CYLINDER
002330  000000                  LC:     .WORD   0               ;LAST CYLINDER
002332  000000                  IC:     .WORD   0               ;INCREMENT CYLINDER
002334  000000                  FT:     .WORD   0               ;FIRST TRACK
002336  000000                  LT:     .WORD   0               ;LAST TRACK
002340  000000                  IT:     .WORD   0               ;INCREMENT TRACK
002342  000000                  FS:     .WORD   0               ;FIRST SECTOR
002344  000000                  LS:     .WORD   0               ;LAST SECTOR
002346  000000                  PAT:    .WORD   0               ;PATTERN CODE
002350  000000  000000  000000          .WORD   0,0,0           ;FILLER WORDS FOR COMMON TABLE USED BY THE
                                                                ;'OPNTST' ROUTINE.

002356  000000                  NC1:    .WORD   0               ;NEW CYLINDER ADDRESS
002360  000000                  NC2:    .WORD   0               ;NEW CYLINDER ADDRESS

                                        ;TABLE OF PARAMETER POINTERS
002362  003116                  PRMPT:  .WORD   PRM0
002364  003132                          .WORD   PRM1
002366  003160                          .WORD   PRM2
002370  003200                          .WORD   PRM3
002372  003220                          .WORD   PRM4
002374  003240                          .WORD   PRM5
002376  003260                          .WORD   PRM6
002400  003300                          .WORD   PRM7
002402  003320                          .WORD   PRM10
002404  003336                          .WORD   PRM11
002406  003356                          .WORD   PRM12
002410  003372                          .WORD   PRM13
002412  003402                          .WORD   PRM14
002414  003412                          .WORD   PRM15
002416  003422                          .WORD   PRM16
002420  003434                          .WORD   PRM17
002422  003444                          .WORD   PRM20
002424  003500                          .WORD   PRM21
002426  003510                          .WORD   PRM22
002430  000000                          .WORD   0               ;TERMINATOR

                                ;TABLE OF PARAMETER UPPER LIMITS
002432  032767                  PRMLMT: .WORD   32767           ;''R''
002434  001466                          .WORD   822.            ;''FC''
002436  001466                          .WORD   822.            ;''LC''
002440  001466                          .WORD   822.            ;''IC''
002442  000004                          .WORD   4               ;''FT''
002444  000004                          .WORD   4               ;''LT''
002446  000004                          .WORD   4               ;''IT''
002450  000022                          .WORD   18.             ;''FT'''
002452  000022                          .WORD   18.             ;''LT'''
002454  000022                          .WORD   18.             ;''IT'''
002456  000037                          .WORD   31.             ;''FS''
002460  000037                          .WORD   31.             ;''LS''
002462  177777                          .WORD   177777          ;''PAT''
```

```
                                        ;TABLE OF PARAMETER MESSAGE POINTERS
002464  045626                  PRMMSG: .WORD   MSG.R
002466  045630                          .WORD   MSG.FC
002470  045633                          .WORD   MSG.LC
002472  045636                          .WORD   MSG.IC
002474  045641                          .WORD   MSG.FT
002476  045644                          .WORD   MSG.LT
002500  045647                          .WORD   MSG.IT
002502  045652                          .WORD   MES.FT
002504  045656                          .WORD   MES.LT
002506  045662                          .WORD   MES.IT
002510  045666                          .WORD   MSG.FS
002512  045671                          .WORD   MSG.LS

                                        ;STATUS/ERROR INDICATOR MESSAGES POINTER TABLE
                                        ;DEFAULT VALUES OF TEST PARAMETERS

002514  002225  000310  001466  DFLT:   .WORD   2225,200.,822.,0,0,0    ;RECAL/SEEK (T0)
002530  006677  000144  000000          .WORD   6677,100.,0,256.,0,0,0,0,0,0,0,0 ;SEEK/SEEK (T1)
002556  002237  000001  000000          .WORD   2237,1,0,822.,1,0,0,0   ;INCREMENT SEEK (T2)
002576  002237  000010  000000          .WORD   2237,10,0,512.,1,0,0,0  ;STEPPING SEEK (T3)
002616  002237  000001  000000          .WORD   2237,1,0,822.,1,0,0,0   ;OSCILLATING SEEK (T4)
002636  002237  000001  000000          .WORD   2237,1,0,822.,1,0,0,0   ;CONVERGING/DIVERGING SEEK (T5)
002656  002237  000001  000000          .WORD   2237,1,0,822.,1,0,0,0   ;SERVO ADDRESSING LOGIC NOISE (T6)
002676  000667  011610  000000          .WORD   667,5000.,0,822.,0,4,0,18.      ;RANDOM SEEK TEST (T7)
002716  000237  000001  000000          .WORD   237,1,0,822.,100.,0,0   ;SERVO SETTLE DOWN TEST (T10)
002734  002237  000001  000000          .WORD   2237,1,0,822.,1,0,0,0   ;ALL SEEKS TEST (T11)
002754  002223  000001  000000          .WORD   2223,1,0,0,0,0          ;ROTATIONAL SPEED TIMING TEST (T12)
002770  000007  000001  000000          .WORD   7,1,0,822.              ;ONE CYLINDER SEEK TIMING TEST (T13)
003000  000007  000001  000000          .WORD   7,1,0,255.              ;ACCESS TIME MEASUREMENT TEST (T14)
003010  000007  000001  000000          .WORD   7,1,0,822.              ;MAXIMUM SEEK TIMING TEST (T15)
003020  000223  000001  000000          .WORD   223,1,0,0,0             ;SECTOR ADDRESSING TEST (T16)
003032  002003  000001  000000          .WORD   2003,1,0,0              ;TRACK ADDRESSING TEST (T17)
003042  017777  000001  000000          .WORD   17777,1,0,821.,64.,0,4,1,0,18.,1,1,0,177777 ;DATA TEST (T20)
003076  000007  001750  000000          .WORD   7,1000.,0,821.          ;EXERCISER (T21)
003106  000007  011610  000000          .WORD   7,5000.,0,255.          ;ACCESS TIME ADJUSTMENT TEST (T22)

                                        ;PARAMETER TABLES

                                        ;RECAL/SEEK (T0)
003116  002225                  PRM0:   .WORD   2225
003120  000310                          .WORD   200.
003122  001466                          .WORD   822.
003124  000000                          .WORD   0
003126  000000                          .WORD   0
003130  000000                          .WORD   0

                                        ;SEEK/SEEK (T1)
003132  006677                  PRM1:   .WORD   6677
003134  000144                          .WORD   100.
003136  000000                          .WORD   0
003140  000400                          .WORD   256.
003142  000000                          .WORD   0
003144  000000                          .WORD   0
003146  000000                          .WORD   0
003150  000000                          .WORD   0
```

```
    100 003152  000000                        .WORD   0
    101 003154  000000                        .WORD   0
    102 003156  000000                        .WORD   0
    103
    104                                 ;INCREMENT SEEK (T2)
    105 003160  002237                  PRM2:   .WORD   2237
    106 003162  000001                          .WORD   1
    107 003164  000000                          .WORD   0
    108 003166  001466                          .WORD   822.
    109 003170  000001                          .WORD   1
    110 003172  000000                          .WORD   0
    111 003174  000000                          .WORD   0
    112 003176  000000                          .WORD   0
    113
    114                                 ;STEPPING SEEK (T3)
    115 003200  002237                  PRM3:   .WORD   2237
    116 003202  000001                          .WORD   1
    117 003204  000000                          .WORD   0
    118 003206  001000                          .WORD   512.
    119 003210  000001                          .WORD   1
    120 003212  000000                          .WORD   0
    121 003214  000000                          .WORD   0
    122 003216  000000                          .WORD   0
    123
    124                                 ;OSCILLATING SEEK (T4)
    125 003220  002237                  PRM4:   .WORD   2237
    126 003222  000001                          .WORD   1
    127 003224  000000                          .WORD   0
    128 003226  001466                          .WORD   822.
    129 003230  000001                          .WORD   1
    130 003232  000000                          .WORD   0
    131 003234  000000                          .WORD   0
    132 003236  000000                          .WORD   0
    133
    134                                 ;CONVERGING/DIVERGING SEEK (T5)
    135 003240  002237                  PRM5:   .WORD   2237
    136 003242  000001                          .WORD   1
    137 003244  000000                          .WORD   0
    138 003246  001466                          .WORD   822.
    139 003250  000001                          .WORD   1
    140 003252  000000                          .WORD   0
    141 003254  000000                          .WORD   0
    142 003256  000000                          .WORD   0
    143
    144                                 ;SERVO ADDRESSING LOGIC NOISE GENERATOR (T6)
    145 003260  002237                  PRM6:   .WORD   2237
    146 003262  000001                          .WORD   1
    147 003264  000000                          .WORD   0
    148 003266  001466                          .WORD   822.
    149 003270  000001                          .WORD   1
    150 003272  000000                          .WORD   0
    151 003274  000000                          .WORD   0
    152 003276  000000                          .WORD   0
    153
    154                                 ;RANDOM SEEK TEST (T7)
    155 003300  000667                  PRM7:   .WORD   667
    156 003302  011610                          .WORD   5000.
```

```
                                        .WORD   0
                                        .WORD   822.
                                        .WORD   0
                                        .WORD   4
                                        .WORD   0
                                        .WORD   18.

                                ;SERVO SETTLE DOWN TEST (T10)
                                PRM10:  .WORD   237
                                        .WORD   1
                                        .WORD   0
                                        .WORD   822.
                                        .WORD   100.
                                        .WORD   0
                                        .WORD   0

                                ;ALL SEEKS TEST (T11)
                                PRM11:  .WORD   2237
                                        .WORD   1
                                        .WORD   0
                                        .WORD   822.
                                        .WORD   1
                                        .WORD   0
                                        .WORD   0
                                        .WORD   0

                                ;ROTATIONAL SPEED TIMING TEST (T12)
                                PRM12:  .WORD   2223
                                        .WORD   1
                                        .WORD   0
                                        .WORD   0
                                        .WORD   0
                                        .WORD   0

                                ;ONE CYLINDER SEEK TIMING TEST (T13)
                                PRM13:  .WORD   7
                                        .WORD   1
                                        .WORD   0
                                        .WORD   822.

                                ;ACCESS TIME MEASUREMENT TEST (T14)
                                PRM14:  .WORD   7
                                        .WORD   1
                                        .WORD   0
                                        .WORD   255.

                                ;MAXIMUM SEEK TIMING TEST (T15)
                                PRM15:  .WORD   7
                                        .WORD   1
                                        .WORD   0
                                        .WORD   822.

                                ;SECTOR ADDRESSING TEST (T16)
                                PRM16:  .WORD   223
                                        .WORD   1
                                        .WORD   0
                                        .WORD   0
```

```
                                        .WORD   0

                                ;TRACK ADDRESSING TEST (T17)
003434  002003                  PRM17:  .WORD   2003
003436  000001                          .WORD   1
003440  000000                          .WORD   0
003442  000000                          .WORD   0

                                ;DATA TEST (T20)
003444  017777                  PRM20:  .WORD   17777
003446  000001                          .WORD   1
003450  000000                          .WORD   0
003452  001465                          .WORD   821.
003454  000100                          .WORD   64.
003456  000000                          .WORD   0
003460  000004                          .WORD   4
003462  000001                          .WORD   1
003464  000000                          .WORD   0
003466  000022                          .WORD   18.
003470  000001                          .WORD   1
003472  000001                          .WORD   1
003474  000000                          .WORD   0
003476  177777                  PTRN15: .WORD   177777

                                ;EXERCISER (T21)
003500  000007                  PRM21:  .WORD   7
003502  001750                          .WORD   1000.
003504  000000                          .WORD   0
003506  001465                          .WORD   821.

                                ;ACCESS TIME ADJUSTMENT TEST (T22)
003510  000007                  PRM22:  .WORD   7
003512  011610                          .WORD   5000.
003514  000000                          .WORD   0
003516  000377                          .WORD   255.
```

```
                                .SBTTL  DATA PATTERN POINTERS AND TABLES

003520  003560          PAT.PT: .WORD   PAT0            ;DATA PATTERN 0
003522  003620                  .WORD   PAT1            ;DATA PATTERN 1
003524  003660                  .WORD   PAT2            ;DATA PATTERN 2
003526  003720                  .WORD   PAT3            ;DATA PATTERN 3
003530  003760                  .WORD   PAT4            ;DATA PATTERN 4
003532  004020                  .WORD   PAT5            ;DATA PATTERN 5
003534  004060                  .WORD   PAT6            ;DATA PATTERN 6
003536  004120                  .WORD   PAT7            ;DATA PATTERN 7
003540  004160                  .WORD   PAT8            ;DATA PATTERN 8
003542  004220                  .WORD   PAT9            ;DATA PATTERN 9
003544  004260                  .WORD   PAT10           ;DATA PATTERN 10
003546  004320                  .WORD   PAT11           ;DATA PATTERN 11
003550  004360                  .WORD   PAT12           ;DATA PATTERN 12
003552  004420                  .WORD   PAT13           ;DATA PATTERN 13
003554  004460                  .WORD   PAT14           ;DATA PATTERN 14
003556  004520                  .WORD   PAT15           ;DATA PATTERN 15

003560  066666          PAT0:   .WORD   066666          ;PATTERN 0
003562  066666                  .WORD   066666
003564  066666                  .WORD   066666
003566  066666                  .WORD   066666
003570  066666                  .WORD   066666
003572  066666                  .WORD   066666
003574  066666                  .WORD   066666
003576  066666                  .WORD   066666
003600  066666                  .WORD   066666
003602  066666                  .WORD   066666
003604  066666                  .WORD   066666
003606  066666                  .WORD   066666
003610  066666                  .WORD   066666
003612  066666                  .WORD   066666
003614  066666                  .WORD   066666
003616  066666                  .WORD   066666

003620  000001          PAT1:   .WORD   000001          ;PATTERN 1
003622  000003                  .WORD   000003
003624  000007                  .WORD   000007
003626  000017                  .WORD   000017
003630  000037                  .WORD   000037
003632  000077                  .WORD   000077
003634  000177                  .WORD   000177
003636  000377                  .WORD   000377
003640  000777                  .WORD   000777
003642  001777                  .WORD   001777
003644  003777                  .WORD   003777
003646  007777                  .WORD   007777
003650  017777                  .WORD   017777
003652  037777                  .WORD   037777
003654  077777                  .WORD   077777
003656  177777                  .WORD   177777

003660  177776          PAT2:   .WORD   177776          ;PATTERN 2
003662  177774                  .WORD   177774
003664  177770                  .WORD   177770
003666  177760                  .WORD   177760
```

```
      46 003670  177740                          .WORD   177740
      47 003672  177700                          .WORD   177700
      48 003674  177600                          .WORD   177600
      49 003676  177400                          .WORD   177400
      50 003700  177000                          .WORD   177000
      51 003702  176000                          .WORD   176000
      52 003704  174000                          .WORD   174000
      53 003706  170000                          .WORD   170000
      54 003710  160000                          .WORD   160000
      55 003712  140000                          .WORD   140000
      56 003714  100000                          .WORD   100000
      57 003716  000000                          .WORD   000000
      58
      59 003720  000000                PAT3:     .WORD   000000          ;PATTERN 3
      60 003722  000000                          .WORD   000000
      61 003724  000000                          .WORD   000000
      62 003726  177777                          .WORD   177777
      63 003730  177777                          .WORD   177777
      64 003732  177777                          .WORD   177777
      65 003734  000000                          .WORD   000000
      66 003736  000000                          .WORD   000000
      67 003740  177777                          .WORD   177777
      68 003742  177777                          .WORD   177777
      69 003744  000000                          .WORD   000000
      70 003746  177777                          .WORD   177777
      71 003750  000000                          .WORD   000000
      72 003752  177777                          .WORD   177777
      73 003754  000000                          .WORD   000000
      74 003756  177777                          .WORD   177777
      75
      76 003760  000000                PAT4:     .WORD   000000          ;PATTERN 4
      77 003762  010421                          .WORD   010421
      78 003764  021042                          .WORD   021042
      79 003766  031463                          .WORD   031463
      80 003770  042104                          .WORD   042104
      81 003772  052525                          .WORD   052525
      82 003774  063146                          .WORD   063146
      83 003776  073567                          .WORD   073567
      84 004000  104210                          .WORD   104210
      85 004002  114631                          .WORD   114631
      86 004004  125252                          .WORD   125252
      87 004006  135673                          .WORD   135673
      88 004010  146314                          .WORD   146314
      89 004012  156735                          .WORD   156735
      90 004014  167356                          .WORD   167356
      91 004016  177777                          .WORD   177777
      92
      93 004020  052525                PAT5:     .WORD   052525          ;PATTERN 5
      94 004022  052525                          .WORD   052525
      95 004024  052525                          .WORD   052525
      96 004026  125252                          .WORD   125252
      97 004030  125252                          .WORD   125252
      98 004032  125252                          .WORD   125252
      99 004034  052525                          .WORD   052525
     100 004036  052525                          .WORD   052525
     101 004040  125252                          .WORD   125252
     102 004042  125252                          .WORD   125252
```

```
                                                .WORD   052525
                                                .WORD   125252
                                                .WORD   052525
                                                .WORD   125252
                                                .WORD   052525
                                                .WORD   125252

                        PAT6:   .WORD   007417          ;PATTERN 6
                                .WORD   007417
                                .WORD   007417
                                .WORD   170360
                                .WORD   170360
                                .WORD   170360
                                .WORD   007417
                                .WORD   007417
                                .WORD   170360
                                .WORD   170360
                                .WORD   007417
                                .WORD   170360
                                .WORD   007417
                                .WORD   170360
                                .WORD   007417

                        PAT7:   .WORD   026455          ;PATTERN 7
                                .WORD   026455
                                .WORD   026455
                                .WORD   151322
                                .WORD   151322
                                .WORD   151322
                                .WORD   026455
                                .WORD   026455
                                .WORD   151322
                                .WORD   026455
                                .WORD   151322
                                .WORD   026455
                                .WORD   151322

                        PAT8:   .WORD   165555          ;PATTERN 8
                                .WORD   133333
                                .WORD   165555
                                .WORD   133333
                                .WORD   165555
                                .WORD   133333
                                .WORD   165555
                                .WORD   133333
                                .WORD   165555
                                .WORD   133333
                                .WORD   165555
                                .WORD   133333
                                .WORD   165555
                                .WORD   133333
```

```
                    
004220  000001      PAT9:   .WORD   000001          ;PATTERN 9
004222  000002              .WORD   000002
004224  000004              .WORD   000004
004226  000010              .WORD   000010
004230  000020              .WORD   000020
004232  000040              .WORD   000040
004234  000100              .WORD   000100
004236  000200              .WORD   000200
004240  000400              .WORD   000400
004242  001000              .WORD   001000
004244  002000              .WORD   002000
004246  004000              .WORD   004000
004250  010000              .WORD   010000
004252  020000              .WORD   020000
004254  040000              .WORD   040000
004256  100000              .WORD   100000
                    
004260  177776      PAT10:  .WORD   177776          ;PATTERN 10
004262  177775              .WORD   177775
004264  177773              .WORD   177773
004266  177767              .WORD   177767
004270  177757              .WORD   177757
004272  177737              .WORD   177737
004274  177677              .WORD   177677
004276  177577              .WORD   177577
004300  177377              .WORD   177377
004302  176777              .WORD   176777
004304  175777              .WORD   175777
004306  173777              .WORD   173777
004310  167777              .WORD   167777
004312  157777              .WORD   157777
004314  137777              .WORD   137777
004316  077777              .WORD   077777
                    
004320  172666      PAT11:  .WORD   172666          ;PATTERN 11
004322  155555              .WORD   155555
004324  172666              .WORD   172666
004326  155555              .WORD   155555
004330  172666              .WORD   172666
004332  155555              .WORD   155555
004334  172666              .WORD   172666
004336  155555              .WORD   155555
004340  172666              .WORD   172666
004342  155555              .WORD   155555
004344  172666              .WORD   172666
004346  155555              .WORD   155555
004350  172666              .WORD   172666
004352  155555              .WORD   155555
004354  172666              .WORD   172666
004356  155555              .WORD   155555
                    
004360  077777      PAT12:  .WORD   077777          ;PATTERN 12
004362  137777              .WORD   137777
004364  157777              .WORD   157777
004366  167777              .WORD   167777
004370  173777              .WORD   173777
```

```
                                                .WORD   175777
                                                .WORD   176777
                                                .WORD   177377
                                                .WORD   177577
                                                .WORD   177677
                                                .WORD   177737
                                                .WORD   177757
                                                .WORD   177767
                                                .WORD   177773
                                                .WORD   177775
                                                .WORD   177776

                                        PAT13:  .WORD   153333          ;PATTERN 13
                                                .WORD   066667
                                                .WORD   153333
                                                .WORD   066667
                                                .WORD   153333
                                                .WORD   066667
                                                .WORD   153333
                                                .WORD   066667
                                                .WORD   153333
                                                .WORD   066667
                                                .WORD   153333
                                                .WORD   066667
                                                .WORD   153333
                                                .WORD   066667

                                        PAT14:  .WORD   000000          ;PATTERN 14
                                                .WORD   177777
                                                .WORD   177777
                                                .WORD   177777
                                                .WORD   177777
                                                .WORD   177777
                                                .WORD   177777
                                                .WORD   177777
                                                .WORD   177777
                                                .WORD   177777
                                                .WORD   177777
                                                .WORD   177777
                                                .WORD   177777
                                                .WORD   177777

                                        PAT15:  .WORD   177777          ;PATTERN 15
                                                .WORD   000000
                                                .WORD   000000
                                                .WORD   000000
                                                .WORD   000000
                                                .WORD   000000
                                                .WORD   000000
                                                .WORD   000000
                                                .WORD   000000
                                                .WORD   000000
                                                .WORD   000000
```

```
    274 004546  000000                        .WORD   000000
    275 004550  000000                        .WORD   000000
    276 004552  000000                        .WORD   000000
    277 004554  000000                        .WORD   000000
    278 004556  000000                        .WORD   000000
    279
    460
```

```
                                                                    .SBTTL  START OF PROGRAM

004560  012737  177777  001322  START3: MOV     #-1,BUSADR          ;GET BUSADR FLAG
004566  000402                          BR      STRT1A
004570  005037  001322          START1: CLR     BUSADR              ;CLR BUSADR FLAG
004574  005037  001320          STRT1A: CLR     CNTRLC              ;NO CONTROL "C"
004600  000411                          BR      START
004602  012737  177777  001322  START4: MOV     #-1,BUSADR          ;SET BUSADR FLAG
004610  000402                          BR      STRT2A
004612  005037  001322          START2: CLR     BUSADR              ;CLR BUSADR FLAG
004616  012737  177777  001320  STRT2A: MOV     #-1,CNTRLC          ;SET CONTROL "C" FLAG
004624  000240                  START:  NOP
004626  005227  000000                  INC     #0                  ;TTY LOOP, WAIT FOR INCREMENT
004632  001375                          BNE     .-4                 ;OF WORD
004634  000005                          RESET                       ;CLEAR THE WORLD

                                                                    .SBTTL  INITIALIZE THE COMMON TAGS
                                                                    ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
004636  012706  001114                  MOV     #$CMTAG,R6          ;;FIRST LOCATION TO BE CLEARED
004642  005026                          CLR     (R6)+               ;;CLEAR MEMORY LOCATION
004644  022706  001154                  CMP     #SWR,R6             ;;DONE?
004650  001374                          BNE     .-6                 ;;LOOP BACK IF NO
004652  012706  001100                  MOV     #STACK,SP           ;;SETUP THE STACK POINTER
                                                                    ;;INITIALIZE A FEW VECTORS
004656  012737  024646  000020          MOV     #$SCOPE,@#IOTVEC    ;;IOT VECTOR FOR SCOPE ROUTINE
004664  012737  000340  000022          MOV     #340,@#IOTVEC+2     ;;LEVEL 7
004672  012737  021532  000030          MOV     #$ERROR,@#EMTVEC    ;;EMT VECTOR FOR ERROR ROUTINE
004700  012737  000340  000032          MOV     #340,@#EMTVEC+2     ;;LEVEL 7
004706  012737  025174  000034          MOV     #$TRAP,@#TRAPVEC    ;;TRAP VECTOR FOR TRAP CALLS
004714  012737  000340  000036          MOV     #340,@#TRAPVEC+2    ;LEVEL 7
004722  013737  021200  021172          MOV     $ENDCT,$EOPCT       ;;SETUP END-OF-PROGRAM COUNTER
004730  012737  176543  025646          MOV     #176543,$HINUM      ;;PRIME THE RANDOM NUMBER GENERATOR
004736  012737  123456  025650          MOV     #123456,$LONUM      ;;BOTH HIGH AND LOW WORDS
004744  005037  001220                  CLR     $TIMES              ;;INITIALIZE NUMBER OF ITERATIONS
004750  005037  001222                  CLR     $ESCAPE             ;;CLEAR THE ESCAPE ON ERROR ADDRESS
004754  112737  000001  001131          MOVB    #1,$ERMAX           ;;ALLOW ONE ERROR PER TEST
004762  012737  004762  001122          MOV     #.,$LPADR           ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
004770  012737  004770  001124          MOV     #.,$LPERR           ;;SETUP THE ERROR LOOP ADDRESS
                                                                    ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
                                                                    ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
004776  013746  000004                  MOV     @#ERRVEC,-(SP)      ;;SAVE ERROR VECTOR
005002  012737  005036  000004          MOV     #64$,@#ERRVEC       ;;SET UP ERROR VECTOR
005010  012737  177570  001154          MOV     #DSWR,SWR           ;;SETUP FOR A HARDWARE SWICH REGISTER
005016  012737  177570  001156          MOV     #DDISP,DISPLAY      ;;AND A HARDWARE DISPLAY REGISTER
005024  022777  177777  174122          CMP     #-1,@SWR            ;;TRY TO REFERENCE HARDWARE SWR
005032  001012                          BNE     66$                 ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                                    ;;AND  THE HARDWARE SWR IS NOT = -1
005034  000403                          BR      65$                 ;;BRANCH IF NO TIMEOUT
005036  012716  005044          64$:    MOV     #65$,(SP)           ;;SET UP FOR TRAP RETURN
005042  000002                          RTI
005044  012737  000176  001154  65$:    MOV     #SWREG,SWR          ;;POINT TO SOFTWARE SWR
005052  012737  000174  001156          MOV     #DISPREG,DISPLAY
005060  012637  000004          66$:    MOV     (SP)+,@#ERRVEC      ;;RESTORE ERROR VECTOR
```

```
    005064  005037  001242                          CLR     $PASS           ;;CLEAR PASS COUNT
    005070  132737  000200  001255                  BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
    005076  001403                                  BEQ     67$             ;;YES,USE NON-APT SWITCH
    005100  012737  001256  001154                  MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
    005106                                  67$:
 22                                         .SBTTL  TYPE PROGRAM NAME
                                            ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
    005106  005227  177777                          INC     #-1             ;;FIRST TIME?
    005112  001054                                  BNE     68$             ;;BRANCH IF NO
    005114  022737  021244  000042                  CMP     #$ENDAD,@#42    ;;ACT-11?
    005122  001450                                  BEQ     68$             ;;BRANCH IF YES
    005124  104401  005172                          TYPE    ,69$            ;;TYPE ASCIZ STRING
                                            .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
    005130  005737  000042                          TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
    005134  001012                                  BNE     70$             ;;BRANCH IF YES
    005136  123727  001254  000001                  CMPB    $ENV,#1         ;;ARE WE RUNNING UNDER APT?
    005144  001406                                  BEQ     70$             ;;BRANCH IF YES
    005146  023727  001154  000176                  CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
    005154  001005                                  BNE     71$             ;;BRANCH IF NO
    005156  104406                                  GTSWR                   ;;GET SOFT-SWR SETTINGS
    005160  000403                                  BR      71$
    005162  112737  000001  001150          70$:    MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
    005170                                  71$:
    005170  000425                                  BR      68$             ;;GET OVER THE ASCIZ
                                            ;;69$:  .ASCIZ  <CRLF>@CZRMVA0 - RM05/3/2 EXTENDED DRIVE TEST@<CRLF>
    005244                                  68$:
 23
 24 005244  012700  001174                          MOV     #$REGAD,R0      ;FIRST ADDRESS
 25 005250  005020                          1$:     CLR     (R0)+           ;CLEAR VARIABLE STORAGE
 26 005252  022700  001224                          CMP     #$BELL,R0       ;DONE?
 27 005256  001374                                  BNE     1$              ;NO--BRANCH
 28 005260  013737  001516  001164                  MOV     TPS,$TPS        ;SETUP THE STATUS AND BUFFER REG'S
 29 005266  013737  001520  001166                  MOV     TPB,$TPB        ;FOR THE TYPE ROUTINE
 30
 31                                         ;THE FOLLOWING FINDS OUT THE PROGRAM CONTROL MODE:
 32                                         ;PAPER TAPE (MANUAL), ACT11, XXDP CHAIN OR DUMP
 33
 34 005274  005037  001466                          CLR     XXDP            ;CLEAR 'XXDP' LOAD DEVICE STORAGE
 35 005300  122737  000016  000041                  CMPB    #16,@#41        ;LOADED FROM AN RM05/3/2 ?
 36 005306  001160                                  BNE     4$              ;BR IF NOT
 37 005310  013737  000040  001466                  MOV     @#40,XXDP       ;GET DEVICE INDICATOR AND NUMBER
 38 005316  122737  000007  001466                  CMPB    #7,XXDP         ;IS IT A VALID NUMBER ?
 39 005324  103002                                  BHIS    2$              ;YES
 40 005326  105037  001466                          CLRB    XXDP            ;NO, DEFAULT TO DRIVE 0
 41 005332  005737  000042          2$:             TST     @#42            ;CHAIN MODE OR ACT11 AUTO ACCEPT ?
 42 005336  001425                                  BEQ     3$              ;BR IF NEITHER
 43 005340  104401  005346                          TYPE    ,73$            ;;TYPE ASCIZ STRING
    005344  000412                                  BR      72$             ;;GET OVER THE ASCIZ
                                            ;;73$:  .ASCIZ  <CRLF>/NOT TESTING DRIVE /
    005372                                  72$:
 44 005372  005046                                  CLR     -(SP)           ;CLEAR WORD ON STACK
 45 005374  113716  001466                          MOVB    XXDP,(SP)       ;GET DRIVE ADDRESS
 46 005400  104403                                  TYPOS                   ;TYPE THE ADDRESS
 47 005402     001                                  .BYTE   1               ;ONLY 1 CHARACTER
 48 005403     000                                  .BYTE   0               ;SUPRESS LEADING ZEROS
 49 005404  104401  001231                          TYPE    ,$CRLF          ;CR-LF
```

```
                                                        BR      4$              ;GET NUMBER OF DRIVES
                                                3$:     INC     #-1             ;FIRST TIME THRU HERE ?
                                                        BNE     4$              ;NO
                                                        TYPE    ,75$            ;;TYPE ASCIZ STRING
                                                        BR      74$             ;;GET OVER THE ASCIZ
                                                ;;75$:  .ASCIZ  <CRLF>/TO TEST DRIVE /
                                                74$:
                                                        CLR     -(SP)           ;CLEAR WORD ON STACK
                                                        MOVB    XXDP,(SP)       ;GET DRIVE ADDRESS
                                                        TYPOS                   ;TYPE DRIVE ADDRESS
                                                        .BYTE   1               ;ONLY 1 CHARACTER
                                                        .BYTE   0               ;SUPRESS LEADING ZEROS
                                                        TYPE    ,77$            ;;TYPE ASCIZ STRING
                                                        BR      76$             ;;GET OVER THE ASCIZ
                                                ;;77$:  .ASCIZ  /, HALT PROGRAM, REMOVE RRDP PACK AND REPLACE IT/<CRLF>
                                                76$:
                                                        TYPE    ,78$            ;;TYPE ASCIZ STRING
                                                        BR      4$              ;;GET OVER THE ASCIZ
                                                ;;78$:  .ASCIZ  /WITH A WORK PACK, CLEAR LOCATION 40 AND RESTART PROGRAM./<CRLF>
                                                4$:
                                                        JSR     PC,$TKINT       ;TURN ON THE TTY KEYBOARD INTERRUPT

                                                .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
                                                        TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
                                                        BNE     79$             ;;BRANCH IF YES
                                                        CMPB    $ENV,#1         ;;ARE WE RUNNING UNDER APT?
                                                        BEQ     79$             ;;BRANCH IF YES
                                                        CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
                                                        BNE     80$             ;;BRANCH IF NO
                                                        GTSWR                   ;;GET SOFT-SWR SETTINGS
                                                        BR      80$
                                                79$:    MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
                                                80$:
                                                        INC     #-1             ;SEE IF FIRST START
                                                        BNE     SRTINT          ;BR IF NOT
                                                        JSR     PC,GETADR       ;GET OR CHECK THE RH/RM ADDRESS

                                                SRTINT: TYPE    ,$CRLF          ;CR-LF
                                                        JSR     PC,LP.AVL       ;CHECK FOR A LINE PRINTER
                                                        CLR     PS              ;ENSURE THE PRIORITY = 0
                                                        MOV     #1,$ICNT        ;SET ITERATION COUNT TO 1
                                                        JSR     PC,GETSWR       ;GO CHECK FOR CONTROL SWITCHES
                                                        JSR     PC,ST.CLK       ;INITIALIZE THE CLOCK

                                                SETVEC: JSR     PC,RMINIT       ;CHECK THE DRIVE STATUS
                                                        MOV     #-1,SAVEFG      ;SET THE SAVE REGISTERS FLAG
                                                        INC     #-1             ;FIRST TIME THRU HERE ?
                                                        BEQ     1$              ;BR IF NO
                                                        TST     CNTRLC          ;CONTROL 'C' SWITCH SET ?
                                                        BNE     SRTDRV          ;CONTINUE IF YES
                                                1$:     MOV     #PR7,PS         ;SET PRIORITY TO 7
                                                        CLR     R4              ;DRIVE TABLE POINTER
                                                        TYPE    ,UNSTAT         ;TYPE 'UNIT STATUS'
                                                2$:     TYPE    ,$CRLF          ;CR-LF
                                                        MOV     R4,-(SP)        ;;SAVE R4 FOR TYPEOUT
                                                                                ;;TYPE DRIVE NUMBER
                                                        TYPOS                   ;;GO TYPE--OCTAL ASCII
```

```
                006032    002                          .BYTE   2               ;;TYPE 2 DIGIT(S)
                006033    000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
     89 006034  104401  046655                         TYPE    ,BLNKS4         ;TYPE 4 SPACES
     90 006040  105764  036740                         TSTB    DRVSTA(R4)      ;CHECK DRIVE'S STATUS
     91 006044  100416                                 BMI     5$              ;BR IF UNSAFE
     92 006046  001020                                 BNE     6$              ;BR IF ONLINE
     93 006050  105764  036750                         TSTB    DRVTYP(R4)      ;SEE IF OFFLINE OR NONEXISTENT
     94 006054  001404                                 BEQ     3$              ;BR IF NONEXISTENT
     95 006056  100006                                 BPL     4$              ;BR IF OFFLINE
     96 006060  104401  046015                         TYPE    ,NOTRM          ;DRIVE NOT AN RM05/3/2
     97 006064  000452                                 BR      11$             ;CHECK NEXT DRIVE
     98
     99 006066  104401  045770          3$:    TYPE    ,NOTPRS         ;DRIVE NOT PRESENT
    100 006072  000447                         BR      11$             ;CHECK NEXT DRIVE
    101
    102 006074  104401  045747          4$:    TYPE    ,UNTOFF         ;DRIVE OFFLINE
    103 006100  000416                         BR      8$              ;PRINT DRIVE TYPE
    104
    105 006102  104401  046005          5$:    TYPE    ,NOTSAF         ;DRIVE UNSAFE
    106 006106  000413                         BR      8$              ;PRINT DRIVE TYPE
    107
    108 006110  005737  001466          6$:    TST     XXDP            ;LOADED FROM THIS DEVICE ?
    109 006114  001406                         BEQ     7$              ;BR IF NO
    110 006116  123704  001466                 CMPB    XXDP,R4         ;LOADED FROM THIS DRIVE ?
    111 006122  001003                         BNE     7$              ;BR IF NO
    112 006124  104401  046036                 TYPE    ,LODEV          ;DRIVE IS LOAD DEVICE
    113 006130  000430                         BR      11$
    114 006132  104401  045760          7$:    TYPE    ,UNTON          ;DRIVE ONLINE
    115 006136  104401  046657          8$:    TYPE    ,BLNKS2         ;TYPE 2 SPACES
    116 006142  005000                         CLR     R0
    117 006144  116400  036750                 MOVB    DRVTYP(R4),R0   ;GET DRIVE TYPE
    118 006150  012737  046063  006210         MOV     #$RM03,10$      ;ASSUME ADDRESS OF RM03 MESSAGE
    119 006156  122700  000004                 CMPB    #4,R0           ;IS DEVICE AN RM03 ?
    120 006162  001411                         BEQ     9$              ;TYPE IT IF YES
    121 006164  012737  046056  006210         MOV     #$RM02,10$      ;ADDRESS OF RM02 MESSAGE
    122 006172  122700  000005                 CMPB    #5,R0           ;IS DEVICE AN RM02 ?
    123 006176  001403                         BEQ     9$              ;BR IF YES
    124 006200  012737  046070  006210         MOV     #$RM05,10$      ;ADDRESS OF RM05 MESSAGE
    125
    126 006206  104401                  9$:    TYPE                    ;TYPE THE DRIVE TYPE MESSAGE
    127 006210  000000                  10$:   .WORD   0               ;MESSAGE ADDRESS HERE
    128
    129 006212  005204                  11$:   INC     R4              ;INCREMENT DRIVE NUMBER/TABLE POINTER
    130 006214  020427  000010                 CMP     R4,#8.          ;FINISHED ?
    131 006220  001300                         BNE     2$              ;BR IF NOT
    132 006222  104401  001231                 TYPE    ,$CRLF          ;CR-LF
    133
    134 006226  005037  177776                 CLR     PS              ;SET PRIORITY BACK TO '0'
    135
    136 006232  005737  001320          SRTDRV: TST    CNTRLC          ;CONTROL "C" START/RESTART?
    137 006236  001417                         BEQ     2$              ;NO--BRANCH
    138 006240  013746  001316                 MOV     SAVCSW,-(SP)    ;GET THE PREVIOUS 'C.SWR' CONTENTS
    139 006244  063716  001314                 ADD     C.SWR,(SP)      ;SET UP TO SEE IF 'BIT00' IS DIFFERENT
    140 006250  032726  000001                 BIT     #BIT00,(SP)+    ;IS 'BIT00' DIFFERENT ?
    141 006254  001405                         BEQ     1$              ;BR IF NOT
    142 006256  013737  001314  001316         MOV     C.SWR,SAVCSW    ;STORE PRESENT 'C.SWR' VALUE
    143 006264  004737  026434                 JSR     PC,LODFLT       ;RESET PARAMETERS TO THEIR DEFAULT VALUES
```

```
                                1$:     JSR     PC,GT.PRM       ;GET PARAMETERS
                                        BR      6$
                                2$:     JSR     PC,LODFLT       ;SETUP DEFAULT PARAMETERS
                                        CLR     DRVSEL          ;NO DRIVES SELECTED
                                        CLR     R0              ;DETERMINE THE DRIVES THAT
                                        MOV     #1,R1           ;ARE AVAILABLE FOR TESTING

                                3$:     TSTB    DRVSTA(R0)      ;IS DRIVE ON-LINE ?
                                        BLE     5$              ;BR IF NO
                                        TST     XXDP            ;LOADED FROM THIS DEVICE ?
                                        BEQ     4$              ;BR IF NO
                                        CMPB    XXDP,R0         ;LOADED FROM THIS DRIVE ?
                                        BEQ     5$              ;BR IF YES
                                4$:     BISB    ATABIT(R0),DRVSEL       ;YES, SELECT DRIVE FOR TESTING
                                5$:     INC     R0              ;TRY NEXT DRIVE
                                        ASLB    R1              ;ANY MORE DRIVES TO CHECK ?
                                        BNE     3$              ;BR IF YES

                                6$:     CLR     SEEKFG          ;CLEAR SEEK FLAG
                                        BIT     #SW08,C.SWR     ;DO SEEK BEFORE DATA TRANSFER?
                                        BNE     7$              ;YES--BRANCH
                                        COM     SEEKFG          ;NO
                                7$:
                                        TYPE    ,DRIVES         ;'DRIVES(S) TO BE TESTED'
                                        CLR     $ENDCT          ;DETERMINE PASSES TO MAKE AND
                                        CLR     R0              ;THE DRIVES TO BE TESTED
                                        MOV     DRVSEL,R1       ;ANY DRIVES SELECTED?
                                        BNE     8$              ;YES--BRANCH
                                        TYPE    ,NONE           ;'NONE'
                                        JMP     $EOP            ;JUMP TO END OF PROGRAM

                                8$:     ASR     R1              ;REPORT THE DRIVES TO BE TESTED
                                        BCC     9$
                                        INC     $ENDCT          ;GIVE THIS DRIVE A PASS
                                        MOV     R0,-(SP)        ;;SAVE R0 FOR TYPEOUT
                                        TYPOS                   ;;GO TYPE--OCTAL ASCII
                                        .BYTE   1               ;;TYPE 1 DIGIT(S)
                                        .BYTE   0               ;;SUPPRESS LEADING ZEROS
                                        TST     R1              ;MORE DRIVES?
                                        BEQ     10$             ;NO--BRANCH
                                        TYPE    ,COMMA          ;','
                                9$:     INC     R0              ;FORM DRIVE NUMBER
                                        BR      8$

                                10$:    MOV     $ENDCT,$EOPCT
                                        TST     CLKSTA          ;IS KW11-P AVAILABLE ?
                                        BGT     RSTRT1          ;BR IF YES
                                        BIT     #36000,TSTNMS   ;ANY TIMING TESTS TO BE PERFORMED ?
                                        BEQ     RSTRT1          ;BR IF NO
                                        TYPE    ,NOCLOK         ;TYPE NO KW11-P CLOCK MESSAGE

                                RSTRT1: TST     DRVSEL          ;ANY DRIVES SELECTED ?
                                        BNE     1$              ;BR IF YES
                                        JMP     START2          ;GET DRIVE SELECTION ENTRY
                                                                ;& RESTART AT BEGINNING
                                1$:     CLR     CHKDRV          ;INIT. THE CHECK DRIVE KEY
                                        MOV     #1,DRVMSK       ;START TO CHECK DESIRED DRIVES
```

```
006530  033737  001352  001326  RSTRT2: BIT     DRVMSK,DRVSEL     ;IS THIS DRIVE SELECTED?
006536  001010                          BNE     DRVOK             ;YES--GO CHECK IF DRIVE IS READY FOR TESTING

006540  012706  001100          RESTART: MOV    #STACK,SP         ;SETUP THE STACK POINTER
006544  005237  001350                  INC     CHKDRV            ;MOVE  TO NEXT DRIVE NUMBER
006550  106337  001352                  ASLB    DRVMSK            ;POSITION THE MASK
006554  103753                          BCS     RSTRT1            ;BRANCH IF THE DRIVE NUMBER NEEDS INITIALIZED
006556  000764                          BR      RSTRT2

006560  013702  001350          DRVOK:  MOV     CHKDRV,R2         ;PICKUP THE DRIVE NUMBER
006564  105762  036740                  TSTB    DRVSTA(R2)        ;IS DESIRED DRIVE ON-LINE?
006570  003005                          BGT     1$                ;YES, BRANCH
006572  104011                          EMT     11
006574  043737  001352  001326          BIC     DRVMSK,DRVSEL     ;CLEAR DRIVE'S SELECTION BIT
006602  000756                          BR      RESTART           ;RETURN
006604  010237  045456          1$:     MOV     R2,DPB.A          ;SET THE DRIVE NUMBER INTO THE DPB'S
006610  010237  045476                  MOV     R2,DPB.B
006614  010237  045516                  MOV     R2,DPB.C
006620  010237  045536                  MOV     R2,DTADPB
006624  004737  027066                  JSR     PC,LDCMD          ;LOAD COMMAND INTO DPB.B AND DPB.C
006630  012737  021020  001346          MOV     #$EOP,BYPASS      ;IF ERROR GO TO END OF PROGRAM
006636  112737  000020  045457          MOVB    #20,DPB.A+1       ;ASSUME 16 BIT FORMAT
006644  032737  000001  001314          BIT     #BIT00,C.SWR      ;16 BIT FORMAT REQUESTED ?
006652  001402                          BEQ     2$                ;BR IF YES
006654  105037  045457                  CLRB    DPB.A+1           ;CLEAR THE 'FMT16' BIT
006660  112737  000143  045460  2$:     MOVB    #SETFORM,DPB.A+2          ;SET THE FORMAT BIT PER DPB.A+1
006666  004037  027132                  JSR     R0,CALL.A         ;GO EXECUTE THE COMMAND
006672  004737  026072                  JSR     PC,CNTCLR         ;GO CLEAR CONTROLLER
006676  112737  000107  045460          MOVB    #RECAL,DPB.A+2    ;RECAL=COMMAND
006704  004037  027132                  JSR     R0,CALL.A         ;GO EXECUTE THE COMMAND
006710  104401  046222                  TYPE    ,TSTDRV           ;'TESTING DRIVE '
006714  010246                          MOV     R2,-(SP)          ;;SAVE R2 FOR TYPEOUT
006716  104403                          TYPOS                     ;;GO TYPE--OCTAL ASCII
006720     002                          .BYTE   2                 ;;TYPE 2 DIGIT(S)
006721     000                          .BYTE   0                 ;;SUPPRESS LEADING ZEROS
006722  104401  046132                  TYPE    ,COMMA            ;TYPE ', '
006726  104401  046242                  TYPE    ,SERIAL           ;TYPE 'MBA SN# '
006732  012700  000004                  MOV     #4,R0             ;FOUR DIGITS TO TYPE
006736  013701  045606                  MOV     RM.REG+30,R1      ;SERIAL NUMBER
006742  005002                  3$:     CLR     R2                ;ZERO
006744  006101                          ROL     R1                ;PUT THE NEXT DIGIT
006746  006102                          ROL     R2                ;INTO R2
006750  006101                          ROL     R1
006752  006102                          ROL     R2
006754  006101                          ROL     R1
006756  006102                          ROL     R2
006760  006101                          ROL     R1
006762  006102                          ROL     R2
006764  062702  000060                  ADD     #'0,R2            ;MAKE IT ASCII
006770  010227                          MOV     R2,(PC)+          ;SAVE IT
006772  000000                  4$:     .WORD   0
006774  104401  006772                  TYPE    ,4$               ;TYPE
007000  005300                          DEC     R0                ;ALL DIGITS TYPED?
007002  003357                          BGT     3$                ;NO -- BRANCH
007004  104401  001231                  TYPE    ,$CRLF            ;CR-LF
007010  113737  001462  001131          MOVB    ERR.CT,$ERMAX     ;SETUP MAX ERROR COUNT
```

```
                                                        B 6
    252
```

```
                        ;;//////////////////////////////////////////////////////////////
                        ;*IN THE DESCRIPTIONS OF THE BELOW TESTS THE VARIABLES USED
                        ;*AND THEIR DEFAULT VALUES (UNLESS SPECIFIED OTHERWISE) ARE:
                        ;*
                        ;*MNEMONIC      VALUE           VARIABLE
                        ;*--------      -----           --------
                        ;*
                        ;*R             1               ITERATIONS (REPEATS)
                        ;*FC            0               FIRST CYLINDER ADDRESS
                        ;*LC            822             LAST CYLINDER ADDRESS
                        ;*IC            1               INCREMENT VALUE
                        ;*NC OF NC1     FC+IC           NEW OR MODIFIED CYLINDER
                        ;*                              ADDRRESS
                        ;*NC2           LC-IC           NEW OR MODIFIED CYLINDER
                        ;*                              ADDRESS
                        ;*
                        ;*FT            0               FIRST TRACK ADDRESS
                        ;*LT            4 OR 18         LAST TRACK ADDRESS
                        ;*IT            1               INCREMENT VALUE
                        ;*NT            FT+IT           NEW OR MODIFIED TRACK ADDRESS
                        ;*
                        ;*FS            0               FIRST SECTOR ADDRESS
                        ;*LS            31              LAST SECTOR ADDRESS
                        ;*
                        ;;//////////////////////////////////////////////////////////////

                        .SBTTL  SEEK TESTS

                        ;;//////////////////////////////////////////////////////////////
                        ;*THE SEEK TESTS WILL BE EXCUTED USING IMPLIED SEEKS. THESE
                        ;*IMPLIED SEEKS WILL BE PERFORMED BY "READ HEADER AND
                        ;*DATA" COMMANDS TO TRACK "FT" SECTOR "FS" OF THE DESIRED CYLINDER.
                        ;*THE WORD COUNT WILL BE SET SUCH THAT ONLY THE CYLINDER AND
                        ;*TRACK/SECTOR WORDS OF THE HEADER ARE READ.
                        ;;//////////////////////////////////////////////////////////////


                        ;;**************************************************************
                        ;*TEST 0         RECAL/SEEK TEST
                        ;*THIS TEST WILL CAUSE THE DRIVE TO EXECUTE A RECALIBRATE
                        ;*COMMAND CYCLE AND THEN SEEK FORWARD TO CYLINDER "LC" AT
                        ;*THE COMPLETION OF BOTH COMMANDS STATUS INDICATIONS ARE
                        ;*CHECKED TO ENSURE NO ERRORS OCCURRED.
                        ;;**************************************************************
007016                  TST0:
007016  000240                    NOP
007020  033737  001526  001330    BIT     BITS+<0*2>,TSTNMS       ;DO THIS TEST?
007026  001002                    BNE     .+6                     ;BR IF YES
007030  000137  007204            JMP     TST1                    ;NO--JUMP TO TEST1

007034  012737  000000  001116    MOV     #0,$TSTNM               ;SET TEST #0 AND CLEAR ($ERFLG)
007042  004737  026672            JSR     PC,LODPRM               ;LOAD THE PARMETERS FOR THE TEST
007046  012737  007166  001124    MOV     #TEST0,$LPERR           ;SETUP THE LOOP ON ERROR ADDRESS
007054  013737  002324  001220    MOV     RPT,$TIMES              ;GET THE ITERATION COUNT
007062  112737  000031  001131    MOVB    #25.,$ERMAX             ;MAX ERRORS ALLOWED FOR TEST
007070  012737  000000  001240    MOV     #0,$TESTN               ;;SET TEST NUMBER IN APT MAIL BOX
```

```
        007076  032777  010000  172050          BIT     #SW12,@SWR      ;INHIBIT TYPING TEST NUMBER ?
        007104  001406                          BEQ     .+16            ;BR IF YES
        007106  104401  046253                  TYPE    ,MSGTST         ;TYPE 'TEST'
        007112  013746  001240                  MOV     $TESTN,-(SP)    ;;SAVE $TESTN FOR TYPEOUT
        007116  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
        007120     002                          .BYTE   2               ;;TYPE 2 DIGIT(S)
        007121     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS

    46  007122  112737  000107  045460          MOVB    #RECAL,DPB.A+2  ;RECAL=COMMAND
    47  007130  113737  002342  045506          MOVB    FS,DPB.B+10     ;FS
    48  007136  113737  002334  045507          MOVB    FT,DPB.B+11     ;FT
    49  007144  013737  002330  045510          MOV     LC,DPB.B+12     ;LC
    58  007152  012737  007202  001346          MOV     #EXIT0,BYPASS   ;GO TO EXIT0 ON ERROR
        007160  012737  007166  001122          MOV     #TEST0,$LPADR   ;SETUP LOOP ADDRESS
        007166                          TEST0:
        007166  012706  001100                  MOV     #STACK,SP       ;SET UP STACK POINTER
        007172  004037  027132                  JSR     R0,CALL.A       ;GO EXECUTE THE COMMAND
        007176  004037  027264                  JSR     R0,CALL.B       ;GO EXECUTE THE COMMAND
        007202  000004                  EXIT0:  SCOPE                   ;CALL SCOPE ROUTINE
    59
    66
    67                                  ;;*****************************************************************
                                        ;*TEST 1        SEEK/SEEK TEST
                                        ;*THIS TEST WILL CAUSE THE DRIVE TO EXECUTE A FORWARD SEEK
                                        ;*CYCLE TO "LC", "LT", "LS" FOLLOWED BY A REVERSE SEEK CYCLE TO
                                        ;*"FC", "FT", "FS".  AT THE COMPLETION OF EACH SEEK, THE PROPER
                                        ;*INDICATORS ARE EXAMINED TO ENSURE PROPER OPERATION.
                                        ;;*****************************************************************
        007204                          TST1:
        007204  000240                          NOP
        007206  033737  001530  001330          BIT     BITS+<1*2>,TSTNMS       ;DO THIS TEST?
        007214  001002                          BNE     .+6             ;BR IF YES
        007216  000137  007412                  JMP     TST2            ;NO--JUMP TO TEST2

        007222  012737  000001  001116          MOV     #1,$TSTNM       ;SET TEST #1 AND CLEAR ($ERFLG)
        007230  004737  026672                  JSR     PC,LODPRM       ;LOAD THE PARMETERS FOR THE TEST
        007234  012737  007374  001124          MOV     #TEST1,$LPERR   ;SETUP THE LOOP ON ERROR ADDRESS
        007242  013737  002324  001220          MOV     RPT,$TIMES      ;GET THE ITERATION COUNT
        007250  112737  000031  001131          MOVB    #25.,$ERMAX     ;MAX ERRORS ALLOWED FOR TEST
        007256  012737  000001  001240          MOV     #1,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
    68
        007264  032777  010000  171662          BIT     #SW12,@SWR      ;INHIBIT TYPING TEST NUMBER ?
        007272  001406                          BEQ     .+16            ;BR IF YES
        007274  104401  046253                  TYPE    ,MSGTST         ;TYPE 'TEST'
        007300  013746  001240                  MOV     $TESTN,-(SP)    ;;SAVE $TESTN FOR TYPEOUT
        007304  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
        007306     002                          .BYTE   2               ;;TYPE 2 DIGIT(S)
        007307     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS

    69  007310  005037  001464                  CLR     BASFLG          ;CLEAR BAD SECTOR ENCOUNTER FOR THE DRIVE
    70  007314  113737  002342  045506          MOVB    FS,DPB.B+10     ;FS
    71  007322  113737  002344  045526          MOVB    LS,DPB.C+10     ;LS
    72  007330  113737  002334  045507          MOVB    FT,DPB.B+11     ;FT
    73  007336  113737  002336  045527          MOVB    LT,DPB.C+11     ;LT
    74  007344  013737  002326  045510          MOV     FC,DPB.B+12     ;FC
    75  007352  013737  002330  045530          MOV     LC,DPB.C+12     ;LC
    80  007360  012737  007410  001346          MOV     #EXIT1,BYPASS   ;GO TO EXIT1 ON ERROR
```

```
                                                    E 6
007366 012737 007374 001122          MOV     #TEST1,$LPADR   ;SETUP LOOP ADDRESS
007374                       TEST1:
 81 007374 012706 001100             MOV     #STACK,SP       ;SET THE STACK POINTER
 82 007400 004037 027466             JSR     R0,CALL.C       ;GO EXECUTE THE COMMAND
 83 007404 004037 027264             JSR     R0,CALL.B       ;GO EXECUTE THE COMMAND
 84 007410 000004             EXIT1: SCOPE                   ;CALL SCOPE ROUTINE
 85
 96
 97                                  ;;*****************************************************************
                                     ;*TEST 2        INCREMENT/SEEK TEST
                                     ;*THIS TEST WILL COMMAND FORWARD SEEK CYCLES TO ADVANCE THE
                                     ;*CYLINDER ADDRESS FROM "FC" TO "LC" BY THE INCREMENT "IC".
                                     ;*WHEN THE RESULTANT CYLINDER ADDRESS (NC) EXCEEDS
                                     ;*"LC" REVERSE SEEK CYCLES ARE INITIATED; STARTING
                                     ;*AT THE LAST LEGAL "NC" AND DECREMENTING BY "IC"
                                     ;*UNTIL "NC" IS LESS THAN "FC". AT THE COMPLETION OF EACH
                                     ;*SEEK COMMAND THE PROPER INDICATORS ARE EXAMINED TO
                                     ;*ENSURE PROPER OPERATION.
                                     ;;*****************************************************************
007412                       TST2:
007412 000240                        NOP
007414 033737 001532 001330          BIT     BITS+<2*2>,TSTNMS       ;DO THIS TEST?
007422 001002                        BNE     .+6             ;BR IF YES
007424 000137 007654                 JMP     TST3            ;NO--JUMP TO TEST3

007430 012737 000002 001116          MOV     #2,$TSTNM       ;SET TEST #2 AND CLEAR ($ERFLG)
007436 004737 026672                 JSR     PC,LODPRM       ;LOAD THE PARMETERS FOR THE TEST
007442 012737 007546 001124          MOV     #TEST2,$LPERR   ;SETUP THE LOOP ON ERROR ADDRESS
007450 013737 002324 001220          MOV     RPT,$TIMES      ;GET THE ITERATION COUNT
007456 112737 000031 001131          MOVB    #25.,$ERMAX     ;MAX ERRORS ALLOWED FOR TEST
007464 012737 000002 001240          MOV     #2,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
 98
007472 032777 010000 171454          BIT     #SW12,@SWR      ;INHIBIT TYPING TEST NUMBER ?
007500 001406                        BEQ     .+16            ;BR IF YES
007502 104401 046253                 TYPE    ,MSGTST         ;TYPE 'TEST'
007506 013746 001240                 MOV     $TESTN,-(SP)    ;;SAVE $TESTN FOR TYPEOUT
007512 104403                        TYPOS                   ;;GO TYPE--OCTAL ASCII
007514    002                        .BYTE   2               ;;TYPE 2 DIGIT(S)
007515    000                        .BYTE   0               ;;SUPPRESS LEADING ZEROS

 99 007516 012737 007524 001122      MOV     #1$,$LPADR      ;SETUP LOOP ADDRESS
100 007524 113737 002342 045506 1$:  MOVB    FS,DPB.B+10     ;FS
101 007532 113737 002334 045507      MOVB    FT,DPB.B+11     ;FT
105 007540 012737 007652 001346      MOV     #EXIT2,BYPASS   ;GO TO EXIT2 ON ERROR
007546                       TEST2:
106 007546 013737 002326 045510      MOV     FC,DPB.B+12     ;FC
107 007554 012737 007554 001124      MOV     #.,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
007562 012706 001100                 MOV     #STACK,SP       ;LOAD THE STACK POINTER
108 007566                   INCSK:
109 007566 004037 027264             JSR     R0,CALL.B       ;GO EXECUTE THE COMMAND
110 007572 063737 002332 045510      ADD     IC,DPB.B+12     ;MOVE TO NEXT CYLINDER
111 007600 023737 002330 045510      CMP     LC,DPB.B+12     ;OUT OF CYLINDERS?
112 007606 002367                    BGE     INCSK           ;NO--BRANCH
113 007610 013737 002330 045510      MOV     LC,DPB.B+12
114 007616 012737 007616 001124      MOV     #.,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
007624 012706 001100                 MOV     #STACK,SP       ;LOAD THE STACK POINTER
115 007630                   DECSK:
```

```
                                                    ;GO EXECUTE THE COMMAND
                                                    ;CALL SCOPE ROUTINE
```

```
                        ;;*****************************************************************
                        ;*TEST 3        STEPPING SEEK TEST
                        ;*THIS TEST WILL COMMAND SEEK CYCLES TO CYLINDER 0, 1, 2, 4,
                        ;*8, 16, 32, 64, 128, 256 AND 512. AT THE COMPLETION OF EACH
                        ;*SEEK COMMAND THE PROPER INDICATORS ARE EXAMINED TO ENSURE
                        ;*PROPER OPERATION.
                        ;;*****************************************************************
```

```
                                                    ;DO THIS TEST?
                                                    ;BR IF YES
                                                    ;NO--JUMP TO TEST4

                                                    ;SET TEST #3 AND CLEAR ($ERFLG)
                                                    ;LOAD THE PARMETERS FOR THE TEST
                                                    ;SETUP THE LOOP ON ERROR ADDRESS
                                                    ;GET THE ITERATION COUNT
                                                    ;MAX ERRORS ALLOWED FOR TEST
                                                    ;;SET TEST NUMBER IN APT MAIL BOX
```

```
                                                    ;INHIBIT TYPING TEST NUMBER ?
                                                    ;BR IF YES
                                                    ;TYPE 'TEST'
                                                    ;;SAVE $TESTN FOR TYPEOUT
                                                    ;;GO TYPE--OCTAL ASCII
                                                    ;;TYPE 2 DIGIT(S)
                                                    ;;SUPPRESS LEADING ZEROS
```

```
                                                    ;SETUP TEST LOOP ADDRESS
                                                    ;FS
                                                    ;FT
                                                    ;GO TO BYPASS ON ERROR
                                                    ;FC
                                                    ;SETUP THE ERROR LOOP ADDRESS
                                                    ;LOAD THE STACK POINTER
                                                    ;GO EXECUTE THE COMMAND
                                                    ;CYLINDER 1
                                                    ;SETUP THE ERROR LOOP ADDRESS
                                                    ;LOAD THE STACK POINTER
                                                    ;DESIRED CYLINDER
                                                    ;GO EXECUTE THE COMMAND
                                                    ;MOVE TO NEXT CYLINDER
                                                    ;DONE?
                                                    ;NO--LOOP
                                                    ;CALL SCOPE ROUTINE
```

```
                        ;;*****************************************************************
                        ;*TEST 4        OSCILLATING SEEK TEST
```

```
                        ;*THIS TEST WILL COMMAND SEEK CYCLES FROM "FC" TO "NC" AND BACK
                        ;*TO "FC".  "NC" STARTS AT "FC" AND INCREMENTS BY "IC" UP TO CYLINDER
                        ;*"LC", THEN IS DECREMENTED BY "IC" BACK TO CYLINDER "FC".  AT THE
                        ;*COMPLETION OF EVERY SEEK COMMAND THE PROPER INDICATORS ARE
                        ;*EXAMINED TO ENSURE PROPER OPERATION.
                        ;;*****************************************************************
010074                  TST4:
010074  000240                  NOP
010076  033737  001536  001330  BIT     BITS+<4*2>,TSTNMS       ;DO THIS TEST?
010104  001002                  BNE     .+6             ;BR IF YES
010106  000137  010506          JMP     TST5            ;NO--JUMP TO TEST5

010112  012737  000004  001116  MOV     #4,$TSTNM       ;SET TEST #4 AND CLEAR ($ERFLG)
010120  004737  026672          JSR     PC,LODPRM       ;LOAD THE PARMETERS FOR THE TEST
010124  012737  010244  001124  MOV     #TEST4,$LPERR   ;SETUP THE LOOP ON ERROR ADDRESS
010132  013737  002324  001220  MOV     RPT,$TIMES      ;GET THE ITERATION COUNT
010140  112737  000031  001131  MOVB    #25.,$ERMAX     ;MAX ERRORS ALLOWED FOR TEST
010146  012737  000004  001240  MOV     #4,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
159
010154  032777  010000  170772  BIT     #SW12,@SWR      ;INHIBIT TYPING TEST NUMBER ?
010162  001406                  BEQ     .+16            ;BR IF YES
010164  104401  046253          TYPE    ,MSGTST         ;TYPE 'TEST'
010170  013746  001240          MOV     $TESTN,-(SP)    ;;SAVE $TESTN FOR TYPEOUT
010174  104403                  TYPOS                   ;;GO TYPE--OCTAL ASCII
010176     002                  .BYTE   2               ;;TYPE 2 DIGIT(S)
010177     000                  .BYTE   0               ;;SUPPRESS LEADING ZEROS

160 010200  012737  010206  001122          MOV     #1$,$LPADR      ;SETUP LOOP ADDRESS
161 010206  113737  002342  045506  1$:     MOVB    FS,DPB.B+10     ;FS
162 010214  113737  002334  045507          MOVB    FT,DPB.B+11     ;FT
170 010222  012737  010504  001346          MOV     #EXIT4,BYPASS   ;GO TO EXIT4 ON ERROR
    010230  005002                  CLR     R2              ;CLEAR STALL SWITCH (NO STALL)
    010232  032737  010000  001314  BIT     #SW12,C.SWR     ;STALL REQUIRED?
    010240  001401                  BEQ     TEST4           ;NO--BRANCH
    010242  005102                  COM     R2              ;YES--SET SWITCH
    010244                  TEST4:
171 010244  013701  002326          MOV     FC,R1           ;SET NC TO FC
172 010250  005037  001434          CLR     STALL0  ;START AT ZERO IF STALLS REQUIRED
173 010254  012737  010254  001124  MOV     #.,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
    010262  012706  001100          MOV     #STACK,SP       ;LOAD THE STACK POINTER
174 010266  010137  045510  1$:     MOV     R1,DPB.B+12     ;NC
175 010272  004037  027264          JSR     R0,CALL.B       ;GO EXECUTE THE COMMAND
176 010276  005702                  TST     R2              ;STALL?
17? 010300  001403                  BEQ     2$              ;NO--BRANCH
178 010302  004037  030664          JSR     R0,STALL        ;YES-GO TO STALL ROUTINE
179 010306  001434                  .WORD   STALL0          ;TIME POINTER
180 010310  013737  002326  045510  2$:     MOV     FC,DPB.B+12     ;FC
181 010316  004037  027264          JSR     R0,CALL.B       ;GO EXECUTE THE COMMAND
182 010322  005702                  TST     R2              ;STALL?
183 010324  001413                  BEQ     3$              ;NO--BRANCH
184 010326  004037  030664          JSR     R0,STALL        ;YES--GO TO STALL ROUTINE
185 010332  001434                  .WORD   STALL0          ;TIME POINTER
186 010334  005237  001434          INC     STALL0  ;UPDATE THE TIME
187 010340  023737  001460  001434  CMP     MXSTAL,STALL0   ;TIME TO BIG?
188 010346  003347                  BGT     1$              ;NO--BRANCH
189 010350  005037  001434          CLR     STALL0  ;YES--START OVER AT ZERO
190 010354  063701  002332  3$:     ADD     IC,R1           ;MOVE TO NEXT CYLINDER
```

```
                                        CMP     R1,LC           ;LAST CYLINDER COMPLETED?
                                        BLE     1$              ;NO--BRANCH
                                        MOV     LC,R1           ;SET NC TO LC
                                        MOV     #.,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
                                        MOV     #STACK,SP       ;LOAD THE STACK POINTER
                                4$:     MOV     R1,DPB.B+12     ;NC
                                        JSR     R0,CALL.B       ;GO EXECUTE THE COMMAND
                                        TST     R2              ;STALL?
                                        BEQ     5$              ;NO--BRANCH
                                        JSR     R0,STALL        ;YES--GO TO STALL ROUTINE
                                        .WORD   STALL0          ;TIME POINTER
                                5$:     MOV     LC,DPB.B+12     ;LC
                                        JSR     R0,CALL.B       ;GO EXECUTE THE COMMAND
                                        TST     R2              ;STALL?
                                        BEQ     6$              ;NO--BRANCH
                                        JSR     R0,STALL        ;YES--GO TO STALL ROUTINE
                                        .WORD   STALL0          ;TIME POINTER
                                        INC     STALL0  ;UPDATE STALL TIME
                                        CMP     MXSTAL,STALL0   ;TIME TOO BIG?
                                        BGT     4$              ;NO--BRANCH
                                        CLR     STALL0  ;YES--SET STALL TIME BACK TO ZERO
                                6$:     SUB     IC,R1           ;NEXT CYLINDER
                                        CMP     R1,FC           ;DONE?
                                        BGE     4$              ;NO--BRANCH
                                EXIT4:  SCOPE                   ;CALL SCOPE ROUTINE

                                ;;*****************************************************************
                                ;*TEST 5         CONVERGING/DIVERGING SEEK TEST
                                ;*THIS TEST WILL CAUSE THE DRIVE TO EXECUTE FORWARD AND REVERSE
                                ;*SEEKS FROM "NC1" AND "NC2" RESPECTIVELY, "NC1" WILL BE INCREMENTED
                                ;*BY "IC" AND "NC2" WILL BE DECREMENTED BY "IC" UNTIL "NC1" IS
                                ;*GREATER THAN THE INITIAL VALUE OF "NC2" AND "NC2" IS
                                ;*LESS THAN THE INITIAL VALUE OF "NC1".  AT THE COMPLETION OF
                                ;*EACH SEEK COMMAND THE PROPER INDICATORS ARE EXAMINED TO
                                ;*ENSURE PROPER OPERATION.  "NC1" AND "NC2" DEFAULT TO
                                ;*"FC" AND "LC" RESPECTIVELY.
                                ;;*****************************************************************
                                TST5:
                                        NOP
                                        BIT     BITS+<5*2>,TSTNMS       ;DO THIS TEST?
                                        BNE     .+6             ;BR IF YES
                                        JMP     TST6            ;NO--JUMP TO TEST6

                                        MOV     #5,$TSTNM       ;SET TEST #5 AND CLEAR ($ERFLG)
                                        JSR     PC,LODPRM       ;LOAD THE PARMETERS FOR THE TEST
                                        MOV     #TEST5,$LPERR   ;SETUP THE LOOP ON ERROR ADDRESS
                                        MOV     RPT,$TIMES      ;GET THE ITERATION COUNT
                                        MOVB    #25.,$ERMAX     ;MAX ERRORS ALLOWED FOR TEST
                                        MOV     #5,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX

                                        BIT     #SW12,@SWR      ;INHIBIT TYPING TEST NUMBER ?
                                        BEQ     .+16            ;BR IF YES
                                        TYPE    ,MSGTST         ;TYPE 'TEST'
                                        MOV     $TESTN,-(SP)    ;;SAVE $TESTN FOR TYPEOUT
                                        TYPOS                   ;;GO TYPE--OCTAL ASCII
                                        .BYTE   2               ;;TYPE 2 DIGIT(S)
```

```
     010611     000                                      .BYTE   0                       ;;SUPPRESS LEADING ZEROS

     010612  012737  010620  001122                      MOV     #1$,$LPADR              ;SETUP LOOP ADDRESS
     010620  113737  002342  045506   1$:                MOVB    FS,DPB.B+10             ;FS
     010626  113737  002334  045507                      MOVB    FT,DPB.B+11             ;FT
     010634  012737  010730  001346                      MOV     #EXIT5,BYPASS           ;GO TO EXIT5 ON ERROR
     010642                                    TEST5:
     010642  013701  002326                              MOV     FC,R1                   ;START NC1 AT FC
     010646  013702  002330                              MOV     LC,R2                   ;START NC2 AT LC
     010652  012737  010652  001124                      MOV     #.,$LPERR               ;SETUP THE ERROR LOOP ADDRESS
     010660  012706  001100                              MOV     #STACK,SP               ;LOAD THE STACK POINTER
     010664  010137  045510           1$:                MOV     R1,DPB.B+12             ;NC1
     010670  004037  027264                              JSR     R0,CALL.B               ;GO EXECUTE THE COMMAND
     010674  010237  045510                              MOV     R2,DPB.B+12             ;NC2
     010700  004037  027264                              JSR     R0,CALL.B               ;GO EXECUTE THE COMMAND
     010704  063701  002332                              ADD     IC,R1                   ;NEXT NC1
     010710  163702  002332                              SUB     IC,R2                   ;NEXT NC2
     010714  020137  002330                              CMP     R1,LC                   ;DONE?
     010720  003003                                      BGT     EXIT5                   ;YES--BRANCH
     010722  020237  002326                              CMP     R2,FC   ;?
     010726  002356                                      BGE     1$                      ;NO--BRANCH
     010730  000004                            EXIT5:    SCOPE                           ;CALL SCOPE ROUTINE



                                               ;;*******************************************************************
                                               ;*TEST 6          SERVO ADDRESSING LOGIC NOISE GENERATOR
                                               ;*IN THIS TEST A SEEK IS DONE TO CYL "NC" THEN A SEEK TO
                                               ;*NC+4 THEN NC+1 THEN NC+3 THEN NC+2 THEN NC+5.  NOW "NC" IS UPDATED
                                               ;*BY "IC" AND THE ABOVE SEQUENCE IS REPEATED UNITL "LC" IS
                                               ;*EXCEEDED BY ANY OF THE ABOVE VALUES.  THE INITIAL VALUE OF "NC"
                                               ;*IS "FC".  AT THE COMPLETION OF EACH SEEK COMMAND THE
                                               ;*PROPER INDICATORS ARE EXAMINED TO ENSURE PROPER OPERATION.
                                               ;;*******************************************************************
     010732                                    TST6:
     010732  000240                                      NOP
     010734  033737  001542  001330                      BIT     BITS+<6*2>,TSTNMS       ;DO THIS TEST?
     010742  001002                                      BNE     .+6                     ;BR IF YES
     010744  000137  011222                              JMP     TST7                    ;NO--JUMP TO TEST7

     010750  012737  000006  001116                      MOV     #6,$TSTNM               ;SET TEST #6 AND CLEAR ($ERFLG)
     010756  004737  026672                              JSR     PC,LODPRM               ;LOAD THE PARMETERS FOR THE TEST
     010762  012737  011066  001124                      MOV     #TEST6,$LPERR           ;SETUP THE LOOP ON ERROR ADDRESS
     010770  013737  002324  001220                      MOV     RPT,$TIMES              ;GET THE ITERATION COUNT
     010776  112737  000031  001131                      MOVB    #25.,$ERMAX             ;MAX ERRORS ALLOWED FOR TEST
     011004  012737  000006  001240                      MOV     #6,$TESTN               ;;SET TEST NUMBER IN APT MAIL BOX

     011012  032777  010000  170134                      BIT     #SW12,@SWR              ;INHIBIT TYPING TEST NUMBER ?
     011020  001406                                      BEQ     .+16                    ;BR IF YES
     011022  104401  046253                              TYPE    ,MSGTST                 ;TYPE 'TEST'
     011026  013746  001240                              MOV     $TESTN,-(SP)            ;;SAVE $TESTN FOR TYPEOUT
     011032  104403                                      TYPOS                           ;;GO TYPE--OCTAL ASCII
     011034     002                                      .BYTE   2                       ;;TYPE 2 DIGIT(S)
     011035     000                                      .BYTE   0                       ;;SUPPRESS LEADING ZEROS

     011036  012737  011044  001122                      MOV     #1$,$LPADR              ;SETUP LOOP ADDRESS
     011044  113737  002342  045506   1$:                MOVB    FS,DPB.B+10             ;FS
     011052  113737  002334  045507                      MOVB    FT,DPB.B+11             ;FT
```

```
    270 011060  012737  011220  001346          MOV     #EXIT6,BYPASS   ;GO TO EXIT6 ON ERROR
        011066                          TEST6:
    271 011066  013701  002326                  MOV     FC,R1           ;PICKUP "FC"
    272 011072  013702  002330                  MOV     LC,R2           ;FORM LAST CYLINDER THAT
    273 011076  162702  000005                  SUB     #5,R2           ;IS AVAILABLE FOR TESTING
    274 011102  012737  011102  001124          MOV     #.,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
        011110  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
    275 011114  020102                  1$:     CMP     R1,R2           ;LAST CYLINDER
    278 011116  003040                          BGT     EXIT6           ;YES--BRANCH
    279 011120  010137  045510                  MOV     R1,DPB.B+12     ;NC
    280 011124  004037  027264                  JSR     R0,CALL.B       ;GO EXECUTE THE COMMAND
    281 011130  062737  000004  045510          ADD     #4,DPB.B+12     ;NC+4
    282 011136  004037  027264                  JSR     R0,CALL.B       ;GO EXECUTE THE COMMAND
    283 011142  162737  000003  045510          SUB     #3,DPB.B+12     ;NC+1
    284 011150  004037  027264                  JSR     R0,CALL.B       ;GO EXECUTE THE COMMAND
    285 011154  062737  000002  045510          ADD     #2,DPB.B+12     ;NC+3
    286 011162  004037  027264                  JSR     R0,CALL.B       ;GO EXECUTE THE COMMAND
    287 011166  162737  000001  045510          SUB     #1,DPB.B+12     ;NC+2
    288 011174  004037  027264                  JSR     R0,CALL.B       ;GO EXECUTE THE COMMAND
    289 011200  062737  000003  045510          ADD     #3,DPB.B+12     ;NC+5
    290 011206  004037  027264                  JSR     R0,CALL.B       ;GO EXECUTE THE COMMAND
    291 011212  063701  002332                  ADD     IC,R1
    292 011216  000736                          BR      1$
    293 011220  000004                  EXIT6:  SCOPE                   ;CALL SCOPE ROUTINE
    294
    303
    304                                 ;;*****************************************************************
                                        ;*TEST 7        RANDOM SEEK TEST
                                        ;*THIS TEST PERFORMS RANDOM SEEK OPERATIONS BETWEEN CYLINDERS 'FC'
                                        ;*'LC'.  AFTER EACH SEEK, THE POSITION OF THE DRIVE IS VERIFIED BY
                                        ;*READING A SECTOR FROM THE CURRENTLY ADDRESSED CYLINDER AND TRACK.
                                        ;*THE TRACK ADDRESS IS INCREMENTED FOR EACH SEEK SO THAT VERIFICATION
                                        ;*OF POSITIONING OCCURS USING EACH HEAD.  TRACK ADDRESSES ARE INCREMENTED
                                        ;*BETWEEN PARAMTERS 'FT' AND 'LT'.
                                        ;;*****************************************************************
    011222                              TST7:
    011222  000240                              NOP
    011224  033737  001544  001330              BIT     BITS+<7*2>,TSTNMS       ;DO THIS TEST?
    011232  001002                              BNE     .+6             ;BR IF YES
    011234  000137  011620                      JMP     TST10           ;NO--JUMP TO TEST10

    011240  012737  000007  001116              MOV     #7,$TSTNM       ;SET TEST #7 AND CLEAR ($ERFLG)
    011246  004737  026672                      JSR     PC,LODPRM       ;LOAD THE PARMETERS FOR THE TEST
    011252  012737  011362  001124              MOV     #TEST7,$LPERR   ;SETUP THE LOOP ON ERROR ADDRESS
    011260  013737  002324  001220              MOV     RPT,$TIMES      ;GET THE ITERATION COUNT
    011266  112737  000031  001131              MOVB    #25.,$ERMAX     ;MAX ERRORS ALLOWED FOR TEST
    011274  012737  000007  001240              MOV     #7,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
305
    011302  032777  010000  167644              BIT     #SW12,@SWR      ;INHIBIT TYPING TEST NUMBER ?
    011310  001406                              BEQ     .+16            ;BR IF YES
    011312  104401  046253                      TYPE    ,MSGTST         ;TYPE 'TEST'
    011316  013746  001240                      MOV     $TESTN,-(SP)    ;;SAVE $TESTN FOR TYPEOUT
    011322  104403                              TYPOS                   ;;GO TYPE--OCTAL ASCII
    011324     002                              .BYTE   2               ;;TYPE 2 DIGIT(S)
    011325     000                              .BYTE   0               ;;SUPPRESS LEADING ZEROS

306 011326  113737  002334  045507              MOVB    FT,DPB.B+11     ;LOAD STARTING TRACK ADDRESS
```

```
                                                ;;*****************************************************************
                                                ;*TEST 10       SERVO SETTLE DOWN TEST
                                                ;*THIS TEST VERIFIES THAT THE SERVO HAS SETTLED DOWN AND THAT
                                                ;*THE DRIVE IS ON CYLINDER WHEN THE DRIVE INDICATES SEEK COMPLETE.
                                                ;*RANDOM SEEKS ARE ISSUED BETWEEN CYLINDERS 'NC1' AND 'NC1+IC'
```

```
 307 011334  112737  000105  045460          MOVB    #SEEK,DPB.A+2   ;SEEK=COMMAND
 308 011342  013704  037066                  MOV     RMADR,R4        ;UNIBUS ADDRESS OF THE RH/RM
 313 011346  012737  011616  001346          MOV     #EXIT7,BYPASS   ;ERROR TERMINATION ADDRESS
     011354  012737  011362  001122          MOV     #TEST7,$LPADR   ;SETUP THE LOOP ON TEST ADDRESS
     011362                           TEST7:
 314 011362  012706  001100                  MOV     #STACK,SP       ;SETUP THE STACK POINTER
 315 011366  013737  002326  045510          MOV     FC,DPB.B+12     ;INITIAL CYLINDER ADDRESS
 316 011374  023737  002326  002330          CMP     FC,LC           ;CYLINDER LIMITS THE SAME ?
 317 011402  001422                          BEQ     1$              ;BR IF THEY ARE
 318 011404  004737  025550                  JSR     PC,$RAND        ;CYCYLE THE RANDOM NUMBER GENERATOR
 319 011410  013746  025646                  MOV     $HINUM,-(SP)    ;USE THE HIGH RANDOM NUMBER
 320 011414  005046                          CLR     -(SP)           ;UPPER DIVIDEND
 321 011416  013746  002330                  MOV     LC,-(SP)        ;FORM THE DIVISOR
 322 011422  005216                          INC     (SP)            ;INCREMENT
 323 011424  163716  002326                  SUB     FC,(SP)         ;SUBTRACT THE LOWER LIMIT
 324 011430  004737  025652                  JSR     PC,$DIV         ;DIVIDE
 325 011434  062637  045510                  ADD     (SP)+,DPB.B+12  ;ADD THE REMAINDER TO THE INITIAL CYLINDER
 326 011440  005726                          TST     (SP)+           ;DISCARD THE QUOTENT
 327 011442  013737  045510  045470          MOV     DPB.B+12,DPB.A+12       ;COPY NEW CYLINDER ADDRESS
 328 011450                           1$:
     011450  012737  011450  001124          MOV     #.,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
     011456  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
 329 011462  004037  027132                  JSR     R0,CALL.A       ;GO EXECUTE THE COMMAND
 330 011466  012737  011466  001124          MOV     #.,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
     011474  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
 331 011500  113764  045456  000010          MOVB    DPB.A,RMCS2(R4) ;SELECT THE DRIVE
 332 011506  016446  000020                  MOV     RMLA(R4),-(SP)  ;GET THE LOOK AHEAD REGISTER
 333 011512  006316                          ASL     (SP)            ;ALIGN THE SECTOR ADDRESS
 334 011514  006316                          ASL     (SP)            ;ALIGN THE SECTOR ADDRESS
 335 011516  000316                          SWAB    (SP)            ;PUT ADDRESS IN LOWER BYTE
 336 011520  105766  000001                  TSTB    1(SP)           ;IN THE 1ST 20% OF SECTOR ?
 337 011524  001401                          BEQ     2$              ;BR IF YES
 338 011526  105216                          INCB    (SP)            ;INCREMENT THE SECTOR ADDRESS
 339 011530  105216                   2$:    INCB    (SP)            ;INCREMENT THE SECTOR ADDRESS
 340 011532  112637  045546                  MOVB    (SP)+,DTADPB+10 ;LOAD THE DPB
 341 011536  013746  002456                  MOV     PRMLMT+24,-(SP) ;PUT LAST SECTOR ADDRESS ON THE STACK
 342 011542  005216                          INC     (SP)            ;INCREMENT IT
 343 011544  122637  045546                  CMPB    (SP)+,DTADPB+10 ;NEW SECTOR ADDRESS TOO LARGE ?
 344 011550  103007                          BHIS    4$              ;BR IF NOT
 345 011552  103403                          BLO     3$              ;BR IF ADDRESS IS 2 GREATER
 346 011554  105037  045546                  CLRB    DTADPB+10       ;RESET TO SECTOR ADDRESS 0
 347 011560  000403                          BR      4$              ;CONTINUE
 348 011562  112737  000001  045546   3$:    MOVB    #1,DTADPB+10    ;RESET ADDRESS TO SECTOR 1
 349 011570                           4$:
     011570  004037  027264                  JSR     R0,CALL.B       ;GO EXECUTE THE COMMAND
 350 011574  105237  045507                  INCB    DPB.B+11        ;INCREMENT THE TRACK ADDRESS
 351 011600  123737  045507  002336          CMPB    DPB.B+11,LT     ;MAXIMUM ?
 354 011606  101403                          BLOS    EXIT7           ;BR IF NOT
 355 011610  113737  002334  045507          MOVB    FT,DPB.B+11     ;RELOAD STARTING TRACK ADDRESS
 356 011616  000004                   EXIT7: SCOPE                   ;CALL SCOPE ROUTINE
 357
 379
 380
```

```
                                ;*('NC1' STARTS AT VALUE 'FC').  AT THE COMPLETION OF 1000 (10) SEEKS,
                                ;*'NC1' IS INCREMENTED BY VALUE 'IC' AND THE SEQUENCE IS REPEATED.
                                ;*THE TEST IS COMPLETED WHEN 'NC1' HAS BEEN INCREMENTED BEYOND 'LC'.
                                ;*
                                ;*WHEN THE SEEK COMPLETES, THE PROGRAM READS THE DRIVE'S LOOK-AHEAD
                                ;*REGISTER (RMLA) TO DETERMINE THE ADDRESS OF THE SECTOR ROTATING  INTO
                                ;*POSITION.  THE PROGRAM THEN ISSUES A WRITE HEADER AND DATA COMMAND
                                ;*FOR THAT SECTOR.
                                ;*ERRORS IN THIS TEST INDICATE THAT THE SERVO SYSTEM MAY NOT BE ADJUSTED
                                ;*CORRECTLY, THAT THE DRIVE IS MALFUNCTIONING, OR THAT A PACK WITH
                                ;*MARGINAL SERVO TRACKS IS MOUNTED ON THE DRIVE.
                                ;*
                                ;*THIS TEST IS VALID ONLY IF THE OPERATION IS STARTED WITHIN A FEW
                                ;*HUNDRED MICRO-SECONDS AFTER SEEK DONE OCCURS.  THE NECESSARY
                                ;*TIME DEPENDENT PARAMETERS OCCUR FREQUENTLY ENOUGH WITHIN THE REQUIRED
                                ;*RANGE TO PERMIT THIS TEST TO BE EFFECTIVE.
                                ;;*****************************************************************
011620                          TST10:
011620  000240                          NOP
011622  033737  001546  001330         BIT     BITS+<10*2>,TSTNMS      ;DO THIS TEST?
011630  001002                          BNE     .+6                     ;BR IF YES
011632  000137  012750                 JMP     TST11                   ;NO--JUMP TO TEST11

011636  012737  000010  001116         MOV     #10,$TSTNM              ;SET TEST #10 AND CLEAR ($ERFLG)
011644  004737  026672                 JSR     PC,LODPRM               ;LOAD THE PARMETERS FOR THE TEST
011650  012737  012052  001124         MOV     #TEST10,$LPERR          ;SETUP THE LOOP ON ERROR ADDRESS
011656  013737  002324  001220         MOV     RPT,$TIMES              ;GET THE ITERATION COUNT
011664  112737  000031  001131         MOVB    #25.,$ERMAX             ;MAX ERRORS ALLOWED FOR TEST
011672  012737  000010  001240         MOV     #10,$TESTN              ;;SET TEST NUMBER IN APT MAIL BOX
381
011700  032777  010000  167246         BIT     #SW12,@SWR              ;INHIBIT TYPING TEST NUMBER ?
011706  001406                          BEQ     .+16                    ;BR IF YES
011710  104401  046253                 TYPE    ,MSGTST                 ;TYPE 'TEST'
011714  013746  001240                 MOV     $TESTN,-(SP)            ;;SAVE $TESTN FOR TYPEOUT
011720  104403                          TYPOS                           ;;GO TYPE--OCTAL ASCII
011722     003                          .BYTE   3                       ;;TYPE 3 DIGIT(S)
011723     000                          .BYTE   0                       ;;SUPPRESS LEADING ZEROS

382 011724  012737  011732  001122         MOV     #1$,$LPADR              ;SETUP THE LOOP ADDRESS
383 011732                      1$:
011732  112737  000105  045460         MOVB    #SEEK,DPB.A+2           ;SEEK=COMMAND
384 011740  112737  000161  045540         MOVB    #WRITE,DTADPB+2         ;COMMAND
385 011746  113737  002334  045547         MOVB    FT,DTADPB+11            ;TRACK ADDRESS FOR THE WRITE
386 011754  013737  002326  045470         MOV     FC,DPB.A+12             ;CYLINDER ADDRESS FOR THE SEEK
387 011762  013737  002326  045550         MOV     FC,DTADPB+12            ;CYLINDER ADDRESS FOR THE WRITE
388 011770  013737  002326  002356         MOV     FC,NC1                  ;STARTING CYLINDER
389 011776  013737  002332  001446         MOV     IC,DELTA                ;CYLINDER INCREMENT VALUE
390 012004  012737  176000  045542         MOV     #-<256.*4.>,DTADPB+4    ;WORD COUNT
391 012012  012737  052776  045544         MOV     #BUFFER,DTADPB+6        ;BUFFER ADDRESS
392 012020  005000                         CLR     R0                      ;PATTERN POINTER (WC PATTERN)
393 012022  004737  032732                 JSR     PC,SETBUF               ;LOAD THE WRITE BUFFER
394 012026  005001                         CLR     R1                      ;CLEAR REGISTER
395 012030  113701  045456                 MOVB    DPB.A,R1                ;LOAD DRIVE ADDRESS
396 012034  013704  037066                 MOV     RMADR,R4                ;UNIBUS ADDRESS OF THE RH/RM
397 012040  004737  044772                 JSR     PC,CLRQUE               ;CLEAR THE OPERATION QUEUES
438 012044  012737  012746  001346         MOV     #EXIT10,BYPASS          ;ERROR EXIT FROM TEST
012052                          TEST10:
```

```
012052  012737  012052  001124          MOV     #.,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
012060  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
012064  012737  000340  177776          MOV     #PR7,PS         ;SET PRIORITY TO 7
012072  005737  001340                  TST     CLKSTA          ;SEE WHICH CLOCK ON SYSTEM
012076  001415                          BEQ     3$              ;BR IF NO CLOCK
012100  100405                          BMI     1$              ;BR IF KW11-L CLOCK
012102  017746  167370                  MOV     @PKV,-(SP)      ;SAVE THE VECTOR
012106  013746  001476                  MOV     PKV,-(SP)       ;SAVE THE VECTOR ADDRESS
012112  000404                          BR      2$              ;CONTINUE
012114  017746  167370          1$:     MOV     @LKV,-(SP)      ;SAVE THE 'L' CLOCK VECTOR
012120  013746  001510                  MOV     LKV,-(SP)       ;SAVE THE VECTOR ADDRESS
012124  012776  012702  000000  2$:     MOV     #TST10B,@(SP)   ;CHANGE THE VECTOR
012132  012777  031272  024730  3$:     MOV     #DORTI,@RMVEC   ;CHANGE THE RM VECTOR
012140  012737  000010  001442          MOV     #8., SEKTMR     ;LOAD THE SEEK TIMER
012146  012764  000040  000010          MOV     #CLR,RMCS2(R4)  ;INIT THE MASSBUS
012154  110164  000010                  MOVB    R1,RMCS2(R4)    ;RESELECT THE DRIVE
012160  013764  045470  000034          MOV     DPB.A+12,RMDC(R4)       ;LOAD THE CYLINDER ADDRESS
012166  013737  045470  001364          MOV     DPB.A+12,CYL.DS ;CYLINDER ADDRESS FOR ERROR MESSAGE
012174  112764  000105  000000          MOVB    #SEEK,RMCS1(R4) ;START THE SEEK
012202  005037  177776                  CLR     PS              ;CLEAR THE PRIORITY
012206  105764  000012          4$:     TSTB    RMDS(R4)        ;HAS THE DRIVE FINISHED ?
012212  100402                          BMI     5$              ;BR IF IT HAS
012214  000001                          WAIT                    ;WAIT FOR THE OPERATION TO COMPLETE
012216  000773                          BR      4$              ;CONTINUE
012220  012737  000340  177776  5$:     MOV     #PR7,PS         ;CHANGE PRIORITY TO MAX
012226  032764  040000  000012          BIT     #ERR,RMDS(R4)   ;ERROR ?
012234  001412                          BEQ     6$              ;BR IF NOT
012236  012702  045456                  MOV     #DPB.A,R2       ;DPB POINTER
012242  004737  044310                  JSR     PC,SVRH70       ;SAVE THE REGISTERS
012246  104023                          EMT     23
012250  012764  000040  000010          MOV     #CLR,RMCS2(R4)  ;INIT THE MASSBUS
012256  110164  000010                  MOVB    R1,RMCS2(R4)    ;RESELECT THE DRIVE
012262  012777  041740  024600  6$:     MOV     #ISR,@RMVEC     ;SETUP THE RM VECTOR
012270  005737  001340                  TST     CLKSTA          ;WHICH CLOCK
012274  001405                          BEQ     TST10A          ;BR IF NONE
012276  016676  000002  000000          MOV     2(SP),@(SP)     ;RELOAD THE CLOCK VECTOR
012304  062706  000004                  ADD     #4,SP           ;CORRECT THE STACK POINTER
012310                          TST10A:
439 012310  012737  012310  001124      MOV     #.,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
    012316  012706  001100              MOV     #STACK,SP       ;LOAD THE STACK POINTER
440 012322  110164  000010              MOVB    R1,RMCS2(R4)    ;SELECT THE DRIVE
441 012326  016446  000020              MOV     RMLA(R4),-(SP)  ;GET THE LOOK AHEAD REGISTER
442 012332  006316                      ASL     (SP)            ;ALIGN THE SECTOR ADDRESS
443 012334  006316                      ASL     (SP)            ;ALIGN THE SECTOR ADDRESS
444 012336  000316                      SWAB    (SP)            ;PUT ADDRESS IN LOWER BYTE
445 012340  122766  000300  000001      CMPB    #300,1(SP)      ;IN THE LAST 20% OR SECTOR ?
446 012346  001001                      BNE     2$              ;BR IF NOT
447 012350  105216                      INCB    (SP)            ;INCREMENT THE SECTOR ADDRESS
448 012352  105216                  2$: INCB    (SP)            ;INCREMENT THE SECTOR ADDRESS
449 012354  112637  045546              MOVB    (SP)+,DTADPB+10 ;LOAD THE DPB
450 012360  013746  002456              MOV     PRMLMT+24,-(SP) ;PUT MAXIMUM SECTOR ADDRESS ON THE STACK
451 012364  005216                      INC     (SP)            ;INCREMENT PAST THE MAXIMUM ADDRESS
452 012366  122637  045546              CMPB    (SP)+,DTADPB+10 ;NEW SECTOR ADDRESS TOO LARGE ?
453 012372  101007                      BHI     4$              ;BR IF NOT
454 012374  103403                      BLO     3$              ;BR IF ADDRESS IS 2 GREATER THAN MAXIMUM
455 012376  105037  045546              CLRB    DTADPB+10       ;RESET TO SECTOR ADDRESS 0
456 012402  000403                      BR      4$              ;CONTINUE
```

```
    457 012404  112737  000001  045546  3$:     MOVB    #1,DTADPB+10    ;RESET ADDRESS TO SECTOR 1
    458 012412  012703  045542          4$:     MOV     #DTADPB+4,R3    ;POINTER
    459 012416  012764  000111  000000          MOV     #DRVCLR,RMCS1(R4)       ;CLEAR THE DRIVE
    460 012424  012364  000002                  MOV     (R3)+,RMWC(R4)  ;LOAD THE WORD COUNT
    461 012430  012364  000004                  MOV     (R3)+,RMBA(R4)  ;LOAD THE BUFFER ADDRESS
    462 012434  012364  000006                  MOV     (R3)+,RMDA(R4)  ;LOAD THE TRACK/SECTOR ADDR
    463 012440  005037  045554                  CLR     DTADPB+16       ;RESET 'DONE' INDICATOR
    464 012444  012737  045536  037000          MOV     #DTADPB,TRNSWT  ;LOAD 'TRANSFER' DPB ADDRESS
    465 012452  010137  037052                  MOV     R1,DTUW         ;ADDRESS OF DRIVE TRANSFERING
    466 012456  112761  000001  036730          MOVB    #1,DRVACT(R1)   ;SET DRIVE ACTIVE INDICATOR
    467 012464  006301                          ASL     R1              ;SHIFT DRIVE ADDRESS
    468 012466  012761  001750  037032          MOV     #1000.,TIMER(R1)        ;SETUP THE OPERATION TIMER
    469 012474  006201                          ASR     R1              ;RESTORE R1
    470 012476  013764  045540  000000          MOV     DTADPB+2,RMCS1(R4)      ;START THE OPERATION
    471 012504  005037  177776                  CLR     PS              ;CLEAR THE PRIORITY
    472 012510  004037  027710                  JSR     R0,DRVCL1       ;WAIT FOR OPERATION TO COMPLETE
    473 012514  023727  001444  001750  5$:     CMP     SEKCNT,#1000.   ;FINISHED SEEKS ?
    474 012522  001026                          BNE     6$              ;BR IF NOT
    475 012524  005037  001444                  CLR     SEKCNT          ;CLEAR THE SEEK COUNT
    476 012530  063737  002332  002356          ADD     IC,NC1          ;ADD THE INCREMENT
    477 012536  023737  002356  002330          CMP     NC1,LC          ;EXCEEDED THE CYLINDER LIMIT ?
    512 012544  103100                          BHIS    EXIT10          ;BR IF IT HAS
        012546  013737  002330  001446          MOV     LC,DELTA        ;GET THE NEXT 'ZONE' ADDRESS
        012554  163737  002356  001446          SUB     NC1,DELTA       ;CHECK THE DIFFERENCE
        012562  023737  002332  001446          CMP     IC,DELTA        ;DIFFERENCE GREATER THAN THE INCREMENT ?
        012570  101003                          BHI     6$              ;BR IF IT IS
        012572  013737  002332  001446          MOV     IC,DELTA        ;USE THE ICREMENT PARAMETER
        012600  005237  001444          6$:     INC     SEKCNT          ;COUNT THE NEXT SEEK
        012604  023737  002326  002330          CMP     FC,LC           ;BEGINNING AND ENDING CYLINDERS THE SAME ?
        012612  001002                          BNE     7$              ;BR IF NOT
        012614  000137  012052                  JMP     TEST10          ;BR IF THEY ARE
        012620  013737  002356  045470  7$:     MOV     NC1,DPB.A+12    ;RESET THE CYLINDER ADDRESS
        012626  004737  025550                  JSR     PC,$RAND        ;CYCLE THE RANDOM NUMBER GENERATOR
        012632  013746  025646                  MOV     $HINUM,-(SP)    ;USE THE HIGH RANDOM NUMBER
        012636  005046                          CLR     -(SP)           ;CLEAR THE UPPER DIVIDEND
        012640  013746  001446                  MOV     DELTA,-(SP)     ;FORM THE DIVISOR
        012644  005216                          INC     (SP)            ;INCREMENT
        012646  004737  025652                  JSR     PC,$DIV         ;DIVIDE
        012652  062637  045470                  ADD     (SP)+,DPB.A+12  ;ADD THE REMAINDER TO THE INITIAL CYLINDER
        012656  005726                          TST     (SP)+           ;DISCARD THE QUOTENT
        012660  023737  045470  045550          CMP     DPB.A+12,DTADPB+12      ;SAME CYLINDER SELECTED AS LAST TIME ?
        012666  001754                          BEQ     7$              ;BR IF IT WAS
        012670  013737  045470  045550          MOV     DPB.A+12,DTADPB+12      ;COPY NEW CYLINDER ADDRESS
        012676  000137  012052                  JMP     TEST10          ;CONTINUE
        012702                          TST10B:
        012702  005337  001442                  DEC     SEKTMR          ;DECREMENT THE SEEK TIMER
        012706  001016                          BNE     1$              ;CONTINUE IF NOT DONE
        012710  012702  045456                  MOV     #DPB.A,R2       ;DPB ADDRESS
        012714  004737  044310                  JSR     PC,SVRH70       ;SAVE THE REGISTERS
        012720  104024                          EMT     24
        012722  012764  000040  000010          MOV     #CLR,RMCS2(R4)  ;INIT THE MASSBUS
        012730  110164  000010                  MOVB    R1,RMCS2(R4)    ;RESELECT THE DRIVE
        012734  016676  000002  000000          MOV     2(SP),@(SP)     ;RESTORE THE CLOCK VECTOR ADDRESS
        012742  000401                          BR      EXIT10          ;ABORT THE TEST
    513 012744  000002                  1$:     RTI                     ;RETURN
    514 012746  000004                  EXIT10: SCOPE                   ;CALL SCOPE ROUTINE
    515
```

```
526
527                                     ;;*****************************************************************
                                        ;*TEST 11        ALL SEEKS TEST
                                        ;*THIS TEST VERIFIES THAT THE DISK DRIVE CAN SEEK FROM EACH CYLINDER
                                        ;*TO ALL OTHER CYLINDERS.
                                        ;*
                                        ;*BEGINNING WITH CYLINDER 'FC', THE TEST SEEKS TO EACH CYLINDER
                                        ;*BETWEEN 'FC' AND 'LC' FROM CYINDER 'FC'.  THE BEGINNING CYLINDER
                                        ;*ADDRESS IS INCREMENTED AND THE TEST SEEKS BETWEEN THE NEW CYLINDER
                                        ;*ADDRESS AND ALL CYLINDERS BETWEEN 'FC' AND 'LC'.  THE SEQUENCE
                                        ;*CONTINUES UNTIL ALL CYLINDERS HAVE BEEN CHECKED.
                                        ;;*****************************************************************
012750                                  TST11:
012750  000240                                  NOP
012752  033737  001550  001330                 BIT     BITS+<11*2>,TSTNMS      ;DO THIS TEST?
012760  001002                                  BNE     .+6                     ;BR IF YES
012762  000137  013214                          JMP     TST12                   ;NO--JUMP TO TEST12

012766  012737  000011  001116                 MOV     #11,$TSTNM              ;SET TEST #11 AND CLEAR ($ERFLG)
012774  004737  026672                          JSR     PC,LODPRM               ;LOAD THE PARMETERS FOR THE TEST
013000  012737  013134  001124                 MOV     #TEST11,$LPERR          ;SETUP THE LOOP ON ERROR ADDRESS
013006  013737  002324  001220                 MOV     RPT,$TIMES              ;GET THE ITERATION COUNT
013014  112737  000031  001131                 MOVB    #25.,$ERMAX             ;MAX ERRORS ALLOWED FOR TEST
013022  012737  000011  001240                 MOV     #11,$TESTN              ;;SET TEST NUMBER IN APT MAIL BOX
528
013030  032777  010000  166116                 BIT     #SW12,@SWR              ;INHIBIT TYPING TEST NUMBER ?
013036  001406                                  BEQ     .+16                    ;BR IF YES
013040  104401  046253                          TYPE    ,MSGTST                 ;TYPE 'TEST'
013044  013746  001240                          MOV     $TESTN,-(SP)            ;;SAVE $TESTN FOR TYPEOUT
013050  104403                                  TYPOS                           ;;GO TYPE--OCTAL ASCII
013052     003                                  .BYTE   3                       ;;TYPE 3 DIGIT(S)
013053     000                                  .BYTE   0                       ;;SUPPRESS LEADING ZEROS

529 013054  012737  013062  001122          1$:     MOV     #1$,$LPADR              ;SETUP THE LOOP ADDRESS
530 013062  113737  002342  045506                  MOVB    FS,DPB.B+10             ;SECTOR ADDRESS
531 013070  113737  002342  045526                  MOVB    FS,DPB.C+10             ;SECTOR ADDRESS
532 013076  113737  002334  045507                  MOVB    FT,DPB.B+11             ;TRACK ADDRESS
533 013104  113737  002334  045527                  MOVB    FT,DPB.C+11             ;TRACK ADDRESS
534 013112  013737  002326  045510                  MOV     FC,DPB.B+12             ;STARTING CYLINDER ADDRESS
535 013120  013737  002326  045530                  MOV     FC,DPB.C+12             ;STARTING CYLINDER ADDRESS
539 013126  012737  013212  001346                  MOV     #EXIT11,BYPASS          ;TEST ABORT EXIT
    013134                                  TEST11:
540 013134  012706  001100                          MOV     #STACK,SP               ;SETUP THE STACK POINTER
541 013140                                  1$:
    013140  004037  027466                          JSR     R0,CALL.C               ;GO EXECUTE THE COMMAND
542 013144  004037  027264                          JSR     R0,CALL.B               ;GO EXECUTE THE COMMAND
543 013150  063737  002332  045530                  ADD     IC,DPB.C+12             ;INCREMENT THE ENDING CYLINDER ADDRESS
544 013156  023737  002330  045530                  CMP     LC,DPB.C+12             ;CHECK IF EXCEEDING MAXIMUM
545 013164  002365                                  BGE     1$                      ;BR IF NOT
546 013166  013737  002326  045530                  MOV     FC,DPB.C+12             ;RESET ENDING CYLINDER ADDRESS
547 013174  063737  002332  045510                  ADD     IC,DPB.B+12             ;INCREMENT THE STARTING ADDRESS
548 013202  023737  002330  045510                  CMP     LC,DPB.B+12             ;EXCEEDING MAXIMUM ?
549 013210  002353                                  BGE     1$                      ;BR IF NOT
550 013212  000004                          EXIT11: SCOPE                           ;CALL SCOPE ROUTINE
551
```

```
                        .SBTTL  TIMING TESTS

                        ;;////////////////////////////////////////////////////////////////
                        ;*THE TIMING TESTS WILL ENSURE THAT THOSE FUNCTIONS BEING
                        ;*TIMED ARE WITHIN THE TOLERANCES SPECIFIED IN THE "RM05/3/2
                        ;*ENGINEERING SPECIFICATIONS".
                        ;*THE SEEK TIMING WILL BE PERFORMED USING EXPLICIT SEEK
                        ;*OPERATIONS. AT THE COMPLETION OF EACH OF THE TIMING
                        ;*TESTS THE MINIMUM, MAXIMUM AND AVERAGE TIMES WILL BE
                        ;*TYPED.
                        ;;////////////////////////////////////////////////////////////////


                        ;;****************************************************************
                        ;*TEST 12       ROTATIONAL SPEED TIMING TEST
                        ;*THIS TEST WILL START A SEARCH TO CYLINDER 'FC', TRACK 'FT',
                        ;*SECTOR 'FS'.  AS SOON AS THE INTERRUPT OCCURS, THE GO BIT
                        ;*IS SET AGAIN AND THE OPERATION IS TIMED.  THIS PROCEDURE
                        ;*IS REPEATED 10 TIMES THEN THE AVERAGE TIME IS CALCULATED
                        ;*AND CHECKED TO ENSURE IT IS WITHIN TOLERANCE:
                        ;*
                        ;*      RM05/3:
                        ;*              16.67 MS/REV + OR - 2% IF 60HZ
                        ;*              16.67 MS/REV + OR - 2.5% IF 50HZ.
                        ;*
                        ;*      RM02:
                        ;*              25.00 MS/REV + OR - 2% IF 60HZ
                        ;*              25.00 MS/REV + OR - 2.5% IF 50HZ.
                        ;;****************************************************************
013214                  TST12:
013214  000240                  NOP
013216  033737  001552  001330  BIT     BITS+<12*2>,TSTNMS      ;DO THIS TEST?
013224  001002                  BNE     .+6                     ;BR IF YES
013226  000137  014136          JMP     TST13                   ;NO--JUMP TO TEST13

013232  012737  000012  001116  MOV     #12,$TSTNM              ;SET TEST #12 AND CLEAR ($ERFLG)
013240  004737  026672          JSR     PC,LODPRM               ;LOAD THE PARMETERS FOR THE TEST
013244  012737  014136  001124  MOV     #TST13,$LPERR           ;SETUP THE ERROR LOOP ADDRESS
013252  013737  002324  001220  MOV     RPT,$TIMES              ;GET THE ITERATION COUNT
013260  112737  000031  001131  MOVB    #25.,$ERMAX             ;MAX ERRORS ALLOWED FOR TEST
013266  012737  000012  001240  MOV     #12,$TESTN              ;;SET TEST NUMBER IN APT MAIL BOX
30
013274  032777  010000  165652  BIT     #SW12,@SWR              ;INHIBIT TYPING TEST NUMBER ?
013302  001406                  BEQ     .+16                    ;BR IF YES
013304  104401  046253          TYPE    ,MSGTST                 ;TYPE 'TEST'
013310  013746  001240          MOV     $TESTN,-(SP)            ;;SAVE $TESTN FOR TYPEOUT
013314  104403                  TYPOS                           ;;GO TYPE--OCTAL ASCII
013316     003                  .BYTE   3                       ;;TYPE 3 DIGIT(S)
013317     000                  .BYTE   0                       ;;SUPPRESS LEADING ZEROS

31 013320  005737  001340          TST     CLKSTA              ;KW11-P CLOCK?
32 013324  003002                  BGT     1$                  ;YES--START TEST
35 013326  000137  014134          JMP     EXIT12              :NO--JUMP TO EXIT12
36 013332  012737  013332  001122  1$:     MOV     #.,$LPADR   ;SETUP LOOP ADDRESS
37 013340  004037  031110          JSR     R0,SRCH00           ;DO A MASSBUS INIT & RECAL
38 013344  000402                  BR      2$                  ;RETURN HERE IF NO ERROR
60 013346  000137  014134          JMP     EXIT12              ;RETURN HERE IF ERROR
```

```
013352  012737  013352  001122  2$:     MOV     #.,$LPADR       ;ERROR LOOP ADDRESS
013360  112737  000105  045460          MOVB    #SEEK,DPB.A+2   ;SEEK=COMMAND
013366  005037  045466                  CLR     DPB.A+10        ;USE TRACK 0 & SECTOR 0
013372  013737  002326  045470          MOV     FC,DPB.A+12     ;STARTING CYLINDER
013400  012737  014134  001346          MOV     #EXIT12,BYPASS  ;GO TO EXIT12 IF ERROR
013406  004037  027132                  JSR     R0,CALL.A       ;GO EXECUTE THE COMMAND
013412  013764  002326  000034          MOV     FC,RMDC(R4)     ;FC
013420  013746  002342                  MOV     FS,-(SP)        ;FS
013424  113766  002334  000001          MOVB    FT,1(SP)        ;FT
013432  012664  000006                  MOV      SP)+,RMDA(R4)  ;LOAD FT/FS
013436  012737  014134  001222          MOV     #EXIT12,$ESCAPE ;;ESCAPE TO EXIT12 ON ERROR
013444  005005                          CLR     R5              ;COUNT UP

                        ;SETUP PARAMETER TABLE FOR RM05/RM03/RM02
013446  010046                          MOV     R0,-(SP)        ;SAVE R0
013450  113700  045536                  MOVB    DTADPB,R0       ;DRIVE ADDRESS
013454  032737  000100  001314          BIT     #SW06,C.SWR     ;CHECK CONTROL SWR FOR 60 HZ.
013462  001425                          BEQ     3$              ;BR IF YES
013464  012703  001636                  MOV     #T7B1,R3        ;LOAD 50 HZ. TABLE FOR RM02
013470  012737  001636  014124          MOV     #T7B1,TP50      ;
013476  012737  001720  014132          MOV     #SP7B1,TPS50    ;
013504  122760  000005  036750          CMPB    #5,DRVTYP(R0)   ;AN RM02 DRIVE ?
013512  001435                          BEQ     4$              ;BRANCH IF SO
013514  012703  001616                  MOV     #T7B,R3         ;LOAD 50 HZ. TABLE FOR RM05/3
013520  012737  001616  014124          MOV     #T7B,TP50       ;
013526  012737  001704  014132          MOV     #SP7B,TPS50     ;
013534  000424                          BR      4$              ;EXIT
013536  012737  001626  014106  3$:     MOV     #T7A1,TP60      ;LOAD 60 HZ. TABLE FOR RM02
013544  012737  001712  014114          MOV     #SP7A1,TPS60    ;
013552  012703  001626                  MOV     #T7A1,R3        ;
013556  122760  000005  036750          CMPB    #5,DRVTYP(R0)   ;AN RM02 DRIVE ?
013564  001410                          BEQ     4$              ;BRANCH IF SO
013566  012737  001606  014106          MOV     #T7A,TP60       ;LOAD 60 HZ. TABLE FOR RM05/3
013574  012703  001606                  MOV     #T7A,R3         ;
013600  012737  001676  014114          MOV     #SP7A,TPS60     ;
013606  012600                  4$:     MOV     (SP)+,R0        ;RESTORE R0
013610  000240                          NOP                     ;EXIT
013612                          TEST12:
013612  012706  001100                  MOV     #STACK,SP       ;SETUP STACK
013616  012701  000012                  MOV     #10.,R1         ;TIME 10 SEARCHES
013622  004737  031274                  JSR     PC,STRTMR       ;INITIALIZE THE TIMERS
013626  012777  014014  165642          MOV     #7$,@PKV        ;SETUP VECTOR IN CASE OF OVERFLOW
013634  012777  031272  023226          MOV     #DORT1,@RMVEC   ;SETUP RM VECTOR
013642  005077  165636          1$:     CLR     @PKB            ;START COUNTING AT ZERO
013646  012777  000131  165626          MOV     #131,@PKCS      ;INT.EN., COUNT UP AT 100KHZ
013654  012714  000131                  MOV     #SEARCH,(R4)    ;START A SEARCH
013660  000001                          WAIT                    ;WAIT ON INTERRUPT
013662  042777  000101  165612          BIC     #101,@PKCS      ;STOP THE CLOCK
013670  032764  040000  000012          BIT     #ERR,RMDS(R4)   ;ERROR?
013676  001411                          BEQ     2$              ;NO--BRANCH
013700  104412                          SAVREG                  ;SAVE R0-R5
013702  012702  045536                  MOV     #DTADPB,R2      ;DPB POINTER
013706  004737  044310                  JSR     PC,SVRH70       ;SAVE ALL THE RH/RM REGISTERS
013712  004737  026072                  JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
013716  104413                          RESREG                  ;RESTORE R0-R5
013720  104017                          EMT     17
013722  005077  165556          2$:     CLR     @PKB            ;START THE COUNT AT ZERO
```

```
                                        MOV     #SEARCH,(R4)    ;START A SEARCH
                                        MOV     #131,@PKCS      ;START THE CLOCK
                                        WAIT                    ;WAIT ON INTERRUPT
                                        BIC     #101,@PKCS      ;STOP THE CLOCK
                                        BIT     #ERR,RMDS(R4)   ;IS "ERR=1"?
                                        BEQ     3$              ;NO--BRANCH
                                        SAVREG                  ;SAVE R0-R5
                                        MOV     #DTADPB,R2      ;DPB POINTER
                                        JSR     PC,SVRH70       ;SAVE ALL THE RH/RM REGISTERS
                                        JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
                                        RESREG                  ;RESTORE R0-R5
                                        EMT     17
                                3$:     JSR     PC,COUNT        ;UPDATE THE COUNT
                                        DEC     R1              ;DONE?
                                        BGT     1$              ;NO--BRANCH
                                        BR      8$              ;YES--GO TO THE EXIT
                                7$:     BIC     #101,@PKCS      ;STOP THE CLOCK
                                        CLR     PS              ;DROP THE PRIORITY
                                        MOV     (SP)+,R0        ;PC OF WAIT+2
                                        TST     (SP)+           ;POP THE PS FROM THE STACK
                                        SAVREG                  ;SAVE R0-R5
                                        MOV     #DTADPB,R2      ;DPB POINTER
                                        JSR     PC,SVRH70       ;SAVE ALL THE RH/RM REGISTERS
                                        JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
                                        RESREG                  ;RESTORE R0-R5
                                        EMT     20
                                8$:     JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
                                        JSR     PC,ST.CLK       ;INITIALIZE THE CLOCK
                                        MOV     #ISR,@RMVEC     ;RESTORE RH/RM INT. VECTOR
                                        BIT     #SW06,C.SWR     ;60 HZ?
                                        BNE     EXIT.A          ;NO-BRANCH
                                        JSR     R0,TYPTIM       ;TYPE THE TIMING
                                TP60:   T7A                     ;TABLE ADDRESS
                                        JSR     R0,SPTYP        ;TYPE THE SPEC
                                TPS60:  SP7A
                                        BR      EXIT12          ;EXIT
                                EXIT.A: JSR     R0,TYPTIM       ;TYPE THE TIMING
                                TP50:   T7B                     ;TABLE ADDRESS
                                        JSR     R0,SPTYP        ;
                                TPS50:  SP7B                    ;
                                EXIT12: SCOPE                   ;CALL SCOPE ROUTINE


                                ;;*****************************************************************
                                ;*TEST 13       ONE CYLINDER SEEK TIMING TEST
                                ;*THIS TEST WILL COMMAND FORWARD SEEK CYCLES TO ADVANCE THE
                                ;*CYLINDER BY ONE UNTIL THE INCREMENT IS GREATER THAN THE
                                ;*CYLINDER 'LC', THEN REVERSE SEEK TO CYLINDER 'FC'.  THE
                                ;*TIME TO PERFORM EACH SEEK IS CHECKED TO ENSURE IT DOES NOT
                                ;*EXCEED THE MAXIMUM TIME PERMITTED FOR A ONE CYLINDER SEEK.
                                ;*THE TIME MUST BE LESS THAN 10MS.
                                ;;*****************************************************************
014136                          TST13:
014136  000240                          NOP
014140  033737  001554  001330          BIT     BITS+<13*2>,TSTNMS      ;DO THIS TEST?
014146  001002                          BNE     .+6                     ;BR IF YES
014150  000137  014640                  JMP     TST14                   ;NO--JUMP TO TEST14
```

```
                014154  012737  000013  001116          MOV     #13,$TSTNM      ;SET TEST #13 AND CLEAR ($ERFLG)
                014162  004737  026672                  JSR     PC,LODPRM       ;LOAD THE PARMETERS FOR THE TEST
                014166  012737  014640  001124          MOV     #TST14,$LPERR   ;SETUP THE ERROR LOOP ADDRESS
                014174  013737  002324  001220          MOV     RPT,$TIMES      ;GET THE ITERATION COUNT
                014202  112737  000031  001131          MOVB    #25.,$ERMAX     ;MAX ERRORS ALLOWED FOR TEST
                014210  012737  000013  001240          MOV     #13,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
            155
                014216  032777  010000  164730          BIT     #SW12,@SWR      ;INHIBIT TYPING TEST NUMBER ?
                014224  001406                          BEQ     .+16            ;BR IF YES
                014226  104401  046253                  TYPE    ,MSGTST         ;TYPE 'TEST'
                014232  013746  001240                  MOV     $TESTN,-(SP)    ;;SAVE $TESTN FOR TYPEOUT
                014236  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
                014240     003                          .BYTE   3               ;;TYPE 3 DIGIT(S)
                014241     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS

            156 014242  005737  001340                  TST     CLKSTA          ;KW11-P CLOCK?
            157 014246  003002                          BGT     1$              ;YES--START TEST
            160 014250  000137  014636                  JMP     EXIT13          ;NO--JUMP TO EXIT13
            161 014254  012737  014254  001122  1$:     MOV     #.,$LPADR       ;SETUP THE LOOP ADDRESS
            162 014262  004037  031110                  JSR     R0,SRCH00       ;DO A MASSBUS INIT. AND RECAL
            163 014266  000402                          BR      2$              ;NO ERROR RETURN
            175 014270  000137  014636                  JMP     EXIT13          ;ERROR RETURN--SCOPE LOOP CALL
                014274  012737  014274  001122  2$:     MOV     #.,$LPADR       ;ERROR LOOP ADDRESS
                014302  112737  000105  045460          MOVB    #SEEK,DPB.A+2   ;SEEK=COMMAND
                014310  005037  045466                  CLR     DPB.A+10        ;USE TRACK 0 & SECTOR 0
                014314  013737  002326  045470          MOV     FC,DPB.A+12     ;STARTING CYLINDER
                014322  012737  014636  001346          MOV     #EXIT13,BYPASS  ;GO TO EXIT13 IF ERROR
                014330  004037  027132                  JSR     R0,CALL.A       ;GO EXECUTE THE COMMAND
                014334  012703  001646                  MOV     #T10,R3         ;PARAMETER POINTER
                014340  012737  014636  001222          MOV     #EXIT13,$ESCAPE ;;ESCAPE TO EXIT13 ON ERROR
                014346                          TEST13:
            176 014346  012706  001100                  MOV     #STACK,SP       ;SETUP STACK
            177 014352  013737  002326  045550          MOV     FC,DTADPB+12    ;START WITH BEGINNING CYLINDER
            178 014360  005237  045550                  INC     DTADPB+12       ;INCREMENT THE BEGINNING CYLINDER
            179 014364  005005                          CLR     R5              ;SET THE UP/DOWN SWITCH TO UP
            180 014366  004737  031274                  JSR     PC,STRTMR       ;INITIALIZE THE TIMERS
            181 014372  012777  014544  165076          MOV     #7$,@PKV        ;SETUP INCASE OF OVERFLOW
            182 014400  012777  031272  022462          MOV     #DORTI,@RMVEC   ;SET RM VECTOR
            183 014406  005077  165072          1$:     CLR     @PKB            ;START THE COUNTER AT ZERO
            184 014412  013764  045550  000034          MOV     DTADPB+12,RMDC(R4)      ;LOAD DESIRED CYLINDER
            185 014420  012714  000105                  MOV     #SEEK,(R4)      ;START A SEEK
            186 014424  012777  000131  165050          MOV     #131,@PKCS      ;START THE CLOCK
            187 014432  000001                          WAIT                    ;WAIT ON INTERRUPT
            188 014434  042777  000101  165040          BIC     #101,@PKCS      ;STOP THE CLOCK
            189 014442  032764  040000  000012          BIT     #ERR,RMDS(R4)   ;ANY DISK ERRORS?
            190 014450  001411                          BEQ     2$              ;NO--BRANCH
            191 014452  104412                          SAVREG                  ;SAVE R0-R5
                014454  012702  045536                  MOV     #DTADPB,R2      ;DPB POINTER
                014460  004737  044310                  JSR     PC,SVRH70       ;SAVE ALL THE RH/RM REGISTERS
                014464  004737  026072                  JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
                014470  104413                          RESREG                  ;RESTORE R0-R5
            192 014472  104017                          EMT     17
            193 014474  004737  031336          2$:     JSR     PC,COUNT        ;COUNT THIS SEEKS TIME
            194 014500  005705                          TST     R5              ;UP OR DOWN?
            195 014502  001011                          BNE     4$              ;DOWN--BRANCH
            196 014504  005237  045550          3$:     INC     DTADPB+12       ;MOVE TO NEXT CYLINDER
```

```
014510  023737  045550  002330          CMP     DTADPB+12,LC    ;OUT OF CYLINDERS?
014516  002733                          BLT     1$              ;NO--GO DO THE NEXT SEEK
014520  012705  177777                  MOV     #-1,R5          ;SET UP/DOWN SWITCH TO DOWN
014524  000730                          BR      1$              ;GO DO THE NEXT SEEK
014526  005337  045550          4$:     DEC     DTADPB+12       ;MOVE TO NEXT CYLINDER
014532  023737  045550  002326          CMP     DTADPB+12,FC    ;OUT OF CYLINDERS?
014540  003322                          BGT     1$              ;NO--GO DO THE NEXT SEEK
014542  000420                          BR      8$              ;GO TO THE EXIT
014544  042777  000101  164730  7$:     BIC     #101,@PKCS      ;STOP THE CLOCK
014552  005037  177776                  CLR     PS              ;DROP THE PRIORITY
014556  012600                          MOV     (SP)+,R0        ;PC OF WAIT+2
014560  005726                          TST     (SP)+           ;POP THE PS FROM THE STACK
014562  104412                          SAVREG                  ;SAVE R0-R5
014564  012702  045536                  MOV     #DTADPB,R2      ;DPB POINTER
014570  004737  044310                  JSR     PC,SVRH70       ;SAVE ALL THE RH/RM REGISTERS
014574  004737  026072                  JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
014600  104413                          RESREG                  ;RESTORE R0-R5
014602  104020                          EMT     20
014604  004737  026072          8$:     JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
014610  004737  026156                  JSR     PC,ST.CLK       ;INITIALIZE THE CLOCK
014614  012777  041740  022246          MOV     #ISR,@RMVEC     ;RESTORE RH/RM INT. VECTOR
014622  004037  031576                  JSR     R0,TYPTIM       ;GO TYPE THE TIMES
014626  001646                          T10                     ;POINTER
014630  004037  031470                  JSR     R0,SPTYP
014634  001726                          SP10
014636  000004                  EXIT13: SCOPE                   ;CALL SCOPE ROUTINE


                        ;;*****************************************************************
                        ;*TEST 14       ACCESS TIME MEASUREMENT TEST
                        ;*THIS TEST WILL COMMAND A FORWARD SEEK FROM CYLINDER 'FC' TO
                        ;*CYLINDER 'LC', THEN A REVERSE  SEEK FROM CYLINDER 'LC' TO
                        ;*CYLINDER 'FC'.  BOTH SEEKS ARE TIMED AND CHECKED TO ENSURE THEY
                        ;*ARE WITHIN THE TOLERANCE ALLOWED FOR THE ACCESS TIME MEASUREMENT.
                        ;*THIS SEQUENCE IS REPEATED 128 TIMES (FOR A TOTAL OF 256 SEEKS).
                        ;*THE AVERAGE ACCESS TIME MUST BE LESS THAN 30 MS.
                        ;;*****************************************************************
014640                  TST14:
014640  000240                  NOP
014642  033737  001556  001330          BIT     BITS+<14*2>,TSTNMS      ;DO THIS TEST?
014650  001002                          BNE     .+6             ;BR IF YES
014652  000137  015400                  JMP     TST15           ;NO--JUMP TO TEST15

014656  012737  000014  001116          MOV     #14,$TSTNM      ;SET TEST #14 AND CLEAR ($ERFLG)
014664  004737  026672                  JSR     PC,LODPRM       ;LOAD THE PARMETERS FOR THE TEST
014670  012737  015400  001124          MOV     #TST15,$LPERR   ;SETUP THE ERROR LOOP ADDRESS
014676  013737  002324  001220          MOV     RPT,$TIMES      ;GET THE ITERATION COUNT
014704  112737  000031  001131          MOVB    #25.,$ERMAX     ;MAX ERRORS ALLOWED FOR TEST
014712  012737  000014  001240          MOV     #14,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX

014720  032777  010000  164226          BIT     #SW12,@SWR      ;INHIBIT TYPING TEST NUMBER ?
014726  001406                          BEQ     .+16            ;BR IF YES
014730  104401  046253                  TYPE    ,MSGTST         ;TYPE 'TEST'
014734  013746  001240                  MOV     $TESTN,-(SP)    ;;SAVE $TESTN FOR TYPEOUT
014740  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
014742     003                          .BYTE   3               ;;TYPE 3 DIGIT(S)
014743     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
```

```
                                                TST     CLKSTA          ;KW11-P CLOCK?
                                                BGT     1$              ;YES--START TEST
                                                JMP     EXIT14          ;NO--JUMP TO EXIT14
                                        001122  1$:     MOV     #.,$LPADR       ;SET THE LOOP ADDRESS
                                                        JSR     R0,SRCH00       ;DO A MASSBUS INIT & RECAL
                                                        BR      2$              ;RETURN HERE IF NO ERROR
                                                        JMP     EXIT14          ;RETURN HERE ON ERROR
                                        001122  2$:     MOV     #.,$LPADR       ;ERROR LOOP ADDRESS
                                        045460          MOVB    #SEEK,DPB.A+2   ;SEEK=COMMAND
                                        045466          CLR     DPB.A+10        ;USE TRACK 0 & SECTOR 0
                                        045470          MOV     FC,DPB.A+12     ;STARTING CYLINDER
                                        001346          MOV     #EXIT14,BYPASS  ;GO TO EXIT14 IF ERROR
                                        027132          JSR     R0,CALL.A       ;GO EXECUTE THE COMMAND
                                        001656          MOV     #T11,R3         ;PARAMETER POINTER
                                        001222          MOV     #EXIT14,$ESCAPE ;;ESCAPE TO EXIT14 ON ERROR
                                                TEST14:
                                        001100          MOV     #STACK,SP       ;SETUP STACK
                                        000200          MOV     #128.,R1        ;REPEAT '''FC'-'LC'-'FC''' 128 TIMES
                                        031274          JSR     PC,STRTMR       ;INIT. THE COUNTERS
                                        164404          MOV     #7$,@PKV        ;SET UP VECTOR IN CASE OF OVERFLOW
                                        021770          MOV     #DORTI,@RMVEC   ;SETUP RM VECTOR
                                                1$:     CLR     @PKB            ;START COUNT AT ZERO
                                        000034          MOV     LC,RMDC(R4)     ;'MIDDLE' CYLINDER
                                        000000          MOV     #SEEK,RMCS1(R4) ;START A SEEK
                                        164354          MOV     #131,@PKCS      ;START THE CLOCK
                                                        WAIT                    ;WAIT ON INTERRUPT
                                        164344          BIC     #101,@PKCS      ;STOP CLOCK
                                        000012          BIT     #ERR,RMDS(R4)   ;ERR=1?
                                                        BEQ     2$              ;NO--BRANCH
                                                        SAVREG                  ;SAVE R0-R5
                                        045536          MOV     #DTADPB,R2      ;DPB POINTER
                                        044310          JSR     PC,SVRH70       ;SAVE ALL THE RH/RM REGISTERS
                                        026072          JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
                                                        RESREG                  ;RESTORE R0-R5
                                                        EMT     17
                                                2$:     CLR     R5              ;SET UP/DOWN SWITCH TO UP
                                        031336          JSR     PC,COUNT        ;UPDATE THE COUNT
                                        164302          CLR     @PKB            ;START THE COUNT AT ZERO
                                        000034          MOV     FC,RMDC(R4)     ;BEGINNING CYLINDER
                                        000000          MOV     #SEEK,RMCS1(R4) ;START A SEEK
                                        164256          MOV     #131,@PKCS      ;START THE CLOCK
                                                        WAIT                    ;WAIT ON INTERRUPT
                                        164246          BIC     #101,@PKCS      ;STOP THE CLOCK
                                        000012          BIT     #ERR,RMDS(R4)   ;ERR=1?
                                                        BEQ     3$              ;NO--BRANCH
                                                        SAVREG                  ;SAVE R0-R5
                                        045536          MOV     #DTADPB,R2      ;DPB POINTER
                                        044310          JSR     PC,SVRH70       ;SAVE ALL THE RH/RM REGISTERS
                                        026072          JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
                                                        RESREG                  ;RESTORE R0-R5
                                                        EMT     17
                                        177777  3$:     MOV     #-1,R5          ;SET UP/DOWN SWITCH TO DOWN
                                        031336          JSR     PC,COUNT        ;UPDATE THE COUNT
                                                        DEC     R1              ;DONE?
                                                        BGT     1$              ;NO--BRANCH
                                                        BR      8$              ;YES--EXIT
```

```
                                                                    I 7
    282 015304  042777  000101  164170  7$:     BIC     #101,@PKCS      ;STOP THE CLOCK
    283 015312  005037  177776                  CLR     PS              ;DROP THE PRIORITY
    284 015316  012600                          MOV     (SP)+,R0        ;PC OF WAIT+2
    285 015320  005726                          TST     (SP)+           ;POP THE PS FROM THE STACK
    286 015322  104412                          SAVREG                  ;SAVE R0-R5
        015324  012702  045536                  MOV     #DTADPB,R2      ;DPB POINTER
        015330  004737  044310                  JSR     PC,SVRH70       ;SAVE ALL THE RH/RM REGISTERS
        015334  004737  026072                  JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
        015340  104413                          RESREG                  ;RESTORE R0-R5
    287 015342  104020                          EMT     20
    288 015344  004737  026072          8$:     JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
    289 015350  004737  026156                  JSR     PC,ST.CLK       ;INITIALIZE THE CLOCK
    290 015354  012777  041740  021506          MOV     #ISR,@RMVEC     ;RESTORE RH/RM INT. VECTOR
    291 015362  004037  031576                  JSR     R0,TYPTIM       ;GO TYPE THE TIMES
        015366  001656                          T11                     ;POINTER
    292 015370  004037  031470                  JSR     R0,SPTYP
    293 015374  001734                          SP11
    294 015376  000004                  EXIT14: SCOPE                   ;CALL SCOPE ROUTINE
    295
    304
    305                                 ;;*****************************************************************
                                        ;*TEST 15       MAXIMUM SEEK TIMING TEST
                                        ;*THIS TEST WILL COMMAND A FORWARD SEEK FROM CYLINDER 'FC' TO
                                        ;*CYLINDER 'LC', THEN A REVERSE SEEK FROM CYLINDER 'LC' TO
                                        ;*CYLINDER 'FC'.  BOTH SEEKS ARE TIMED AND CHECKED TO ENSURE
                                        ;*THEY ARE WITHIN THE TOLERANCE ALLOWED FOR THE MAXIMUM SEEK
                                        ;*TIME.  THIS SEQUENCE IS REPEATED 128 TIMES (FOR A TOTAL OF
                                        ;*256 SEEKS).  THE MAXIMUM SEEK TIME MUST BE LESS THAN 54 MS.
                                        ;;*****************************************************************
    015400                              TST15:
    015400  000240                              NOP
    015402  033737  001560  001330              BIT     BITS+<15*2>,TSTNMS      ;DO THIS TEST?
    015410  001002                              BNE     .+6             ;BR IF YES
    015412  000137  016140                      JMP     TST16           ;NO--JUMP TO TEST16

    015416  012737  000015  001116              MOV     #15,$TSTNM      ;SET TEST #15 AND CLEAR ($ERFLG)
    015424  004737  026672                      JSR     PC,LODPRM       ;LOAD THE PARMETERS FOR THE TEST
    015430  012737  016140  001124              MOV     #TST16,$LPERR   ;SETUP THE ERROR LOOP ADDRESS
    015436  013737  002324  001220              MOV     RPT,$TIMES      ;GET THE ITERATION COUNT
    015444  112737  000031  001131              MOVB    #25.,$ERMAX     ;MAX ERRORS ALLOWED FOR TEST
    015452  012737  000015  001240              MOV     #15,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
306
    015460  032777  010000  163466              BIT     #SW12,@SWR      ;INHIBIT TYPING TEST NUMBER ?
    015466  001406                              BEQ     .+16            ;BR IF YES
    015470  104401  046253                      TYPE    ,MSGTST         ;TYPE 'TEST'
    015474  013746  001240                      MOV     $TESTN,-(SP)    ;;SAVE $TESTN FOR TYPEOUT
    015500  104403                              TYPOS                   ;;GO TYPE--OCTAL ASCII
    015502     003                              .BYTE   3               ;;TYPE 3 DIGIT(S)
    015503     000                              .BYTE   0               ;;SUPPRESS LEADING ZEROS

    307 015504  005737  001340                  TST     CLKSTA          ;KW11-P CLOCK
    308 015510  003002                          BGT     1$              ;YES--START TEST
    311 015512  000137  016136                  JMP     EXIT15          ;NO--JUMP TO EXIT15
    312 015516  012737  015516  001122  1$:     MOV     #.,$LPADR       ;SETUP THE LOOP ADDRESS
    313 015524  004037  031110                  JSR     R0,SRCH00       ;DO A MASSBUS INIT & RECAL
    314 015530  000402                          BR      2$              ;RETURN HERE IF NO ERROR
    326 015532  000137  016136                  JMP     EXIT15          ;RETURN HERE ON ERROR
```

```
        015536  012737  015536  001122  2$:     MOV     #.,$LPADR       ;ERROR LOOP ADDRESS
        015544  112737  000105  045460          MOVB    #SEEK,DPB.A+2   ;SEEK=COMMAND
        015552  005037  045466                  CLR     DPB.A+10        ;USE TRACK 0 & SECTOR 0
        015556  013737  002326  045470          MOV     FC,DPB.A+12     ;STARTING CYLINDER
        015564  012737  016136  001346          MOV     #EXIT15,BYPASS  ;GO TO EXIT15 IF ERROR
        015572  004037  027132                  JSR     R0,CALL.A       ;GO EXECUTE THE COMMAND
        015576  012703  001666                  MOV     #T12,R3         ;PARAMETER POINTER
        015602  012737  016136  001222          MOV     #EXIT15,$ESCAPE ;;ESCAPE TO EXIT15 ON ERROR
        015610                          TEST15:
    327 015610  012706  001100                  MOV     #STACK,SP       ;SETUP STACK
    328 015614  012701  000200                  MOV     #128.,R1        ;REPEAT '"FC'-'LC'-'FC"' 128 TIMES
    329 015620  004737  031274                  JSR     PC,STRTMR       ;INIT. THE TIMERS
    330 015624  012777  016044  163644          MOV     #7$,@PKV        ;SETUP VECTOR IN CASE OF OVERFLOW
    331 015632  012777  031272  021230          MOV     #DORTI,@RMVEC   ;SETUP RM VECTOR
    332 015640  005077  163640          1$:     CLR     @PKB            ;START COUNTING FROM ZERO
    333 015644  013764  002330  000034          MOV     LC,RMDC(R4)     ;MAXIMUM CYLINDER
    334 015652  012764  000105  000000          MOV     #SEEK,RMCS1(R4) ;START A SEEK
    335 015660  012777  000131  163614          MOV     #131,@PKCS      ;START THE CLOCK
    336 015666  000001                          WAIT                    ;WAIT ON INTERRUPT
    337 015670  042777  000101  163604          BIC     #101,@PKCS      ;STOP THE CLOCK
    338 015676  032764  040000  000012          BIT     #ERR,RMDS(R4)   ;ERR=1?
    339 015704  001411                          BEQ     2$              ;NO--BRANCH
    340 015706  104412                          SAVREG                  ;SAVE R0-R5
        015710  012702  045536                  MOV     #DTADPB,R2      ;DPB POINTER
        015714  004737  044310                  JSR     PC,SVRH70       ;SAVE ALL THE RH/RM REGISTERS
        015720  004737  026072                  JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
        015724  104413                          RESREG                  ;RESTORE R0-R5
    341 015726  104017                          EMT     17
    342 015730  005005                  2$:     CLR     R5              ;SET THE UP/DOWN SWITCH TO UP
    343 015732  004737  031336                  JSR     PC,COUNT        ;UP THE COUNT
    344 015736  005077  163542                  CLR     @PKB            ;START COUNT AT ZERO
    345 015742  013764  002326  000034          MOV     FC,RMDC(R4)     ;BEGINNING CYLINDER
    346 015750  012764  000105  000000          MOV     #SEEK,RMCS1(R4) ;START A SEEK
    347 015756  012777  000131  163516          MOV     #131,@PKCS      ;START THE CLOCK
    348 015764  000001                          WAIT                    ;WAIT ON INTERRUPT
    349 015766  042777  000101  163506          BIC     #101,@PKCS      ;STOP THE CLOCK
    350 015774  032764  040000  000012          BIT     #ERR,RMDS(R4)   ;"ERR"=1?
    351 016002  001411                          BEQ     3$              ;NO--BRANCH
    352 016004  104412                          SAVREG                  ;SAVE R0-R5
        016006  012702  045536                  MOV     #DTADPB,R2      ;DPB POINTER
        016012  004737  044310                  JSR     PC,SVRH70       ;SAVE ALL THE RH/RM REGISTERS
        016016  004737  026072                  JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
        016022  104413                          RESREG                  ;RESTORE R0-R5
    353 016024  104017                          EMT     17
    354 016026  012705  177777          3$:     MOV     #-1,R5          ;SET THE UP/DOWN SWITCH TO DOWN
    355 016032  004737  031336                  JSR     PC,COUNT        ;UPDATE THE COUNT
    356 016036  005301                          DEC     R1              ;DONE?
    357 016040  003277                          BGT     1$              ;NO--BRANCH
    358 016042  000420                          BR      8$              ;YES--EXIT
    359 016044  042777  000101  163430  7$:     BIC     #101,@PKCS      ;STOP THE CLOCK
    360 016052  005037  177776                  CLR     PS              ;DROP THE PRIORITY
    361 016056  012600                          MOV     (SP)+,R0        ;PC OF WAIT+2
    362 016060  005726                          TST     (SP)+           ;POP THE PS FROM THE STACK
    363 016062  104412                          SAVREG                  ;SAVE R0-R5
        016064  012702  045536                  MOV     #DTADPB,R2      ;DPB POINTER
        016070  004737  044310                  JSR     PC,SVRH70       ;SAVE ALL THE RH/RM REGISTERS
        016074  004737  026072                  JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
```

```
        016100  104413                          RESREG                  ;RESTORE R0-R5
    364 016102  104020                          EMT     20
    365 016104  004737  026072          8$:     JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
    366 016110  004737  026156                  JSR     PC,ST.CLK       ;INITIALIZE THE CLOCK
    367 016114  012777  041740  020746          MOV     #ISR,@RMVEC     ;RESTORE RH/RM INT. VECTOR
    368 016122  004037  031576                  JSR     R0,TYPTIM       ;GO TYPE THE TIMES
        016126  001666                          T12                     ;POINTER
    369 016130  004037  031470                  JSR     R0,SPTYP        ;
    370 016134  001742                          SP12
    371 016136  000004                  EXIT15: SCOPE                   ;CALL SCOPE ROUTINE
    372
```

```
                        .SBTTL  ADDRESSING TESTS
                        ;;////////////////////////////////////////////////////////////////
                        ;*THE ADDRESSING TESTS ENSURES PROPER OPERATION OF THE TRACK
                        ;*AND SECTOR ADDRESS CIRCUITRY.  BOTH ADDRESSING TESTS WILL
                        ;*BE PERFORMED ON CYLINDER FC.
                        ;;////////////////////////////////////////////////////////////////

                        ;;****************************************************************
                        ;*TEST 16       SECTOR ADDRESSING TEST
                        ;*THIS TEST WRITES DATA INTO ALL SECTORS OF TRACK "FT".  THE
                        ;*DATA WILL BE 256 WORDS OF THE SECTOR ADDRESS OF THE SECTOR
                        ;*BEING WRITTEN.  A WRITE CHECK IS PERFORMED, THE BUFFER IS
                        ;*CLEARED (TO 177400) AND THE DATA IS READ AND COMPARED.  THEN
                        ;*SECTOR 0 IS REWRITTEN AND SECTORS 0 - 31 ARE WRITE CHECKED.
                        ;*THEN SECTOR 1 IS REWRITTEN AND SECTORS 0 - 31 ARE WRITE CHECKED.
                        ;*THIS REWRITE AND WRITE CHECK PROCEDURE IS CONTINUED UP THROUGH
                        ;*REWRITE SECTOR 31 AND WRITE CHECK SECTORS 0-31.
                        ;;****************************************************************
016140                  TST16:
016140  000240                  NOP
016142  033737  001562  001330  BIT     BITS+<16*2>,TSTNMS      ;DO THIS TEST?
016150  001002                  BNE     .+6                     ;BR IF YES
016152  000137  016562          JMP     TST17                   ;NO--JUMP TO TEST17

016156  012737  000016  001116  MOV     #16,$TSTNM              ;SET TEST #16 AND CLEAR ($ERFLG)
016164  004737  026672          JSR     PC,LODPRM               ;LOAD THE PARMETERS FOR THE TEST
016170  012737  016260  001124  MOV     #TEST16,$LPERR          ;SETUP THE LOOP ON ERROR ADDRESS
016176  013737  002324  001220  MOV     RPT,$TIMES              ;GET THE ITERATION COUNT
016204  113737  001462  001131  MOVB    ERR.CT,$ERMAX           ;MAX ERRORS ALLOWED FOR TEST
016212  012737  000016  001240  MOV     #16,$TESTN              ;;SET TEST NUMBER IN APT MAIL BOX

016220  032777  010000  162726  BIT     #SW12,@SWR              ;INHIBIT TYPING TEST NUMBER ?
016226  001406                  BEQ     .+16                    ;BR IF YES
016230  104401  046253          TYPE    ,MSGTST                 ;TYPE 'TEST'
016234  013746  001240          MOV     $TESTN,-(SP)            ;;SAVE $TESTN FOR TYPEOUT
016240  104403                  TYPOS                           ;;GO TYPE--OCTAL ASCII
016242     003                  .BYTE   3                       ;;TYPE 3 DIGIT(S)
016243     000                  .BYTE   0                       ;;SUPPRESS LEADING ZEROS

016244  012737  016560  001346  MOV     #EXIT16,BYPASS
016252  012737  016260  001122  MOV     #TEST16,$LPADR          ;SETUP THE LOOP ADDRESS
016260                  TEST16:
016260  012706  001100          MOV     #STACK,SP               ;SET THE STACK POINTER
016264  004737  032256          JSR     PC,FILBUF               ;FILL THE BUFFER WITH SECTOR ADDRESSES
016270  013737  002326  045550  MOV     FC,DTADPB+12            ;CYLINDER
016276  113737  002334  045547  MOVB    FT,DTADPB+11            ;TRACK
016304  105037  045546          CLRB    DTADPB+10               ;SECTOR
016310  013737  001450  045542  MOV     TRCKWC,DTADPB+4         ;WORD COUNT
016316  012737  052776  045544  MOV     #BUFFER,DTADPB+6                ;BUFFER ADDRESS
016324  012737  016324  001124  MOV     #.,$LPERR               ;SETUP THE ERROR LOOP ADDRESS
016332  012706  001100          MOV     #STACK,SP               ;LOAD THE STACK POINTER
016336  012737  000161  045540  MOV     #WRITE,DTADPB+2         ;COMMAND=WRITE DATA
016344  004037  027670          JSR     R0,DRVCAL               ;START A DATA TRANSFER
016350  012737  000151  045540  MOV     #WRCKD,DTADPB+2         ;COMMAND=WRITE CHECK DATA
016356  012737  016356  001124  MOV     #.,$LPERR               ;SETUP THE ERROR LOOP ADDRESS
```

```
                016364  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
     39         016370  004037  027670                  JSR     R0,DRVCAL       ;START A DATA TRANSFER
     40         016374  012737  016374  001124          MOV     #.,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
                016402  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
     41         016406  004037  032314                  JSR     R0,CLRBUF       ;CLEAR BUFFER
     42         016412  012737  000171  045540          MOV     #READ,DTADPB+2  ;COMMAND = READ
     43         016420  004037  027670                  JSR     R0,DRVCAL       ;START A DATA TRANSFER
     44         016424  004037  032362                  JSR     R0,CKSCTR       ;CHECK THE SECTOR DATA READ
     45         016430  012700  052776                  MOV     #BUFFER,R0      ;BUFFER ADDRESS
     46         016434  005001                          CLR     R1              ;FIRST SECTOR
     47         016436  012737  016436  001124          MOV     #.,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
                016444  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
     48         016450  012737  000161  045540  1$:     MOV     #WRITE,DTADPB+2 ;COMMAND=WRITE DATA
     49         016456  012737  177400  045542          MOV     #SCTRWC,DTADPB+4        ;WORD COUNT
     50         016464  010037  045544                  MOV     R0,DTADPB+6     ;BUFFER ADDRESS
     51         016470  110137  045546                  MOVB    R1,DTADPB+10    ;SECTOR
     52         016474  004037  027670                  JSR     R0,DRVCAL       ;START A DATA TRANSFER
     53         016500  012737  016500  001124          MOV     #.,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
                016506  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
     54         016512  012737  000151  045540          MOV     #WRCKD,DTADPB+2 ;COMMAND=WRITE CHECK DATA
     55         016520  013737  001450  045542          MOV     TRCKWC,DTADPB+4 ;WORD COUNT
     56         016526  012737  052776  045544          MOV     #BUFFER,DTADPB+6        ;BUFFER ADDRESS
     57         016534  105037  045546                  CLRB    DTADPB+10       ;SECTOR
     58         016540  004037  027670                  JSR     R0,DRVCAL       ;START A DATA TRANSFER
     59         016544  062700  001000                  ADD     #512.,R0        ;MOVE TO NEXT SECTOR
     60         016550  005201                          INC     R1
     61         016552  023701  002456                  CMP     PRMLMT+24,R1    ;DONE?
     62         016556  103334                          BHIS    1$              ;NO--BRANCH
     63         016560  000004                  EXIT16: SCOPE                   ;CALL SCOPE ROUTINE
     64
     76
     77                                         ;;*****************************************************************
                                                ;*TEST 17       TRACK ADDRESSING TEST
                                                ;*THIS TEST WILL WRITE DATA IN THE FORM OF TRACK ADDRESSES
                                                ;*IN CYLINDER "FC" SECTOR "FS" OF EVERY TRACK WITH EACH TRACK
                                                ;*GETTING ITS OWN TRACK ADDRESS.
                                                ;*A WRITE CHECK IS THEN PERFORMED ON EACH TRACK TO ENSURE
                                                ;*THE DATA IS VALID.  THEN TRACK 0 IS REWRITTEN AND TRACK 1
                                                ;*THROUGH LAST TRACK IS WRITE CHECKED.  THEN TRACK 1 IS
                                                ;*REWRITTEN AND TRACK 2 THROUGH LAST TRACK IS WRITE CHECKED.
                                                ;*THIS PROCEDURE IS CONTINUED UP THROUGH REWRITING NEXT TO LAST
                                                ;*TRACK AND WRITE CHECKING LAST TRACK.
                                                ;;*****************************************************************
                016562                  TST17:
                016562  000240                    NOP
                016564  033737  001564  001330      BIT     BITS+<17*2>,TSTNMS      ;DO THIS TEST?   .
                016572  001002                      BNE     .+6             ;BR IF YES
                016574  000137  017226              JMP     TST20           ;NO--JUMP TO TEST20

                016600  012737  000017  001116      MOV     #17,$TSTNM      ;SET TEST #17 AND CLEAR ($ERFLG)
                016606  004737  026672              JSR     PC,LODPRM       ;LOAD THE PARMETERS FOR THE TEST
                016612  012737  016702  001124      MOV     #TEST17,$LPERR  ;SETUP THE LOOP ON ERROR ADDRESS
                016620  013737  002324  001220      MOV     RPT,$TIMES      ;GET THE ITERATION COUNT
                016626  113737  001462  001131      MOVB    ERR.CT,$ERMAX   ;MAX ERRORS ALLOWED FOR TEST
                016634  012737  000017  001240      MOV     #17,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
     78
                016642  032777  010000  162304      BIT     #SW12,@SWR      ;INHIBIT TYPING TEST NUMBER ?
```

```
                                                                    N  7
        016650  001406                          BEQ     .+16            ;BR IF YES
        016652  104401  046253                  TYPE    ,MSGTST         ;TYPE 'TEST'
        016656  013746  001240                  MOV     $TESTN,-(SP)    ;;SAVE $TESTN FOR TYPEOUT
        016662  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
        016664     003                          .BYTE   3               ;;TYPE 3 DIGIT(S)
        016665     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS

     83 016666  012737  017224  001346          MOV     #EXIT17,BYPASS
        016674  012737  016702  001122          MOV     #TEST17,$LPADR  ;SETUP THE LOOP ADDRESS
        016702                          TEST17:
     84 016702  012706  001100                  MOV     #STACK,SP       ;SET THE STACK POINTER
     85 016706  004737  032256                  JSR     PC,FILBUF       ;FILL THE BUFFER WITH TRACK ADDRESS
     86 016712  012737  000161  045540          MOV     #WRITE,DTADPB+2 ;COMMAND=WRITE DATA
     87 016720  013737  002326  045550          MOV     FC,DTADPB+12    ;CYLINDER
     88 016726  113737  002342  045546          MOVB    FS,DTADPB+10    ;SECTOR
     89 016734  012737  177400  045542          MOV     #SCTRWC,DTADPB+4        ;WORD COUNT
     90 016742  012737  052776  045544          MOV     #BUFFER,DTADPB+6        ;BUFFER ADDRESS
     91 016750  005000                          CLR     R0              ;TRACK=0
     92 016752  012737  016752  001124          MOV     #.,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
        016760  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
     93
     94 016764  110037  045547          1$:     MOVB    R0,DTADPB+11    ;TRACK ADDRESS
     95 016770  004037  027670                  JSR     R0,DRVCAL       ;START A DATA TRANSFER
     96 016774  062737  001000  045544          ADD     #256.*2.,DTADPB+6       ;UPDATE BUFFER ADDRESS
     97 017002  005200                          INC     R0              ;UPDATE TRACK NUMBER
     98 017004  023700  001372                  CMP     LSTRK,R0        ;OUT OF TRACKS?
     99 017010  002365                          BGE     1$              ;NO--BRANCH
    100 017012  012737  052776  045544          MOV     #BUFFER,DTADPB+6        ;BUFFER ADDRESS
    101 017020  005000                          CLR     R0
    102 017022  012737  017022  001124        * MOV     #.,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
        017030  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
    103
    104 017034  012737  000151  045540          MOV     #WRCKD,DTADPB+2 ;COMMAND=WRITE CHECK
    105 017042  110037  045547          2$:     MOVB    R0,DTADPB+11    ;TRACK ADDRESS
    106 017046  004037  027670                  JSR     R0,DRVCAL       ;START A DATA TRANSFER
    107 017052  062737  001000  045544          ADD     #256.*2.,DTADPB+6       ;UPDATE BUFFER ADDRESS
    108 017060  005200                          INC     R0              ;UPDATE TRACK NUMBER
    109 017062  023700  001372                  CMP     LSTRK,R0        ;OUT OF TRACKS?
    110 017066  002365                          BGE     2$              ;NO--BRANCH
    111 017070  005000                          CLR     R0              ;FIRST TRACK ADDRESS
    112
    113 017072  110037  045547          3$:     MOVB    R0,DTADPB+11    ;TRACK
    114 017076  010001                          MOV     R0,R1           ;FORM BUFFER ADDRESS
    115 017100  012737  052776  045544          MOV     #BUFFER,DTADPB+6        ;BUFFER ADDRESS
    116 017106  005301                  4$:     DEC     R1
    117 017110  002411                          BLT     5$
    118 017112  062737  001000  045544          ADD     #256.*2.,DTADPB+6
    119 017120  000772                          BR      4$
    120 017122  012737  017122  001124          MOV     #.,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
        017130  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
    121 017134  012737  000161  045540  5$:     MOV     #WRITE,DTADPB+2 ;COMMAND=WRITE DATA
    122 017142  004037  027670                  JSR     R0,DRVCAL       ;START A DATA TRANSFER
    123
    124 017146  062737  001000  045544  6$:     ADD     #256.*2.,DTADPB+6       ;UPDATE BUFFER ADDRESS
    125 017154  105237  045547                  INCB    DTADPB+11       ;MOVE TO NEXT TRACK
    126 017160  012737  017160  001124          MOV     #.,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
        017166  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
```

```
                                        MOV     #WRCKD,DTADPB+2 ;COMMAND=WRITE CHECK DATA
                                        JSR     R0,DRVCAL       ;START A DATA TRANSFER
                                        CMPB    LSTRK,DTADPB+11 ;OUT OF TRACKS?
                                        BGT     6$              ;NO--BRANCH
                                        INC     R0              ;NEXT TRACK TO WRITE
                                        CMP     LSTRK,R0        ;OUT OF TRACKS?
                                        BGT     3$              ;NO--BRANCH
                                EXIT17: SCOPE                   ;CALL SCOPE ROUTINE


                                ;;*******************************************************************
                                ;*TEST 20       DATA TEST
                                ;*THIS TEST PERFORMS DATA STORAGE AND RETRIEVAL ON CYLINDERS
                                ;*"FC" THROUGH "LC" BY THE INCREMENT "IC" USING THE DATA PATTERNS
                                ;*SPECIFIED. THE FOLLOWING SEQUENCE OCCURS FOR EACH CYLINDER:
                                ;* (1.  SET "NT" TO "FT" THEN REPEAT 2-4 UNTIL "NT" > "LT"
                                ;* (2.  WRITE THEN WRITE CHECK "FS" THROUGH "LS" OF TRACK "NT"
                                ;* (3.  READ THEN SOFTWARE COMPARE "FS" THROUGH "LS" OF TRACK "NT"
                                ;* (4.  INCREMENT "NT" BY "IT"
                                ;* (5.  REPEAT STEPS 1-4 FOR EACH DATA PATTERN
                                ;* (6.  REPEAT STEPS 1-5 FOR "FC" THROUGH "LC" ADVANCING BY "IC"
                                ;*
                                ;*IF A WRITE CHECK ERROR OCCURS THE ERROR IS REPORTED AND
                                ;*THE TRACK IN ERROR IS REWRITTEN AND CHECKED. THIS CHECK IS
                                ;*ACCOMPLISHED BY PERFORMING TWO(2) SUCCESSIVE ERROR FREE
                                ;*WRITE CHECKS. IF THE CHECK FAILS THE ERROR IS REPORTED AS
                                ;*FATAL AND NO READ OCCURS.
                                ;*FS DEFAULTS TO 1 AND LS DEFAULTS TO 0
                                ;*PAT DEFAULTS TO 177777 (ALL POSSIBLE PATTERNS)
                                ;;*******************************************************************
017226                          TST20:
017226  000240                          NOP
017230  033737  001526  001332          BIT     BITS+<20*2-40>,TSTNMS+2 ;DO THIS TEST ?
017236  001002                          BNE     .+6             ;BR IF YES
017240  000137  020006                  JMP     TST21           ;NO--JUMP TO TEST21

017244  012737  000020  001116          MOV     #20,$TSTNM      ;SET TEST #20 AND CLEAR ($ERFLG)
017252  004737  026672                  JSR     PC,LODPRM       ;LOAD THE PARMETERS FOR THE TEST
017256  012737  017454  001124          MOV     #TEST20,$LPERR  ;SETUP THE LOOP ON ERROR ADDRESS
017264  013737  002324  001220          MOV     RPT,$TIMES      ;GET THE ITERATION COUNT
017272  113737  001462  001131          MOVB    ERR.CT,$ERMAX   ;MAX ERRORS ALLOWED FOR TEST
017300  012737  000020  001240          MOV     #20,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX

017306  032777  010000  161640          BIT     #SW12,@SWR      ;INHIBIT TYPING TEST NUMBER ?
017314  001406                          BEQ     .+16            ;BR IF YES
017316  104401  046253                  TYPE    ,MSGTST         ;TYPE 'TEST'
017322  013746  001240                  MOV     $TESTN,-(SP)    ;;SAVE $TESTN FOR TYPEOUT
017326  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
017330     003                          .BYTE   3               ;;TYPE 3 DIGIT(S)
017331     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS


                                        ;SET UP THE TRACK WORD COUNT FOR DATA TRANFER

017332  012737  017332  001122          MOV     #.,$LPADR       ;SETUP THE LOOP ADDRESS
017340  005000                          CLR     R0              ;CLEAR SWITCH
017342  005004                          CLR     R4              ;FORM WORD COUNT IN R4
```

```
                                                MOV     LS,R1           ;LOAD LAST SECTOR
                                                SUB     FS,R1           ;COMPARE WITH FIRST SECTOR
                                                BGE     1$              ;SKIP NEXT IF FS < OR = LS

                                                ADD     PRMLMT+24,R1    ;ADD MAXIMUM SECTOR ADDRESS TO
                                                INC     R1              ;MAKE THE DIFFERENCE POSITIVE
                                                COM     R0              ;SET SWITCH FOR FS > LS

                                        1$:     ADD     #256.,R4        ;WORDS/SECTOR
                                                DEC     R1              ;LS - FS SECTORS MINUS ONE
                                                BGE     1$              ;INCR. WORD COUNT IF NO. SECTORS STILL +
                                                NEG     R4              ;FORM NEGATIVE WORD COUNT
                                                MOV     R4,R5           ;COPY NORMAL WORD COUNT INTO SMALL WC
                                                TST     R0              ;FS > LS SWITCH SET?
                                                BEQ     3$              ;NO, SKIP NEXT

                                                CLR     R5              ;FORM WORD COUNT FOR FS > LS
                                                MOV     PRMLMT+24,R1    ;MAX SECTOR ADDRESS
                                                SUB     FS,R1           ;SUBTRACT THE FIRST SECTOR
                                        2$:     ADD     #256.,R5        ;WORDS/SECTOR
                                                DEC     R1              ;NO SECTORS MINUS ONE
                                                BGE     2$              ;INCR. WORD COUNT IF NO. SECTORS STILL +
                                                NEG     R5              ;FORM NEGATIVE WORD COUNT FOR THIS CASE


                                                ;SET UP FOR DATA TRANSFERS AND PATTERN VARIATIONS

                                        3$:     MOVB    FS,DTADPB+10    ;SECTOR
                                                MOV     #BUFFER,DTADPB+6        ;DATA BUFFER
                                                MOV     #EXIT20,BYPASS
                                        TEST20:
                                                MOV     #STACK,SP       ;LOAD THE STACK POINTER
                                                CLR     WCEFLG  ;CLEAR THE WRITE CHECK ERROR FLAG
                                                MOV     FC,R1           ;PICKUP FIRST CYLINDER
                                                BR      2$              ;SKIP PATTERN SET-UP FIRST TIME THRU

                                        1$:     TST     (R0)+           ;MOVE TO NEXT DATA PATTERN
                                                CMP     #16.*2.,R0      ;OUT OF PATTERNS?
                                                BGT     3$              ;NO, STAY ON THIS CYLINDER UNTIL DONE
                                                JSR     R0,INCCYL       ;MOVE TO NEXT CYLINDER
                                                BR      EXIT20          ;OUT OF CYLINDERS

                                                ;DO NEXT CYLINDER

                                        2$:     CLR     R0              ;START WITH PATTERN 0
                                        3$:     BIT     BITS(R0),PAT            ;THIS PATTERN SELECTED?
                                                BEQ     1$              ;NO, GO BACK AND GET ONE THAT WAS
                                                MOV     FT,R2           ;FIRST TRACK
                                                MOV     R1,DTADPB+12    ;LOAD THE CYLINDER
                                        4$:     MOVB    R2,DTADPB+11    ;LOAD THE TRACK

                                                CMP     R1,#822.        ;CHECK FOR LAST DISK CYLINDER
                                                BNE     10$             ;SKIP LAST TRACK CHECK IF NOT
                                                CMP     R2,LSTRK        ;LAST DISK TRACK ?
                                                BEQ     EXIT20          ;DON'T TEST LAST TRACK AS IT HAS BAD
                                        10$:                            ;BLOCK INFORMATION STORED ON IT

                                                MOV     R4,DTADPB+4     ;LOAD THE WORD COUNT
```

```
                                                                    D 8
 223 017556  023701  002330                    CMP     LC,R1           ;LAST DISK CYLINDER TO TEST ?
 224 017562  003005                            BGT     5$              ;NO, SKIP TRACK CHECK
 225 017564  023702  001372                    CMP     LSTRK,R2        ;LAST DISK TRACK TO TEST ?
 226 017570  002002                            BGE     5$              ;NO, SKIP NEXT
 227 017572  010537  045542                    MOV     R5,DTADPB+4     ;SHORT WORD COUNT
 228 017576  017703  161352          5$:       MOV     @SWR,R3         ;INHIBIT WRITE AND
 229 017602  005103                            COM     R3              ;WRITE CHECK?
 230 017604  032703  000030                    BIT     #SW04!SW03,R3
 231 017610  001436                            BEQ     7$              ;YES--BRANCH
 232 017612  004737  032732                    JSR     PC,SETBUF       ;MOVE DATA PATTERN INTO THE BUFFER
 233 017616  032777  000020  161330            BIT     #SW04,@SWR      ;INHIBIT WRITE?
 234 017624  001012                            BNE     6$              ;YES, DO NEXT PORTION OF TESTING
 235 017626  012737  017626  001124            MOV     #.,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
     017634  012706  001100                    MOV     #STACK,SP       ;LOAD THE STACK POINTER
 236 017640  012737  000161  045540            MOV     #WRITE,DTADPB+2 ;COMMAND=WRITE DATA
 237
 238                                           ;DO A DATA TRANSFER
 239
 240 017646  004037  027670                    JSR     R0,DRVCAL       ;START A DATA TRANSFER
 241 017652  032777  000010  161274  6$:       BIT     #SW03,@SWR      ;INHIBIT WRITE CHECK?
 242 017660  001012                            BNE     7$              ;YES--BRANCH
 243 017662  012737  017662  001124            MOV     #.,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
     017670  012706  001100                    MOV     #STACK,SP       ;LOAD THE STACK POINTER
 244 017674  012737  000151  045540            MOV     #WRCKD,DTADPB+2 ;COMMAND=WRITE CHECK DATA
 245 017702  004037  027670                    JSR     R0,DRVCAL       ;START A DATA TRANSFER
 246 017706  005737  001432          7$:       TST     WCEFLG  ;WRITE CHECK ERROR FLAG SET?
 247 017712  001404                            BEQ     8$              ;NO--BRANCH
 248 017714  032777  000001  161232            BIT     #SW00,@SWR      ;PERFORM READ AFTER FATAL 'WCE'?
 249 017722  001424                            BEQ     9$              ;NO--BRANCH
 250 017724  032777  000004  161222  8$:       BIT     #SW02,@SWR      ;INHIBIT READ DATA AND SOFTWARE COMPARE?
 251 017732  001020                            BNE     9$              ;YES--BRANCH
 252 017734  012737  017734  001124            MOV     #.,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
     017742  012706  001100                    MOV     #STACK,SP       ;LOAD THE STACK POINTER
 253 017746  012737  000171  045540            MOV     #READ,DTADPB+2  ;COMMAND=READ
 254 017754  004037  027670                    JSR     R0,DRVCAL       ;START A DATA TRANSFER
 255 017760  032777  000002  161166            BIT     #SW01,@SWR      ;COMPARE THE DATA?
 256 017766  001002                            BNE     9$              ;NO--BRANCH
 257 017770  004737  033022                    JSR     PC,DATCMP       ;YES--DO IT
 258 017774  004037  032152          9$:       JSR     R0,INCTRK       ;MOVE TO NEXT TRACK
 259 020000  000634                            BR      1$              ;OUT OF TRACKS GO TO NEXT PATTERN
 260 020002  000653                            BR      4$              ;LOOP
 261 020004  000004                  EXIT20:   SCOPE                   ;CALL SCOPE ROUTINE
 262
 287
 288
                                    ;;*****************************************************************
                                    ;*TEST 21       RANDOM ADDRESS AND RANDOM PATTERN TEST
                                    ;*STARTING AT "FC" AND GOING SEQUENTIALLY TO "LC" THE DISK
                                    ;*PACK IS WRITTEN WITH A RANDOM PATTERN. THE FIRST TWO WORDS
                                    ;*OF EACH SECTOR WILL BE THE BASE OF THE RANDOM GENERATOR
                                    ;*FOR THAT SECTOR.

                                    ;*THE TEST THEN PERFORMS THE FOLLOWING SEQUENCE "R" TIMES
                                    ;*"R" DEFAULTS TO 1000.
                                    ;*
                                    ;*1)    GENERATE A RANDOM SECTOR ADDRESS
                                    ;*2)    WRITE A RANDOM PATTERN AT THE ADDRESS
                                    ;*      GENERATED IN 1.
```

```
                                ;*3)    GENERATE A NEW RANDOM SECTOR ADDRESS
                                ;*4)    READ THE SECTOR AT THE ADDRESS
                                ;*      GENERATED IN 3.
                                ;*5)    DO A SOFTWARE CHECK OF THE DATA READ IN 4 AGAINST
                                ;*      THE ORIGINAL RANDOM PACK DATA.
                                ;*6)    DO A WRITE CHECK OF THE DATA WRITTEN IN 2
                                ;*7)    GENERATE A NEW RANDOM SECTOR ADDRESS
                                ;*8)    READ THE SECTOR AT THE ADDRESS
                                ;*      GENERATED IN 7.
                                ;*9)    DO A SOFTWARE CHECK OF THE DATA READ IN 8
                                ;*10)   DO A WRITE CHECK OF THE DATA WRITTEN IN 2
                                ;;*****************************************************************
     020006                     TST21:
     020006  000240                     NOP
     020010  033737  001530  001332     BIT     BITS+<21*2-40>,TSTNMS+2 ;DO THIS TEST ?
     020016  001002                     BNE     .+6             ;BR IF YES
     020020  000137  020624             JMP     TST22           ;NO--JUMP TO TEST22

     020024  012737  000021  001116     MOV     #21,$TSTNM      ;SET TEST #21 AND CLEAR ($ERFLG)
     020032  004737  026672             JSR     PC,LODPRM       ;LOAD THE PARMETERS FOR THE TEST
     020036  012737  020302  001124     MOV     #TEST21,$LPERR  ;SETUP THE LOOP ON ERROR ADDRESS
     020044  013737  002324  001220     MOV     RPT,$TIMES      ;GET THE ITERATION COUNT
     020052  113737  001462  001131     MOVB    ERR.CT,$ERMAX   ;MAX ERRORS ALLOWED FOR TEST
     020060  012737  000021  001240     MOV     #21,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
289
     020066  032777  010000  161060     BIT     #SW12,@SWR      ;INHIBIT TYPING TEST NUMBER ?
     020074  001406                     BEQ     .+16            ;BR IF YES
     020076  104401  046253             TYPE    ,MSGTST         ;TYPE 'TEST'
     020102  013746  001240             MOV     $TESTN,-(SP)    ;;SAVE $TESTN FOR TYPEOUT
     020106  104403                     TYPOS                   ;;GO TYPE--OCTAL ASCII
     020110     003                     .BYTE   3               ;;TYPE 3 DIGIT(S)
     020111     000                     .BYTE   0               ;;SUPPRESS LEADING ZEROS

290  020112  012737  020112  001122     MOV     #.,$LPADR       ;SETUP THE LOOP ADDRESS
293  020120  012737  020622  001346     MOV     #EXIT21,BYPASS
294  020126  012737  176543  025646     MOV     #176543,$HINUM  ;PRIME THE RANDOM NUMBER GENERATOR
295  020134  012737  123456  025650     MOV     #123456,$LONUM
296  020142  013737  002326  045550     MOV     FC,DTADPB+12    ;CYLINDER
297  020150  013737  001450  045542     MOV     TRCKWC,DTADPB+4 ;WORD COUNT FOR 32/30 SECTORS (FULL TRACK)
298  020156  012737  052776  045544     MOV     #BUFFER,DTADPB+6        ;BUFFER ADDRESS
299  020164  012737  000161  045540     MOV     #WRITE,DTADPB+2 ;COMMMAND
300  020172  032737  100000  001314     BIT     #SW15,C.SWR     ;WRITE THE DISK PACK BEFORE TESTING?
301  020200  001027                     BNE     3$              ;NO--BEGIN TESTING
302  020202  004037  033340             JSR     R0,FILRAN       ;FILL DATA BUFFER WITH RANDOM DATA
303  020206  005037  045546     1$:     CLR     DTADPB+10       ;SECTOR AND TRACK
304  020212  012737  020212  001124     MOV     #.,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
     020220  012706  001100             MOV     #STACK,SP       ;LOAD THE STACK POINTER
305  020224                     2$:
     020224  004037  027670             JSR     R0,DRVCAL       ;START A DATA TRANSFER
306  020230  105237  045547             INCB    DTADPB+11       ;NEXT TRACK
307  020234  123737  001372  045547     CMPB    LSTRK,DTADPB+11         ;TIME FOR NEXT CYLINDER ?
308  020242  002370                     BGE     2$              ;NO--DO NEXT TRACK ON THIS CYL.
309  020244  005237  045550             INC     DTADPB+12       ;INCR CYLINDER ADDRESS
310  020250  023737  002330  045550     CMP     LC,DTADPB+12    ;OUT OF CYLINDERS?
311  020256  002353                     BGE     1$              ;NO--CONTINUE SEQUENTIAL RANDOM WRITE
312
313
```

```
                                3$:     MOV     #SCTRWC,DTADPB+4        ;WORD COUNT
                                        MOV     #TEST21,$LPADR
                                        MOV     #TEST21,$LPERR
                                TEST21:
                                        MOV     #STACK,SP       ;SET STACK POINTER
                                        JSR     R0,RANADR       ;GENERATE A RANDOM ADDRESS
                                        MOV     DTADPB+10,SVADR ;SAVE THE TRACK/SECTOR
                                        MOV     DTADPB+12,SVADR+2       ;SAVE THE CYLINDER
                                        MOV     #WRITE,DTADPB+2 ;COMMAND=WRITE DATA
                                        MOV     #BUFFER,R1      ;WRITE BUFFER ADDRESS FOR "RANPAT"
                                        MOV     R1,DTADPB+6     ;INTO THE DATA PARAMETER BLOCK
                                        JSR     R0,RANPAT       ;GENERATE RANDOM 256 WORD PATTERN
                                                                ;AND PUT INTO THE WRITE BUFFER
                                        MOV     #.,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
                                        MOV     #STACK,SP       ;LOAD THE STACK POINTER
                                        JSR     R0,DRVCAL       ;START A DATA TRANSFER

                                        JSR     R0,RANADR       ;GENERATE A NEW RANDOM ADDRESS
                                        MOV     #READ,DTADPB+2  ;COMMAND=READ DATA
                                        MOV     #BUFFER+512.,DTADPB+6   ;READ BUFFER ADDRESS
                                        MOV     #.,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
                                        MOV     #STACK,SP       ;LOAD THE STACK POINTER
                                        JSR     R0,DRVCAL       ;START A DATA TRANSFER
                                        TST     BASFLG          ;IF BAD SECTOR ENCOUNTERED,SKIP NEXT CALL
                                        BMI     .+6             ;DON'T COMPARE DATA
                                        JSR     R0,RANCK        ;SOFTWARE CHECK THE DATA

                                        MOV     SVADR,DTADPB+10 ;GET ADDRESS OF WHERE THE LAST
                                        MOV     SVADR+2,DTADPB+12       ;WRITE WAS PERFORMED
                                        MOV     #WRCKD,DTADPB+2 ;COMMAND=WRITE CHECK DATA
                                        MOV     #BUFFER,DTADPB+6        ;DATA BUFFER ADDRESS FOR HARDWARE
                                                                ;CHECK OF THE DATA
                                        MOV     #.,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
                                        MOV     #STACK,SP       ;LOAD THE STACK POINTER
                                        JSR     R0,DRVCAL       ;START A DATA TRANSFER

                                        JSR     R0,RANADR       ;GENERATE A NEW RANDOM ADDRESS
                                        MOV     #READ,DTADPB+2  ;COMMAND=READ
                                        MOV     #BUFFER+512.,DTADPB+6   ;DATA BUFFER ADDRESS
                                        MOV     #.,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
                                        MOV     #STACK,SP       ;LOAD THE STACK POINTER
                                        JSR     R0,DRVCAL       ;START A DATA TRANSFER
                                        TST     BASFLG          ;ENCOUNTER BAD SECTOR ?
                                        BMI     .+6             ;DON'T COMPARE DATA IF SO
                                        JSR     R0,RANCK        ;SOFTWARE CHECK THE DATA

                                        MOV     SVADR,DTADPB+10 ;GET DISK ADDRESS OF THE
                                        MOV     SVADR+2,DTADPB+12       ;LAST WRITE
                                        MOV     #WRCKD,DTADPB+2 ;COMMAND=WRITE CHECK DATA
                                        MOV     #BUFFER,DTADPB+6        ;DATA BUFFER ADDRESS
                                        MOV     #.,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
                                        MOV     #STACK,SP       ;LOAD THE STACK POINTER
                                        JSR     R0,DRVCAL       ;START A DATA TRANSFER
                                EXIT21: SCOPE                   ;CALL SCOPE ROUTINE


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

```
                                ;*TEST 22       ACCESS TIME ADJUSTMENT TEST
                                ;*THIS TEST PERFORMS SEEKS BETWEEN CYLINDERS 0 & 255 TO ALLOW THE
                                ;*OPERATOR TO ADJUST THE ACCESS TIME ON AN RM05/3/2 USING THE
                                ;*DDU.  THE PROGRAM STALLS APPROXIMATELY 5 SECONDS BETWEEN SEEKS
                                ;*SO THAT THE ACCESS TIME INDICATORS ON THE DDU MAY BE OBSERVED.
                                ;;*****************************************************************
020624                          TST22:
020624  000240                          NOP
020626  033737  001532  001332          BIT     BITS+<22*2-40>,TSTNMS+2 ;DO THIS TEST ?
020634  001002                          BNE     .+6             ;BR IF YES
020636  000137  021020                  JMP     $EOP            ;NO--GO TO THE END OF THE PROGRAM

020642  012737  000022  001116          MOV     #22,$TSTNM      ;SET TEST #22 AND CLEAR ($ERFLG)
020650  004737  026672                  JSR     PC,LODPRM       ;LOAD THE PARMETERS FOR THE TEST
020654  012737  020736  001124          MOV     #TEST22,$LPERR  ;SETUP THE LOOP ON ERROR ADDRESS
020662  013737  002324  001220          MOV     RPT,$TIMES      ;GET THE ITERATION COUNT
020670  112737  000144  001131          MOVB    #100.,$ERMAX    ;MAX ERRORS ALLOWED FOR TEST
020676  012737  000022  001240          MOV     #22,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
374
020704  032777  010000  160242          BIT     #SW12,@SWR      ;INHIBIT TYPING TEST NUMBER ?
020712  001406                          BEQ     .+16            ;BR IF YES
020714  104401  046253                  TYPE    ,MSGTST         ;TYPE 'TEST'
020720  013746  001240                  MOV     $TESTN,-(SP)    ;;SAVE $TESTN FOR TYPEOUT
020724  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
020726     003                          .BYTE   3               ;;TYPE 3 DIGIT(S)
020727     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
378 020730  012737  020736  001122          MOV     #TEST22,$LPADR  ;SETUP THE LOOP ADDRESS
    020736                      TEST22:
379 020736  012706  001100                  MOV     #STACK,SP       ;SETUP THE STACK POINTER
380 020742  013737  002330  045470          MOV     LC,DPB.A+12     ;ENDING CYLINDER
381 020750  112737  000105  045460          MOVB    #SEEK,DPB.A+2   ;SEEK=COMMAND
382 020756  004037  027132                  JSR     R0,CALL.A       ;GO EXECUTE THE COMMAND
383 020762  004037  030664                  JSR     R0,STALL        ;STALL
384 020766  001456                          .WORD   STALL3          ;ADDRESS OF STALL VALUE
385 020770  013737  002326  045470          MOV     FC,DPB.A+12     ;STARTING CYLINDER
386 020776  112737  000105  045460          MOVB    #SEEK,DPB.A+2   ;SEEK=COMMAND
387 021004  004037  027132                  JSR     R0,CALL.A       ;GO EXECUTE THE COMMAND
388 021010  004037  030664                  JSR     R0,STALL        ;STALL
389 021014  001456                          .WORD   STALL3          ;ADDRESS OF STALL VALUE
390 021016  000004                  EXIT22: SCOPE                   ;CALL SCOPE ROUTINE
391
```

```
      1                         .SBTTL  END OF PASS ROUTINE

                                ;;*****************************************************************
                                ;*INCREMENT THE PASS NUMBER ($PASS)
                                ;*INDICATE END-OF-PROGRAM AFTER 8. PASSES THRU THE PROGRAM
                                ;*IF THERES A MONITOR GO TO IT
                                ;*IF THERE ISN'T JUMP TO RESTART

 021020                         $EOP:
 021020  104401  021026                 TYPE    ,65$            ;;TYPE ASCIZ STRING
 021024  000407                         BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <CRLF><LF>/END OF PASS/
 021044                         64$:
 021044  005737  001326                 TST     DRVSEL          ;ANY DRIVES SELECTED?
 021050  001434                         BEQ     1$              ;NO--BRANCH
 021052  104401  021060                 TYPE    ,67$            ;;TYPE ASCIZ STRING
 021056  000405                         BR      66$             ;;GET OVER THE ASCIZ
                                ;;67$:  .ASCIZ  / ON DRIVE/
 021072                         66$:
 021072  013746  001350                 MOV     CHKDRV,-(SP)    ;;SAVE CHKDRV FOR TYPEOUT
 021076  104403                         TYPOS                   ;;GO TYPE--OCTAL ASCII
 021100     002                         .BYTE   2               ;;TYPE 2 DIGIT(S)
 021101     000                         .BYTE   0               ;;SUPPRESS LEADING ZEROS
 021102  104401  021110                 TYPE    ,69$            ;;TYPE ASCIZ STRING
 021106  000412                         BR      68$             ;;GET OVER THE ASCIZ
                                ;;69$:  .ASCIZ  /   ERRORS DETECTED=/
 021134                         68$:
 021134  013746  001126                 MOV     $ERTTL,-(SP)    ;;SAVE $ERTTL FOR TYPEOUT
 021140  104402                         TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 021142  005037  001126         1$:     CLR     $ERTTL          ;ZERO ERROR TOTAL
 021146  005037  001116                 CLR     $TSTNM          ;;ZERO THE TEST NUMBER
 021152  005037  001220                 CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
 021156  005237  001242                 INC     $PASS           ;;INCREMENT THE PASS NUMBER
 021162  042737  100000  001242         BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
 021170  005327                         DEC     (PC)+           ;;LOOP?
 021172  000010                 $EOPCT: .WORD   8.
 021174  003027                         BGT     $DOAGN          ;;YES
 021176  012737                         MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
 021200  000010                 $ENDCT: .WORD   8.
 021202  021172                         $EOPCT
 021204  104401  021212                 TYPE    ,65$            ;;TYPE ASCIZ STRING
 021210  000407                         BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <CRLF>/END OF TEST/
 021230                         64$:
 021230  104401  021260                 TYPE    ,$ENULL         ;TYPE NULL CHARACTER
 021234  013700  000042         $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
 021240  001405                         BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
 021242  000005                         RESET                   ;;CLEAR THE WORLD
 021244  004710                 $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
 021246  000240                         NOP                     ;;SAVE ROOM
 021250  000240                         NOP                     ;;FOR
 021252  000240                         NOP                     ;;ACT11
 021254                         $DOAGN:
 021254  000137                         JMP     @(PC)+          ;;RETURN
 021256  006540                 $RTNAD: .WORD   RESTART
 021260     377      377      000 $ENULL: .BYTE   -1,-1,0         ;;NULL CHARACTER STRING
                                        .EVEN
```

```
                                .SBTTL  APT COMMUNICATIONS ROUTINE

                                ;;*****************************************************************
021264  112737  000001  021530  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
021272  112737  000001  021526  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
021300  000403                  $ATY4:  BR      $ATYC
021302  112737  000001  021530          MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
021310                          $ATYC:
021310  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
021312  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
021314  105737  021526                  TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
021320  001450                          BEQ     5$              ;;IF NOT:  BR
021322  122737  000001  001254          CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
021330  001031                          BNE     3$              ;;IF NOT:  BR
021332  132737  000100  001255          BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
021340  001425                          BEQ     3$              ;;IF NOT:  BR
021342  017600  000004                  MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
021346  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
021354  005737  001234          1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
021360  001375                          BNE     1$              ;;IF NOT:  WAIT
021362  010037  001250                  MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
021366  105720                  2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
021370  001376                          BNE     2$
021372  163700  001250                  SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
021376  006200                          ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
021400  010037  001252                  MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
021404  012737  000004  001234          MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
021412  000413                          BR      5$
021414  017637  000004  021440  3$:     MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
021422  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
021430  013746  177776                  MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
021434  004737  022366                  JSR     PC,$TYPE        ;;CALL TYPE MACRO
021440  000000                  4$:     .WORD   0
021442                          5$:
021442  105737  021530          10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
021446  001416                          BEQ     12$             ;;IF NOT:  BR
021450  005737  001254                  TST     $ENV            ;;RUNNING UNDER APT?
021454  001413                          BEQ     12$             ;;IF NOT:  BR
021456  005737  001234          11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
021462  001375                          BNE     11$             ;;IF NOT:  WAIT
021464  017637  000004  001236          MOV     @4(SP),$FATAL   ;;GET ERROR #
021472  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
021500  005237  001234                  INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
021504  105037  021530          12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
021510  105037  021527                  CLRB    $LFLG           ;;CLEAR LOG FLAG
021514  105037  021526                  CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
021520  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
021522  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
021524  000207                          RTS     PC              ;;RETURN
021526     000          $MFLG:  .BYTE   0               ;;MESSG. FLAG
021527     000          $LFLG:  .BYTE   0               ;;LOG FLAG
021530     000          $FFLG:  .BYTE   0               ;;FATAL FLAG
                                .EVEN
            000200      APTSIZE = 200
            000001      APTENV  = 001
            000100      APTSPOOL= 100
```

```
                000040                  APTCSUP = 040

                                        .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 TYPERR 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
                                        ;*SW09=1         LOOP ON ERROR
                                        ;*CALL
                                        ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER

021532                                  $ERROR:
021532  104407                                  CKSWR                           ;;TEST FOR CHANGE IN SOFT-SWR
021534  032777  000400  157412                  BIT     #SW08,@SWR              ;SEND ERROR MESSAGE TO TTY?
021542  001411                                  BEQ     7$                      ;YES--BRANCH
021544  005737  001324                          TST     LPTAVL                  ;IS THERE A LINE PRINTER AVAILABLE?
021550  001406                                  BEQ     7$                      ;NO--BRANCH
021552  013737  001522  001164                  MOV     LPS,$TPS                ;YES--SETUP STATUS
021560  013737  001524  001166                  MOV     LPB,$TPB                ;AND BUFFER REG.'S FOR LINE PRINTER
021566  105237  001117          7$:             INCB    $ERFLG                  ;;SET THE ERROR FLAG
021572  001775                                  BEQ     7$                      ;;DON'T LET THE FLAG GO TO ZERO
021574  013777  001116  157354                  MOV     $TSTNM,@DISPLAY         ;;DISPLAY TEST NUMBER AND ERROR FLAG
021602  032777  002000  157344                  BIT     #BIT10,@SWR             ;;BELL ON ERROR?
021610  001402                                  BEQ     1$                      ;;NO - SKIP
021612  104401  001224                          TYPE    ,$BELL                  ;;RING BELL
021616  005237  001126          1$:             INC     $ERTTL                  ;;COUNT THE NUMBER OF ERRORS
021622  011637  001132                          MOV     (SP),$ERRPC             ;;GET ADDRESS OF ERROR INSTRUCTION
021626  162737  000002  001132                  SUB     #2,$ERRPC
021634  117737  157272  001130                  MOVB    @$ERRPC,$ITEMB          ;;STRIP AND SAVE THE ERROR ITEM CODE
021642  032777  020000  157304                  BIT     #BIT13,@SWR             ;;SKIP TYPEOUT IF SET
021650  001004                                  BNE     20$                     ;;SKIP TYPEOUTS
021652  004737  022000                          JSR     PC,TYPERR               ;;GO TO USER ERROR ROUTINE
021656  104401  001231                          TYPE    ,$CRLF
021662                          20$:
021662  122737  000001  001254                  CMPB    #APTENV,$ENV            ;;RUNNING IN APT MODE
021670  001007                                  BNE     2$                      ;;NO,SKIP APT ERROR REPORT
021672  113737  001130  021704                  MOVB    $ITEMB,21$              ;;SET ITEM NUMBER AS ERROR NUMBER
021700  004737  021302                          JSR     PC,$ATY4                ;;REPORT FATAL ERROR TO APT
021704     000          21$:            .BYTE   0
021705     000                          .BYTE   0
021706  000777          22$:            BR      22$                     ;;APT ERROR LOOP
021710  005777  157240          2$:     TST     @SWR                    ;;HALT ON ERROR
021714  100002                          BPL     3$                      ;;SKIP IF CONTINUE
021716  000000                          HALT                            ;;HALT ON ERROR!
021720  104407                          CKSWR                           ;;TEST FOR CHANGE IN SOFT-SWR
021722  032777  001000  157224  3$:     BIT     #BIT09,@SWR             ;;LOOP ON ERROR SWITCH SET?
021730  001402                          BEQ     4$                      ;;BR IF NO
021732  013716  001124                  MOV     $LPERR,(SP)             ;;FUDGE RETURN FOR LOOPING
021736  005737  001222          4$:     TST     $ESCAPE                 ;;CHECK FOR AN ESCAPE ADDRESS
021742  001402                          BEQ     5$                      ;;BR IF NONE
021744  013716  001222                  MOV     $ESCAPE,(SP)            ;;FUDGE RETURN ADDRESS FOR ESCAPE
021750                          5$:
```

```
021750  022737  021244  000042          CMP     #$ENDAD,@#42    ;;ACT-11 AUTO-ACCEPT?
021756  001001                          BNE     6$              ;;BRANCH IF NO
021760  000000                          HALT                    ;;YES
021762                          6$:
021762  013737  001516  001164          MOV     TPS,$TPS        ;SET STATUS AND BUFFER REG.'S
021770  013737  001520  001166          MOV     TPB,$TPB        ;FOR TTY
021776  000002                          RTI                     ;RETURN FROM ERROR CALL
                                .SBTTL  TYPERR - TYPE ERROR ROUTINE
                                ;THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE
                                ;WHICH ERROR IS TO BE REPORTED, IT THEN OBTAINS, FROM THE "ERROR
                                ;TABLE" ($ERRTB), AND REPORTS THE APPROPIATE INFORMATION
                                ;CONCERNING THE ERROR.
                                ;CALL
                                ;       JSR     PC,TYPERR
                                ;       RETURN
022000  113737  001116  001212  TYPERR: MOVB    $TSTNM,$TMP0    ;SAVE THE TEST NUMBER
022006  104412                          SAVREG                  ;SAVE R0 - R5
022010  162700  000004                  SUB     #4,R0           ;FORM TEST PC
022014  010037  001176                  MOV     R0,$REG0        ;COPY R0-R5 IN $REG0-$REG5
022020  010137  001200                  MOV     R1,$REG1
022024  010237  001202                  MOV     R2,$REG2
022030  010337  001204                  MOV     R3,$REG3
022034  010437  001206                  MOV     R4,$REG4
022040  010537  001210                  MOV     R5,$REG5
022044  113700  001130                  MOVB    $ITEMB,R0       ;PICKUP ERROR ITEM NUMBER
022050  010001                          MOV     R0,R1           ;AND COPY IT INTO R1
022052  005300                          DEC     R0              ;FORM INDEX FOR ERROR TABLE
022054  106300                          ASLB    R0
022056  106300                          ASLB    R0
022060  106300                          ASLB    R0
022062  103002                          BCC     1$              ;IS ERROR > 37?
022064  062700  000240                  ADD     #ITEM41-$ERRTB,R0       ;YES--FORM OFFSET
022070  062700  002002          1$:     ADD     #$ERRTB,R0      ;FORM ADDRESS
022074  012037  022110                  MOV     (R0)+,2$        ;GET ERROR MESSAGE (EM) POINTER
022100  001447                          BEQ     7$              ;BRANCH IF THERE ISN'T ONE
022102  104401  001231                  TYPE    ,$CRLF          ;"CARRIAGE RETURN - LINE FEED
022106  104401                          TYPE
022110  000000                  2$:     .WORD   0               ;"EM" POINTER GOES HERE
022112  162701  000041                  SUB     #41,R1          ;SPECIAL ERROR ITEM NUMBER?
022116  100440                          BMI     7$              ;NO--BRANCH
022120  013701  001354                  MOV     $VSTAT,R1       ;GET STATUS/ERROR INDICATOR
022124  106301                          ASLB    R1              ;STRIP "DONE" BIT (BIT07)
022126  006301                          ASL     R1              ;STRIP "ERROR" BIT (BIT15)
022130  012702  001750                  MOV     #STATBL,R2      ;1ST ADDRESS ON STATUS MESSAGE POINTERS
022134  005003                          CLR     R3              ;CARRIAGE RETURN-LINE FEED SWITCH
022136  104401  022144                  TYPE    ,65$            ;;TYPE ASCIZ STRING
022142  000402                          BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  / (/
022150                          64$:
022150  012237  022172          3$:     MOV     (R2)+,5$        ;MESSAGE POINTER
022154  006301                          ASL     R1              ;TYPE THIS MESSAGE?
022156  103013                          BCC     6$              ;NO--BRANCH
022160  005103                          COM     R3              ;YES--TYPE A "CR" & "LF"?
022162  001002                          BNE     4$              ;NO--BRANCH
```

```
                                        TYPE    ,$CRLF          ;YES
                                4$:     TYPE
                                5$:     .WORD   0               ;MESSAGE POINTER GOES HERE
                                        TST     R1              ;MORE TO TYPE?
                                        BEQ     6$              ;NO--BRANCH
                                        TYPE    ,BLNKS2         ;TYPE 2 SPACES
                                        BR      3$              ;LOOP
                                6$:     BNE     3$              ;BRANCH IF NOT FINISHED
                                        TYPE    ,67$            ;;TYPE ASCIZ STRING
                                        BR      66$             ;;GET OVER THE ASCIZ
                                ;;67$:  .ASCIZ  /)/
                                66$:
                                7$:     MOV     (R0)+,8$        ;PICK UP DATA HEADER (DH) POINTER
                                        BEQ     9$              ;BRANCH IF NONE
                                        TYPE    ,$CRLF          ;CARRIAGE RETURN-LINE FEED
                                        TYPE
                                8$:     .WORD   0               ;"DH" POINTER GOES HERE
                                9$:     MOV     (R0)+,R1        ;PICKUP DATA TABLE (DT) POINTER
                                        BEQ     20$             ;BRANCH IF NONE
                                        CLR     R5              ;SET INDENT SWITCH
                                        MOV     (R0)+,R0        ;DATA FORMAT (DF) POINTER
                                        MOV     (R0)+,R2        ;NUMBER OF DH'S TO TYPE
                                        BEQ     17$             ;BRANCH IF DH NUMBER IS 0
                                        COM     R5              ;NO INDENT
                                        TYPE    ,$CRLF          ;CARRIAGE RETURN-LINE FEED
                                10$:    MOVB    (R0)+,R3        ;NUMBER OF DATA WORDS TO TYPE
                                        MOVB    (R0)+,R4        ;AND HOW TO TYPE THEM
                                11$:    ROR     R4              ;OCTAL OR DECIMAL?
                                        BCS     12$             ;DECIMAL--BRANCH
                                        MOV     @(R1)+,-(SP)    ;;SAVE @(R1)+ FOR TYPEOUT
                                        TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
                                        BR      13$
                                12$:
                                        MOV     @(R1)+,-(SP)    ;;SAVE @(R1)+ FOR TYPEOUT
                                        TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
                                13$:    DEC     R3              ;MORE NUMBERS TO TYPE?
                                        BEQ     14$             ;NO--BRANCH
                                        TYPE    ,BLNKS2         ;TYPE 2 SPACES
                                        BR      11$             ;LOOP
                                14$:    DEC     R2              ;MORE DH'S?
                                        BLE     20$             ;NO--BRANCH
                                        TYPE    ,$CRLF          ;YES--START A NEW LINE
                                        COM     R5              ;INDENT?
                                        BNE     15$             ;NO--BRANCH
                                        TYPE    ,BLNKS2         ;TYPE 2 SPACES
                                15$:    MOV     (R0)+,16$       ;GET NEXT DH
                                        TYPE                    ;AND TYPE IT
                                16$:    .WORD   0               ;DH POINTER GOES HERE
                                        TYPE    ,$CRLF          ;CARRIAGE RETURN-LINE FEED
                                        TST     R5              ;INDENT?
                                        BNE     10$             ;NO--BRANCH
                                17$:    TYPE    ,BLNKS2         ;TYPE 2 SPACES
                                        BR      10$             ;LOOP
                                20$:    RESREG                  ;RESTORE R0 - R5
                                        RTS     PC              ;RETURN


                                .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
                        ;*

022366  105737  001173          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
022372  100002                  BPL     1$                      ;;BR IF YES
022374  000000                  HALT                            ;;HALT HERE IF NO TERMINAL
022376  000430                  BR      3$                      ;;LEAVE
022400  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
022402  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
022406  122737  000001  001254          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
022414  001011                  BNE     62$                     ;;NO,GO CHECK FOR APT CONSOLE
022416  132737  000100  001255          BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
022424  001405                  BEQ     62$                     ;;NO,GO CHECK FOR CONSOLE
022426  010037  022436                  MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
022432  004737  021272                  JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
022436  000000                  61$:    .WORD   0               ;;MESSAGE ADDRESS
022440  132737  000040  001255  62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
022446  001003                  BNE     60$                     ;;YES,SKIP TYPE OUT
022450  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
022452  001005                  BNE     4$                      ;;BR IF IT ISN'T THE TERMINATOR
022454  005726                  TST     (SP)+                   ;;IF TERMINATOR POP IT OFF THE STACK
022456  012600                  60$:    MOV     (SP)+,R0        ;;RESTORE R0
022460  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
022464  000002                  RTI                             ;;RETURN
022466  122716  000011          4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
022472  001430                  BEQ     8$
022474  122716  000200                  CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
022500  001006                  BNE     5$
022502  005726                  TST     (SP)+                   ;;POP  <CR><LF> EQUIV
022504  104401                  TYPE                            ;;TYPE A CR AND LF
022506  001231                  $CRLF
022510  105037  022716                  CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
022514  000755                  BR      2$                      ;;GET NEXT CHARACTER
022516  004737  022600          5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
022522  123726  001172          6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
022526  001350                  BNE     2$                      ;;IF NO GO GET NEXT CHAR.
022530  013746  001170                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
                                                                ;;AND THE NULL CHAR.
022534  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
022540  002770                  BLT     6$                      ;;BR IF NO--GO POP THE NULL OFF OF STACK
022542  004737  022600                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
022546  105337  022716                  DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
022552  000770                  BR      7$                      ;;LOOP
```

```
                                ;HORIZONTAL TAB PROCESSOR

022554  112716  000040          8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
022560  004737  022600          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
022564  132737  000007  022716          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
022572  001372                          BNE     9$              ;;TAB STOP
022574  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
022576  000724                          BR      2$              ;;GET NEXT CHARACTER
022600                          $TYPEC:
022600  105777  156354                  TSTB    @$TKS           ;;CHAR IN KYBD BUFFER?
022604  100022                          BPL     10$             ;;BR IF NOT
022606  017746  156350                  MOV     @$TKB,-(SP)     ;;GET CHAR
022612  042716  177600                  BIC     #177600,(SP)    ;;STRIP EXTRANEOUS BITS
022616  122716  000023                  CMPB    #$XOFF,(SP)     ;;WAS CHAR XOFF
022622  001012                          BNE     102$            ;;BR IF NOT
022624                          101$:
022624  105777  156330                  TSTB    @$TKS           ;;WAIT FOR CHAR
022630  100375                          BPL     101$
022632  117716  156324                  MOVB    @$TKB,(SP)      ;;GET CHAR
022636  042716  177600                  BIC     #177600,(SP)    ;;STRIP IT
022642  122716  000021                  CMPB    #$XON,(SP)      ;;WAS IT XON?
022646  001366                          BNE     101$            ;;BR IF NOT
022650                          102$:
022650  005726                          TST     (SP)+           ;;FIX STACK
022652                          10$:
022652  105777  156306                  TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
022656  100375                          BPL     10$
022660  116677  000002  156300          MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
022666  122766  000015  000002          CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
022674  001003                          BNE     1$              ;;BRANCH IF NO
022676  105037  022716                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
022702  000406                          BR      $TYPEX          ;;EXIT
022704  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
022712  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
022714  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
022716  000000                  $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
022720  000207                  $TYPEX: RTS     PC

104
105                             .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

022722  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
022726  116637  000001  023145          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
022734  112637  023147                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
022740  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
022744  000406                          BR      $TYPON
022746  112737  000001  023145  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
022754  112737  000006  023147          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
022762  112737  000005  023144  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
022770  010346                          MOV     R3,-(SP)        ;;SAVE R3
022772  010446                          MOV     R4,-(SP)        ;;SAVE R4
022774  010546                          MOV     R5,-(SP)        ;;SAVE R5
022776  113704  023147                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
023002  005404                          NEG     R4
023004  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
023010  110437  023146                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
023014  113704  023145                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
023020  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
023024  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
023026  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
023030  000404                          BR      3$              ;;GO DO MSB
023032  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
023034  006105                          ROL     R5
023036  006105                          ROL     R5
023040  010503                          MOV     R5,R3
023042  006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
023044  105337  023146                  DECB    $OMODE          ;;TYPE THIS DIGIT?
023050  100016                          BPL     7$              ;;BR IF NO
023052  042703  177770                  BIC     #177770,R3      ;;GET RID OF JUNK
023056  001002                          BNE     4$              ;;TEST FOR 0
023060  005704                          TST     R4              ;;SUPPRESS THIS 0?
023062  001403                          BEQ     5$              ;;BR IF YES
023064  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
023066  052703  000060                  BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
023072  052703  000040          5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
023076  110337  023142                  MOVB    R3,8$           ;;SAVE FOR TYPING
023102  104401  023142                  TYPE    ,8$             ;;GO TYPE THIS DIGIT
023106  105337  023144          7$:     DECB    $OCNT           ;;COUNT BY 1
023112  003347                          BGT     2$              ;;BR IF MORE TO DO
023114  002402                          BLT     6$              ;;BR IF DONE
023116  005204                          INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
023120  000744                          BR      2$              ;;GO DO THE LAST DIGIT
023122  012605                  6$:     MOV     (SP)+,R5        ;;RESTORE R5
023124  012604                          MOV     (SP)+,R4        ;;RESTORE R4
023126  012603                          MOV     (SP)+,R3        ;;RESTORE R3
023130  016666  000002  000004          MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
023136  012616                          MOV     (SP)+,(SP)
023140  000002                          RTI                     ;;RETURN
023142     000                  8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
023143     000                          .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
023144     000          $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
```

```
023145  000                 $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
023146  000000              $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
106
107                         .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

023150                      $TYPDS:
023150  010046                      MOV     R0,-(SP)        ;;PUSH R0 ON STACK
023152  010146                      MOV     R1,-(SP)        ;;PUSH R1 ON STACK
023154  010246                      MOV     R2,-(SP)        ;;PUSH R2 ON STACK
023156  010346                      MOV     R3,-(SP)        ;;PUSH R3 ON STACK
023160  010546                      MOV     R5,-(SP)        ;;PUSH R5 ON STACK
023162  012746  020200              MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
023166  016605  000020              MOV     20(SP),R5       ;;GET THE INPUT NUMBER
023172  100004                      BPL     1$              ;;BR IF INPUT IS POS.
023174  005405                      NEG     R5              ;;MAKE THE BINARY NUMBER POS.
023176  112766  000055  000001      MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
023204  005000              1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
023206  012703  023364              MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
023212  112723  000040              MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
023216  005002              2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
023220  016001  023354              MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
023224  160105              3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
023226  002402                      BLT     4$              ;;BR IF DONE
023230  005202                      INC     R2              ;;INCREASE THE BCD DIGIT BY 1
023232  000774                      BR      3$
023234  060105              4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
023236  005702                      TST     R2              ;;CHECK IF BCD DIGIT=0
023240  001002                      BNE     5$              ;;FALL THROUGH IF 0
023242  105716                      TSTB    (SP)            ;;STILL DOING LEADING 0'S?
023244  100407                      BMI     7$              ;;BR IF YES
023246  106316              5$:     ASLB    (SP)            ;;MSD?
023250  103003                      BCC     6$              ;;BR IF NO
023252  116663  000001  177777      MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
023260  052702  000060      6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
023264  052702  000040      7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
023270  110223                      MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
023272  005720                      TST     (R0)+           ;;JUST INCREMENTING
023274  020027  000010              CMP     R0,#10          ;;CHECK THE TABLE INDEX
023300  002746                      BLT     2$              ;;GO DO THE NEXT DIGIT
023302  003002                      BGT     8$              ;;GO TO EXIT
023304  010502                      MOV     R5,R2           ;;GET THE LSD
023306  000764                      BR      6$              ;;GO CHANGE TO ASCII
023310  105726              8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
023312  100003                      BPL     9$              ;;BR IF NO
023314  116063  177777  177776      MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
023322  105013              9$:     CLRB    (R3)            ;;SET THE TERMINATOR
023324  012605                      MOV     (SP)+,R5        ;;POP STACK INTO R5
```

```
023326  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
023330  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
023332  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
023334  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
023336  104401  023364                  TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
023342  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
023350  012616                          MOV     (SP)+,(SP)
023352  000002                          RTI                     ;;RETURN TO USER
023354  023420                  $DTBL:  10000.
023356  001750                          1000.
023360  000144                          100.
023362  000012                          10.
023364                          $DBLK:  .BLKW   4
    108
    109                                 .SBTTL  TTY INPUT ROUTINE

                                        ;;**********************************************************************
                                        .ENABL  LSB
023374  000000                  $TKCNT: .WORD   0               ;;NUMBER OF ITEMS IN QUEUE
023376  000000                  $TKQIN: .WORD   0               ;;INPUT POINTER
023400  000000                  $TKQOUT: .WORD  0               ;;OUTPUT POINTER
023402                          $TKQSRT: .BLKB  2               ;;TTY KEYBOARD QUEUE
        023404                  $TKQEND=.

                                ;*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
                                ;
023404  005037  023374          $TKINT: CLR     $TKCNT          ;;CLEAR COUNT OF ITEMS IN QUEUE
023410  012737  023402  023376          MOV     #$TKQSRT,$TKQIN ;;MOVE THE STARTING ADDRESS OF THE
023416  013737  023376  023400          MOV     $TKQIN,$TKQOUT  ;;QUEUE INTO THE INPUT & OUTPUT POINTERS.
023424  012737  023454  000060          MOV     #$TKSRV,@#TKVEC ;;INITIALIZE THE KEYBOARD VECTOR
023432  012737  000200  000062          MOV     #200,@#TKVEC+2  ;;"BR" LEVEL 4
023440  005777  155516                  TST     @$TKB           ;;CLEAR DONE FLAG
023444  012777  000100  155506          MOV     #100,@$TKS      ;;ENABLE TTY KEYBOARD INTERRUPT
023452  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 (START2)
                                ;
023454  117746  155502          $TKSRV: MOVB    @$TKB,-(SP)     ;;PICKUP THE CHARACTER
023460  042716  177600                  BIC     #^C177,(SP)     ;;STRIP THE JUNK
023464  021627  000003                  CMP     (SP),#3         ;;IS IT A CONTROL C?
023470  001007                          BNE     1$              ;;BRANCH IF NO
023472  104401  024604                  TYPE    ,$CNTLC         ;;TYPE A CONTROL-C (^C)
023476  004737  023404                  JSR     PC,$TKINT       ;;INIT THE KEYBOARD
023502  005726                          TST     (SP)+           ;;CLEAN UP STACK
023504  000137  004612                  JMP     START2          ;;CONTROL C RESTART
023510  021627  000007          1$:     CMP     (SP),#7         ;;IS IT A CONTROL G?
```

```
023514  001004                          BNE     2$              ;;BRANCH IF NO
023516  022737  000176  001154          CMP     #SWREG,SWR      ;;IS SOFT-SWR SELECTED?
023524  001500                          BEQ     6$              ;;GO TO SWR CHANGE

023526                          2$:
023526  022737  000002  023374          CMP     #2,$TKCNT       ;;IS THE QUEUE FULL?
023534  001004                          BNE     3$              ;;BRANCH IF NO
023536  104401  001224                  TYPE    ,$BELL          ;;RING THE TTY BELL
023542  005726                          TST     (SP)+           ;;CLEAN CHARACTER OFF OF STACK
023544  000451                          BR      5$              ;;EXIT
023546  021627  000023          3$:     CMP     (SP),#23        ;;IS IT A CONTROL-S?
023552  001021                          BNE     32$             ;;BRANCH IF NO
023554  005077  155400                  CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
023560  005726                          TST     (SP)+           ;;CLEAN CHAR OFF STACK
023562  105777  155372          31$:    TSTB    @$TKS           ;;WAIT FOR A CHAR
023566  100375                          BPL     31$             ;;LOOP UNTIL ITS THERE
023570  117746  155366                  MOVB    @$TKB,-(SP)     ;;GET THE CHARACTER
023574  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
023600  022627  000021                  CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
023604  001366                          BNE     31$             ;;BRANCH IF NO
023606  012777  000100  155344          MOV     #100,@$TKS      ;;REENABLE TTY KEYBOARD INTERRUPTS
023614  000002                          RTI                     ;;RETURN
023616  005237  023374          32$:    INC     $TKCNT          ;;COUNT THIS CHARACTER
023622  021627  000140                  CMP     (SP),#140       ;;IS IT UPPER CASE?
023626  002405                          BLT     4$              ;;BRANCH IF YES
023630  021627  000175                  CMP     (SP),#175       ;;IS IT A SPECIAL CHAR?
023634  003002                          BGT     4$              ;;BRANCH IF YES
023636  042716  000040                  BIC     #40,(SP)        ;;MAKE IT UPPER CASE
023642  112677  177530          4$:     MOVB    (SP)+,@$TKQIN   ;;AND PUT IT IN QUEUE
023646  005237  023376                  INC     $TKQIN          ;;UPDATE THE POINTER
023652  023727  023376  023404          CMP     $TKQIN,#$TKQEND ;;GO OFF THE END?
023660  001003                          BNE     5$              ;;BRANCH IF NO
023662  012737  023402  023376          MOV     #$TKQSRT,$TKQIN ;;RESET THE POINTER
023670  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.
023672  022737  000176  001154  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED
023700  001124                          BNE     15$             ;;EXIT IF NOT
023702  105777  155252                  TSTB    @$TKS           ;;IS A CHAR WAITING?
023706  100121                          BPL     15$             ;;IF NOT, EXIT
023710  117746  155246                  MOVB    @$TKB,-(SP)     ;;YES
023714  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT  7-BIT ASCII
023720  021627  000007                  CMP     (SP),#7         ;;IS IT A CONTROL-G?
023724  001300                          BNE     2$              ;;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.
023726  123727  001150  000001  6$:     CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
023734  001674                          BEQ     2$              ;;BRANCH IF YES
023736  005726                          TST     (SP)+           ;;CLEAR CONTROL-G OFF STACK
```

```
023740  004737  023404                  JSR     PC,$TKINT       ;;FLUSH THE TTY INPUT QUEUE
023744  005077  155210                  CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
023750  112737  000001  001151          MOVB    #1,$INTAG       ;;SET INTERRUPT MODE INDICATOR

023756  104401  024616                  TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
023762  104401  024623          $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
023766  013746  000176                  MOV     $WREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
023772  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
023774  104401  024634                  TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
024000  005046                  19$:    CLR     -(SP)           ;;CLEAR COUNTER
024002  005046                          CLR     -(SP)           ;;THE NEW SWR
024004  105777  155150          7$:     TSTB    @$TKS           ;;CHAR THERE?
024010  100375                          BPL     7$              ;;IF NOT TRY AGAIN

024012  117746  155144                  MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
024016  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII

024022  021627  000003                  CMP     (SP),#3         ;;IS IT A CONTROL-C?
024026  001015                          BNE     9$              ;;BRANCH IF NOT
024030  104401  024604                  TYPE    ,$CNTLC         ;;YES, ECHO CONTROL-C (^C)
024034  062706  000006                  ADD     #6,SP           ;;CLEAN UP STACK
024040  123727  001151  000001          CMPB    $INTAG,#1       ;;REENABLE TTY KEYBOARD INTERRUPTS?
024046  001003                          BNE     8$              ;;BRANCH IF NO
024050  012777  000100  155102          MOV     #100,@$TKS      ;;ALLOW TTY KEYBOARD INTERRUPTS
024056  000137  004612          8$:     JMP     START2          ;;CONTROL-C RESTART

024062  021627  000025          9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
024066  001005                          BNE     10$             ;;BRANCH IF NOT
024070  104401  024611                  TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (^U)
024074  062706  000006          20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
024100  000737                          BR      19$             ;;LET'S TRY IT AGAIN


024102  021627  000015          10$:    CMP     (SP),#15        ;;IS IT A <CR>?
024106  001022                          BNE     16$             ;;BRANCH IF NO
024110  005766  000004                  TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
024114  001403                          BEQ     11$             ;;BRANCH IF YES
024116  016677  000002  155030          MOV     2(SP),@SWR      ;;SAVE NEW SWR
024124  062706  000006          11$:    ADD     #6,SP           ;;CLEAR UP STACK
024130  104401  001231          14$:    TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
024134  123727  001151  000001          CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
024142  001003                          BNE     15$             ;;BRANCH IF NOT
024144  012777  000100  155006          MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
024152  000002                  15$:    RTI                     ;;RETURN
024154  004737  022600          16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
024160  021627  000060                  CMP     (SP),#60        ;;CHAR < 0?
024164  002420                          BLT     18$             ;;BRANCH IF YES
024166  021627  000067                  CMP     (SP),#67        ;;CHAR > 7?
024172  003015                          BGT     18$             ;;BRANCH IF YES
024174  042726  000060                  BIC     #60,(SP)+       ;;STRIP-OFF ASCII
024200  005766  000002                  TST     2(SP)           ;;IS THIS THE FIRST CHAR
024204  001403                          BEQ     17$             ;;BRANCH IF YES
024206  006316                          ASL     (SP)            ;;NO, SHIFT PRESENT
024210  006316                          ASL     (SP)            ;;    CHAR OVER TO MAKE
024212  006316                          ASL     (SP)            ;;    ROOM FOR NEW ONE.
024214  005266  000002          17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
```

```
024220  056616  177776                  BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
024224  000667                          BR      7$              ;;GET THE NEXT ONE
024226  104401  001230          18$:    TYPE    ,$QUES          ;;TYPE ?<CR><LF>
024232  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
                                ;

024234  011646                  $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC AND
024236  016666  000004  000002          MOV     4(SP),2(SP)     ;;THE PS
024244  005066  000004                  CLR     4(SP)           ;;GET READY FOR A CHARACTER
024250  005046                          CLR     -(SP)           ;;PUT NEW PS ON STACK
024252  012746  024260                  MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
024256  000002                          RTI                     ;;POP NEW PC AND PS
024260                          64$:
024260  005737  023374          1$:     TST     $TKCNT          ;;WAIT ON A CHARACTER
024264  001775                          BEQ     1$
024266  005337  023374                  DEC     $TKCNT          ;;DECREMENT THE COUNTER
024272  117766  177102  000004          MOVB    @$TKQOUT,4(SP)  ;;GET ONE CHARACTER
024300  005237  023400                  INC     $TKQOUT         ;;UPDATE THE POINTER
024304  023727  023400  023404          CMP     $TKQOUT,#$TKQEND ;;DID IT GO OFF OF THE END?
024312  001003                          BNE     2$              ;;BRANCH IF NO
024314  012737  023402  023400          MOV     #$TKQSRT,$TKQOUT ;;RESET THE POINTER
024322  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

024324  010346                  $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
024326  005046                          CLR     -(SP)           ;;CLEAR THE RUBOUT KEY
024330  012703  024560          1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
024334  022703  024604          2$:     CMP     #$TTYIN+20.,R3  ;;BUFFER FULL?
024340  101456                          BLOS    4$              ;;BR IF YES
024342  104410                          RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
024344  112613                          MOVB    (SP)+,(R3)      ;;GET CHARACTER
024346  122713  000177          10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
024352  001022                          BNE     5$              ;;BR IF NO
024354  005716                          TST     (SP)            ;;IS THIS THE FIRST RUBOUT?
024356  001007                          BNE     6$              ;;BR IF NO
024360  112737  000134  024556          MOVB    #'\,9$          ;;TYPE A BACK SLASH
024366  104401  024556                  TYPE    ,9$
024372  012716  177777                  MOV     #-1,(SP)        ;;SET THE RUBOUT KEY
024376  005303                  6$:     DEC     R3              ;;BACKUP BY ONE
024400  020327  024560                  CMP     R3,#$TTYIN      ;;STACK EMPTY?
024404  103434                          BLO     4$              ;;BR IF YES
024406  111337  024556                  MOVB    (R3),9$         ;;SETUP TO TYPEOUT THE DELETED CHAR.
024412  104401  024556                  TYPE    ,9$             ;;GO TYPE
```

```
024416  000746                          BR      2$              ;;GO READ ANOTHER CHAR.
024420  005716                  5$:     TST     (SP)            ;;RUBOUT KEY SET?
024422  001406                          BEQ     7$              ;;BR IF NO
024424  112737  000134  024556          MOVB    #'\,9$          ;;TYPE A BACK SLASH
024432  104401  024556                  TYPE    ,9$
024436  005016                          CLR     (SP)            ;;CLEAR THE RUBOUT KEY
024440  122713  000025          7$:     CMPB    #25,(R3)        ;;IS CHARACTER A CTRL U?
024444  001003                          BNE     8$              ;;BR IF NO
024446  104401  024611                  TYPE    ,$CNTLU         ;;TYPE A CONTROL "U"
024452  000726                          BR      1$              ;;GO START OVER
024454  122713  000022          8$:     CMPB    #22,(R3)        ;;IS CHARACTER A "^R"?
024460  001011                          BNE     3$              ;;BRANCH IF NO
024462  105013                          CLRB    (R3)            ;;CLEAR THE CHARACTER
024464  104401  001231                  TYPE    ,$CRLF          ;;TYPE A "CR" & "LF"
024470  104401  024560                  TYPE    ,$TTYIN         ;;TYPE THE INPUT STRING
024474  000717                          BR      2$              ;;GO PICKUP ANOTHER CHACTER
024476  104401  001230          4$:     TYPE    ,$QUES          ;;TYPE A '?'
024502  000712                          BR      1$              ;;CLEAR THE BUFFER AND LOOP
024504  111337  024556          3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
024510  104401  024556                  TYPE    ,9$
024514  122723  000015                  CMPB    #15,(R3)+       ;;CHECK FOR RETURN
024520  001305                          BNE     2$              ;;LOOP IF NOT RETURN
024522  105063  177777                  CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
024526  104401  001232                  TYPE    ,$LF            ;;TYPE A LINE FEED
024532  005726                          TST     (SP)+           ;;CLEAN RUBOUT KEY FROM THE STACK
024534  012603                          MOV     (SP)+,R3        ;;RESTORE R3
024536  011646                          MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
024540  016666  000004  000002          MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
024546  012766  024560  000004          MOV     #$TTYIN,4(SP)
024554  000002                          RTI                     ;;RETURN
024556     000                  9$:     .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
024557     000                          .BYTE   0               ;;TERMINATOR
024560                          $TTYIN: .BLKB   20.             ;;RESERVE 20. BYTES FOR TTY INPUT
024604     136     103     015  $CNTLC: .ASCIZ  /^C/<15><12>    ;;CONTROL "C"
024611     136     125     015  $CNTLU: .ASCIZ  /^U/<15><12>    ;;CONTROL "U"
024616     136     107     015  $CNTLG: .ASCIZ  /^G/<15><12>    ;;CONTROL "G"
024623     015     012     123  $MSWR:  .ASCIZ  <15><12>/SWR = /
024634     040     040     116  $MNEW:  .ASCIZ  /  NEW = /
                                .EVEN

110
111                             .SBTTL  SCOPE HANDLER ROUTINE

                                ;;**********************************************************************
                                ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
                                ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
                                ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
                                ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
                                ;*SW14=1         LOOP ON TEST
                                ;*SW11=1         INHIBIT ITERATIONS
                                ;*SW09=1         LOOP ON ERROR
                                ;*CALL
                                ;*      SCOPE           ;;SCOPE=IOT

024646                          $SCOPE:
024646  104407                          CKSWR                   ;;TEST  FOR CHANGE IN SOFT-SWR
024650  032777  040000  154276  1$:     BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
024656  001101                          BNE     $OVER           ;;YES IF SW14=1
```

```
                                        ;#####START OF CODE FOR THE XOR TESTER#####
024660  000416                  $XTSTR: 3R      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
                                                                ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
024662  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
024666  012737  024706  000004          MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
024674  005737  177060                  TST     @#177060        ;;TIME OUT ON XOR?
024700  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
024704  000450                          BR      $SVLAD          ;;GO TO THE NEXT TEST
024706  022626                  5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
024710  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
024714  000413                          BR      7$              ;;LOOP ON THE PRESENT TEST
024716                          6$:;#####END OF CODE FOR THE XOR TESTER#####
024716  105737  001117          2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
024722  001421                          BEQ     3$              ;;BR IF NO
024724  123737  001131  001117          CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
024732  101015                          BHI     3$              ;;BR IF NO
024734  032777  001000  154212          BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
024742  001404                          BEQ     4$              ;;BR IF NO
024744  013737  001124  001122  7$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
024752  000443                          BR      $OVER
024754  105037  001117          4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
024760  005037  001220                  CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
024764  000412                          BR      1$              ;;ESCAPE TO THE NEXT TEST
024766  032777  004000  154160  3$:     BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
024774  001006                          BNE     1$              ;;BR IF YES
024776  005237  001120                  INC     $ICNT           ;;INCREMENT ITERATION COUNT
025002  023737  001220  001120          CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
025010  002024                          BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
025012  012737  000001  001120  1$:     MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
025020  013737  025076  001220          MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
025026  105237  001116          $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
025032  113737  001116  001240          MOVB    $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
025040  011637  001122                  MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
025044  011637  001124                  MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
025050  005037  001222                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
025054  112737  000001  001131          MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
025062  013777  001116  154066  $OVER:  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
025070  013716  001122                  MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
025074  000002                          RTI                     ;;FIXES PS
025076  000001                  $MXCNT: 1                       ;;MAX. NUMBER OF ITERATIONS
112
113                                     .SBTTL  SAVE AND RESTORE R0-R5 ROUTINES

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

```
                                        ;*SAVREG:
025100                                  ;*      MOV     R0,-(SP)        ;;PUSH R0 ON STACK
025100  010046                         ;*      MOV     R1,-(SP)        ;;PUSH R1 ON STACK
025102  010146                         ;*      MOV     R2,-(SP)        ;;PUSH R2 ON STACK
025104  010246                         ;*      MOV     R3,-(SP)        ;;PUSH R3 ON STACK
025106  010346                         ;*      MOV     R4,-(SP)        ;;PUSH R4 ON STACK
025110  010446                         ;*      MOV     R5,-(SP)        ;;PUSH R5 ON STACK
025112  010546                         ;*      MOV     22(SP),-(SP)    ;;SAVE PS OF MAIN FLOW
025114  016646  000022                 ;*      MOV     22(SP),-(SP)    ;;SAVE PC OF MAIN FLOW
025120  016646  000022                 ;*      MOV     22(SP),-(SP)    ;;SAVE PS OF CALL
025124  016646  000022                 ;*      MOV     22(SP),-(SP)    ;;SAVE PC OF CALL
025130  016646  000022                 ;*      RTI
025134  000002                         ;*
                                        ;*RESTORE R0-R5
                                        ;*CALL:
                                        ;*      RESREG
025136                                  ;*RESREG:
025136  012666  000022                 ;*      MOV     (SP)+,22(SP)    ;;RESTORE PC OF CALL
025142  012666  000022                 ;*      MOV     (SP)+,22(SP)    ;;RESTORE PS OF CALL
025146  012666  000022                 ;*      MOV     (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
025152  012666  000022                 ;*      MOV     (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
025156  012605                         ;*      MOV     (SP)+,R5        ;;POP STACK INTO R5
025160  012604                         ;*      MOV     (SP)+,R4        ;;POP STACK INTO R4
025162  012603                         ;*      MOV     (SP)+,R3        ;;POP STACK INTO R3
025164  012602                         ;*      MOV     (SP)+,R2        ;;POP STACK INTO R2
025166  012601                         ;*      MOV     (SP)+,R1        ;;POP STACK INTO R1
025170  012600                         ;*      MOV     (SP)+,R0        ;;POP STACK INTO R0
025172  000002                         ;*      RTI
114
115                                     .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.

025174  010046                         $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
025176  016600  000002                         MOV     2(SP),R0        ;;GET TRAP ADDRESS
025202  005740                                 TST     -(R0)           ;;BACKUP BY 2
025204  111000                                 MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
025206  006300                                 ASL     R0              ;;POSITION FOR INDEXING
025210  016000  025230                         MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
025214  000200                                 RTS     R0              ;;GO TO ROUTINE


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

025216  011646                         $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
025220  016666  000004  000002                 MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
025226  000002                                 RTI                     ;;RESTORE THE PSW

                                        .SBTTL  TRAP TABLE

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

```
                                        ;       ROUTINE
                                        ;       -------
     025230  025216                     $TRPAD: .WORD   $TRAP2
     025232  022366                             $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
     025234  022746                             $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
     025236  022722                             $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
     025240  022762                             $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
     025242  023150                             $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)

     025244  023762                             $GTSWR  ;;CALL=GTSWR    TRAP+6(104406)  GET SOFT-SWR SETTING

     025246  023672                             $CKSWR  ;;CALL=CKSWR    TRAP+7(104407)  TEST FOR CHANGE IN SOFT-SWR
     025250  024234                             $RDCHR  ;;CALL=RDCHR    TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
     025252  024324                             $RDLIN  ;;CALL=RDLIN    TRAP+11(104411) TTY TYPEIN STRING ROUTINE
     025254  025100                             $SAVREG ;;CALL=SAVREG   TRAP+12(104412) SAVE R0-R5 ROUTINE
     025256  025136                             $RESREG ;;CALL=RESREG   TRAP+13(104413) RESTORE R0-R5 ROUTINE
116
117                                     .SBTTL  SINGLE LENGTH BINARY TO DECIMAL ASCIZ ROUTINE

                                        ;;**********************************************************************
                                        ;*THIS ROUTINE WILL CONVERT A 16-BIT UNSIGNED BINARY NUMBER TO AN
                                        ;*UNSIGNED DECIMAL ASCIZ NUMBER.
                                        ;*CALL
                                        ;*      MOV     NUMBER,-(SP)    ;;PUT BINARY NUMBER ON THE STACK
                                        ;*      JSR     PC,@#$SB2D      ;;CALL
                                        ;*      RETURN                  ;;ADDRESS OF THE 1ST ASCIZ CHAR.IS ON THE STACK

     025260  016637  000002  025310     $SB2D:  MOV     2(SP),1$        ;;SAVE BINARY NUMBER
     025266  012746  025310             1$:     MOV     #1$,-(SP)       ;;SET POINTER
     025272  004737  025314             2$:     JSR     PC,@#$DB2D      ;;CALL DOUBLE LENGTH CONVERT
     025276  062716  000005                     ADD     #5,(SP)         ;;ONLY ALLOW FIVE CHARACTERS
     025302  012666  000002                     MOV     (SP)+,2(SP)     ;;PICKUP POINTER
     025306  000207                             RTS     PC              ;;RETURN
     025310  000000  000000             1$:     .WORD   0,0
118
119                                     .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

     025314  104412                     $DB2D:  SAVREG                  ;;SAVE REGISTERS
     025316  016602  000002                     MOV     2(SP),R2        ;;PICKUP THE DATA POINTER
     025322  012700  025474                     MOV     #$DECVL,R0      ;;GET ADDRESS OF "$DECVL" STRING
     025326  010066  000002                     MOV     R0,2(SP)        ;;PUT ADDRESS OF ASCIZ STRING ON STACK
     025332  012201                             MOV     (R2)+,R1        ;;PICKUP THE BINARY NUMBER
     025334  012202                             MOV     (R2)+,R2
     025336  012737  000012  025412             MOV     #10.,4$         ;;SET UP TO DO 10 CONVERSIONS
```

```
025344  012704  025424                  MOV     #$TNPWR,R4      ;;ADDRESS OF TEN POWER
025350  012705  025426                  MOV     #$TNPWR+2,R5
025354  005003                  1$:     CLR     R3              ;;CLEAR PARTIAL
025356  161401                  2$:     SUB     (R4),R1         ;;SUBTRACT TEN POWER
025360  005602                          SBC     R2
025362  161502                          SUB     (R5),R2
025364  002402                          BLT     3$              ;;BR IF TEN POWER TO LARGE
025366  005203                          INC     R3              ;;ADD 1 TO PARTIAL
025370  000772                          BR      2$              ;;LOOP
025372  062401                  3$:     ADD     (R4)+,R1        ;;RESTORE SUBTRACTED VALUE
025374  005502                          ADC     R2
025376  062402                          ADD     (R4)+,R2
025400  022525                          CMP     (R5)+,(R5)+     ;;MOVE TO NEXT TEN POWER
025402  052703  000060                  BIS     #'0,R3          ;;CHANGE PARTIAL TO ASCII
025406  110320                          MOVB    R3,(R0)+        ;;SAVE IT
025410  005327                          DEC     (PC)+           ;;DONE?
025412  000000                  4$:     .WORD   0
025414  001357                          BNE     1$              ;;BR IF NO
025416  105020                          CLRB    (R0)+           ;;TERMINATOR
025420  104413                          RESREG                  ;;RESTORE REGISTERS
025422  000207                          RTS     PC              ;;RETURN
025424  145000                  $TNPWR: 145000                  ;;1.0E09
025426  035632                          35632
025430  160400                          160400                  ;;1.0E08
025432  002765                          2765
025434  113200                          113200                  ;;1.0E07
025436  000230                          230
025440  041100                          041100                  ;;1.0E06
025442  000017                          17
025444  103240                          103240                  ;;1.0E05
025446  000001                          1
025450  023420                          23420                   ;;1.0E04
025452  000000                          0
025454  001750                          1750                    ;;1.0E03
025456  000000                          0
025460  000144                          144                     ;;1.0E02
025462  000000                          0
025464  000012                          12                      ;;1.0E01
025466  000000                          0
025470  000001                          1                       ;;1.0E00
025472  000000                          0
025474                          $DECVL: .BLKB   12.             ;;RESERVE STORAGE FOR ASCIZ STRING
120
121                                     .SBTTL  TYPE NUMERICAL ASCIZ STRING SUPPRESS LEADING ZEROS

                                        ;;**********************************************************************
                                        ;*THIS ROUTINE IS USED TO TYPE AN ASCIZ NUMBER SUPPRESSING THE
                                        ;*LEADING NUMBERS.
                                        ;*CALL
                                        ;*      MOV     #NUMADR,-(SP)   ;;FIRST ADDRESS OF ASCIZ STRING
                                        ;*      JSR     PC,@#$SUPRS

025510  010046                  $SUPRS: MOV     R0,-(SP)        ;;SAVE R0
025512  016600  000004                  MOV     4(SP),R0        ;;PICKUP THE POINTER
025516  105710                  1$:     TSTB    (R0)            ;;TERMINATEOR?
025520  001403                          BEQ     2$              ;;BR IF YES
```

```
025522  122720  000060                  CMPB    #'0,(R0)+       ;;IS THIS AN ASCII "0" ?
025526  001773                          BEQ     1$              ;;BR IF YES
025530  005300                  2$:     DEC     R0              ;;BACKUP BY "1"
025532  010037  025540                  MOV     R0,3$           ;;SAVE FOR TYPING
025536  104401                          TYPE                    ;;GO TYPE
025540  000000                  3$:     .WORD   0               ;;ASCIZ POINTER GOES HERE
025542  012600                          MOV     (SP)+,R0        ;;RESTORE R0
025544  012616                          MOV     (SP)+,(SP)      ;;RESTORE THE STACK
025546  000207                          RTS     PC              ;;RETURN
  122
  123                           .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

025550                          $RAND:
025550  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
025552  010146                          MOV     R1,-(           ;;PUSH R1 ON STACK
025554  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
025556  013700  025650                  MOV     $LONUM,R0       ;;SET R0 WITH LOW
025562  013701  025646                  MOV     $HINUM,R1       ;;SET R1 WITH HIGH
025566  012702  177771                  MOV     #-7,R2          ;;SET SHIFT COUNT
025572  006300                  1$:     ASL     R0              ;;SHIFT R0 LEFT AND
025574  006101                          ROL     R1              ;;ROTATE CARRY INTO R1 AND
025576  005202                          INC     R2              ;;CHECK FOR DONE
025600  001374                          BNE     1$              ;;CONTINUE SHIFT LOOP
025602  063700  025650                  ADD     $LONUM,R0       ;;ADD NUMBER TO MAKE X 129
025606  005501                          ADC     R1              ;;PROPOGATE CARRY
025610  063701  025646                  ADD     $HINUM,R1       ;;ADD NUMBER TO MAKE X 129
025614  062700  001057                  ADD     #1057,R0        ;;ADD LOW CONSTANT
025620  005501                          ADC     R1              ;;PROPOGATE CARRY
025622  062701  047401                  ADD     #47401,R1       ;;ADD HIGH CONSTANT
025626  010037  025650                  MOV     R0,$LONUM       ;;SAVE R0
025632  010137  025646                  MOV     R1,$HINUM       ;;SAVE R1
025636  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
025640  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
025642  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
025644  000207                          RTS     PC              ;;RETURN
025646  176543                  $HINUM: .WORD   176543
025650  123456                  $LONUM: .WORD   123456
  124
  125                           .SBTTL  INTEGER DIVIDE ROUTINE

                                ;;**********************************************************************
                                ;*THIS ROUTINE WILL DIVIDE A 32-BIT TWO'S COMPLEMENT INTEGER
                                ;*DIVIDEND BY A 16-BIT TWO'S COMPLEMENT INTEGER DIVISOR GIVING
                                ;*A 16-BIT TWO'S COMPLEMENT INTEGER QUOTIENT AND A 16-BIT REMAINDER.
                                ;*DIVISION WILL BE PERFORMED SO THAT THE REMAINDER IS OF THE
                                ;*SAME SIGN AS THE DIVIDEND.
                                ;*CALL:
                                ;*      MOV     LOW DIVIDEND,-(SP)      ;;THE HIGH DIVIDEND MUST BE < 1/2
```

```
                                ;*      MOV     HIGH DIVIDEND,-(SP)     ;;AS LARGE AS THE DIVISOR
                                ;*      MOV     DIVISOR,-(SP)
                                ;*      JSR     PC,$DIV
                                ;*      RETURN                          ;;QUOTIENT & REMAINDER ARE ON THE STACK
                                ;*      "V"=0   IMPLIES NO ERROR
                                ;*      "V"=1   IMPLIES ERROR OCCURRED
                                ;*              "C"=0   DIVIDE OVERFLOW OCCURRED
                                ;*              "C"=1   ATTEMPTED TO DIVIDE BY ZERO
                                ;*
                                ;*
                                ;*      STACK   NO ERROR        OVERFLOW        DIVIDE BY ZERO
                                ;*      -----   --------        --------        --------------
                                ;*      TOP     REMAINDER       ALL ZEROS       ALL ONES
                                ;*      +2      QUOTIENT        ALL ZEROS       ALL ONES

025652                          $DIV:
025652  104400                          TRAP                    ;;PUSH OLD PSW AND PC ON STACK
025654  042716  000017                  BIC     #17,(SP)        ;;STRIP AWAY CONDITION CODES
025660  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
025662  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
025664  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
025666  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
025670  005046                          CLR     -(SP)           ;;SAVE A PLACE FOR SIGNS
025672  012746  000021                  MOV     #17.,-(SP)      ;;SETUP THE ITERATION COUNTER
025676  016601  000024                  MOV     24(SP),R1       ;;PICKUP THE DIVIDEND
025702  016600  000022                  MOV     22(SP),R0
025706  100005                          BPL     1$              ;;CHECK THE SIGN
025710  105366  000003                  DECB    3(SP)           ;;KEEP TRACK OF THE SIGN
025714  005400                          NEG     R0              ;;AND NEGATE THE ORIGINAL
025716  005401                          NEG     R1              ;;NUMBER
025720  005600                          SBC     R0
025722  016602  000020          1$:     MOV     20(SP),R2       ;;PICKUP THE DIVISOR
025726  002407                          BLT     2$              ;;CHECK THE SIGN
025730  003011                          BGT     3$              ;;DIVISOR OF 0 IS A NO-NO
025732  052766  000003  000014          BIS     #3,14(SP)       ;;SET "V" & "C"
025740  012700  177777                  MOV     #-1,R0          ;;SET REMAINDER TO ALL ONES
025744  000424                          BR      7$              ;;EXIT
025746  005266  000002          2$:     INC     2(SP)           ;;KEEP TRACK OF DIVISORS SIGN
025752  000401                          BR      4$
025754  005402                  3$:     NEG     R2              ;;NEGATE THE ORIGINAL NUMBER
025756  000241                  4$:     CLC                     ;;CLEAR "C"
025760  000405                          BR      6$              ;;START FORMING QUOTIENT
025762  006100                  5$:     ROL     R0              ;;POSITION MSB'S
025764  010003                          MOV     R0,R3           ;;COPY
025766  060203                          ADD     R2,R3           ;;COMPARE DIVIDEND & DIVISOR
025770  103001                          BCC     6$              ;;BR IF DIVIDEND > DIVISOR
025772  010300                          MOV     R3,R0           ;;REMAINDER AFTER THIS LOOP
025774  006101                  6$:     ROL     R1              ;;QUOTIENT BIT ENTERS HERE
025776  005316                          DEC     (SP)            ;;DONE?
026000  001370                          BNE     5$              ;;BR IF NO
026002  005701                          TST     R1              ;;OVERFLOW?
026004  100005                          BPL     8$              ;;BR IF NO
026006  052766  000002  000014          BIS     #2,14(SP)       ;;SET "V" IN RETURN STATUS WORD
026014  005000                          CLR     R0              ;;SET REMAINDER TO ALL ZEROS
026016  010001                  7$:     MOV     R0,R1           ;;COPY REMAINDER INTO QUOTIENT
026020  005726                  8$:     TST     (SP)+           ;;CLEAR COUNTER FROM STACK
026022  005716                          TST     (SP)            ;;REMAINDER SIGN CORRECTION NEEDED?
```

```
026024  002004                          BGE     9$              ;;BR IF NO
026026  005400                          NEG     R0              ;;NEGATE REMAINDER
026030  105066  000001                  CLRB    1(SP)           ;;CLEAR SIGN
026034  005316                          DEC     (SP)            ;;BUT DON'T FORGET QUOTIENT
026036  005726                  9$:     TST     (SP)+           ;;QUOTIENT SIGN CORRECTION NEEDED?
026040  001401                          BEQ     10$             ;;BR IF NO
026042  005401                          NEG     R1              ;;NEGATE QUOTIENT
026044  010166  000020          10$:    MOV     R1,20(SP)       ;;RETURN QUOTIENT AND
026050  010066  000016                  MOV     R0,16(SP)       ;;REMAINDER TO USER
026054  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
026056  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
026060  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
026062  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
026064  012666  000002                  MOV     (SP)+,2(SP)     ;;SETUP TO RETURN CONDITION CODES
026070  000002                          RTI                     ;;RETURN
126
127                                     ;THIS SUBROUTINE CLEARS THE MASSBUS CONTROLLER, MASSBUS ADAPTER,
128                                     ;AND DRIVERS, THEN SELECTS THE DRIVE.
129                                     ;CALL:
130                                     ;       JSR     PC,CNTCLR       ;CALL TO ROUTINE
131                                     ;
132
133 026072  013704  037066      CNTCLR: MOV     RMADR,R4        ;GET RMCS1 BASE ADDRESS
134 026076  012764  000040  000010      MOV     #CLR,RMCS2(R4)  ;ISSUE MASSBUS CLEAR AND
135 026104  013764  001350  000010      MOV     CHKDRV,RMCS2(R4)        ;SELECT THE DRIVE
136 026112  000207                      RTS     PC              ;RETURN
137
138                                     ;SET "LPTAVL" TO THE PROPER STATE.
139                                     ; LPTAVL = 0 IF NO LINE PRINTER AVAILABLE
140                                     ; LPTAVL = 1 IF LINE PRINTER IS AVAILABLE
141                                     ;CALL
142                                     ;       JSR     PC,LP.AVL
143                                     ;       RETURN
144
145 026114  005037  001324      LP.AVL: CLR     LPTAVL          ;START WITH NO PRINTER AVAIABLE
146 026120  012737  026144  000004      MOV     #1$,ERRVEC      ;SETUP THE TIMEOUT VECTOR
147 026126  005037  000006              CLR     ERRVEC+2
148 026132  005777  153364              TST     @LPS            ;IS THERE A LINE PRINTER?
149 026136  005237  001324              INC     LPTAVL          ;YES--SET AVAILABLE SWITCH
150 026142  000401                      BR      2$
151 026144  022626              1$:     CMP     (SP)+,(SP)+     ;NO--POP STACK
152 026146  012737  000006  000004  2$: MOV     #ERRVEC+2,ERRVEC        ;RESTORE TIMEOUT VECTOR
153 026154  000207                      RTS     PC              ;RETURN
154
155                                     ;THIS ROUTINE WILL DETERMINE IF THERE IS A CLOCK ON THE SYSTEM
156                                     ;AND IF THERE IS IT WILL SETUP THE VECTOR AND START THE CLOCK
157                                     ;"CLKSTA" WILL INDICATE THE CLOCK TYPE
158                                     ; 0= NO CLOCK
159                                     ;+1= KW11-P
160                                     ;-1= KW11-L
161                                     ;THIS ROUTINE WILL ALSO SETUP "TICKMS" (TIME
162                                     ;PER CLOCK TICK IN MILLISECONDS) AND "TICKUS"
163                                     ;(TIME PER CLOCK TICK IN MICROSECONDS) AS
164                                     ;PER SW00.
165                                     ;SW00=0 -- 60HZ
166                                     ;SW00=1 -- 50HZ
167                                     ;CALL
```

```
                                        ;       JSR     PC,ST.CLK
                                        ;       RETURN

026156  010146                  ST.CLK: MOV     R1,-(SP)        ;SAVE R1
026160  012701  000006                  MOV     #ERRVEC+2,R1    ;SAVE AND SETUP TIMEOUT VECTOR
026164  011146                          MOV     (R1),-(SP)
026166  005011                          CLR     (R1)            ;LEVEL 0
026170  014146                          MOV     -(R1),-(SP)
026172  012711  026222                  MOV     #1$,(R1)        ;GO TO 1$ ON TIMEOUT
026176  005037  001340                  CLR     CLKSTA          ;SET CLOCK STATUS TO NO CLOCK
026202  005777  153274                  TST     @PKCS           ;IS THERE A KW11-P?
026206  012737  000001  001340          MOV     #1,CLKSTA       ;YES--SET STATUS TO KW11-P
026214  004737  026324                  JSR     PC,ST.PCLK      ;START THE KW11-P
026220  000414                          BR      3$              ;GO TO EXIT
026222  022626                  1$:     CMP     (SP)+,(SP)+     ;CLEAN UP THE STACK
026224  012711  026250                  MOV     #2$,(R1)        ;IF TIMEOUT GO TO 2$
026230  005777  153260                  TST     @LKS            ;IS THERE A KW11-L?
026234  012737  177777  001340          MOV     #-1,CLKSTA      ;YES-- SET STATUS TO KW11-L
026242  004737  026366                  JSR     PC,ST.LCLK      ;START THE KW11-L
026246  000401                          BR      3$              ;EXIT
026250  022626                  2$:     CMP     (SP)+,(SP)+     ;CLEAN UP THE STACK
026252  012621                  3$:     MOV     (SP)+,(R1)+     ;RESTORE THE TIMEOUT VECTOR
026254  012621                          MOV     (SP)+,(R1)+
026256  012601                          MOV     (SP)+,R1        ;RESTORE R1
026260  032737  000100  001314          BIT     #SW06,C.SWR     ;50HZ OR 60HZ?
026266  001407                          BEQ     4$              ;BRANCH IF 60
026270  012737  000020  001342          MOV     #20,TICKMS      ;SETUP TIME PER
026276  012737  047040  001344          MOV     #20000.,TICKUS  ;TICK FOR 50HZ
026304  000406                          BR      5$
026306  012737  000016  001342  4$:     MOV     #16,TICKMS      ;SETUP TIME PER
026314  012737  040432  001344          MOV     #16666.,TICKUS  ;TICK FOR 60HZ
026322  000207                  5$:     RTS     PC              ;RETURN

026324                          ST.PCLK:
026324  032737  000040  001314          BIT     #SW05,C.SWR     ;ALLOW SOFTWARE TIMEOUTS?
026332  001014                          BNE     1$              ;NO--BRANCH
026334  012777  026422  153134          MOV     #SRVCLK,@PKV    ;SETUP THE KW11-P VECTOR
026342  012777  000300  153130          MOV     #300,@PKV+2
026350  012777  000001  153126          MOV     #1,@PKB         ;COUNT ONE TICK
026356  012777  000115  153116          MOV     #115,@PKCS      ;"INT.EN.",COUNT DOWN", "MODE 1 (REPEAT)",
                                                                ;"LINE FREQ", AND "RUN"
026364  000207                  1$:     RTS     PC              ;RETURN

026366                          ST.LCLK:
026366  032737  000040  001314          BIT     #SW05,C.SWR     ;ALLOW SOFTWARE TIMEOUTS?
026374  001011                          BNE     1$              ;NO--BRANCH
026376  012777  026422  153104          MOV     #SRVCLK,@LKV    ;SETUP THE KW11-L VECTOR
026404  012777  000300  153100          MOV     #300,@LKV+2
026412  012777  000100  153074          MOV     #100,@LKS       ;START THE KW11-L
026420  000207                  1$:     RTS     PC              ;RETURN

026422  013746  001342          SRVCLK: MOV     TICKMS,-(SP)    ;TIME PER TICK IN MILLISECONDS
026426  004737  043324                  JSR     PC,RMTMR        ;COUNT THE ELASPED TIME
026432  000002                          RTI                     ;RETURN AFTER INTERRUPT

                                ;THIS ROUTINE SETS UP DEFAULT PARAMETER VALUES WHEN THE PROGRAM IS
                                ;STARTED OR WHEN THE VALUE OF BIT00 IN 'C.SWR' IS CHANGED.
```

```
                                        ;CALL
                                        ;       JSR     PC,LODFLT
                                        ;       RETURN
                                        LODFLT:
026434  010046                                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
026436  010146                                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
026440  010246                                  MOV     R2,-(SP)        ;;PUSH R2 ON STACK
026442  010346                                  MOV     R3,-(SP)        ;;PUSH R3 ON STACK
026444  012737  176777  001330                  MOV     #176777,TSTNMS  ;SELECT TESTS 0-10, 12-17
026452  012737  000003  001332                  MOV     #3,TSTNMS+2     ;SELECT TESTS 20 & 21
026460  012700  002514                          MOV     #DFLT,R0        ;DEFAULT PARAMETERS POINTER
026464  012701  003116                          MOV     #PRM0,R1        ;TABLE POINTER
026470  010102                                  MOV     R1,R2           ;STOP ADDRESS
026472  012021                          1$:     MOV     (R0)+,(R1)+     ;MOVE DEFAULT PARAMETERS INTO
026474  020002                                  CMP     R0,R2           ;RUN TIME TABLES ** DONE?
026476  103775                                  BLO     1$              ;NO--BRANCH
026500  012700  004160                          MOV     #PAT8,R0        ;PAT0 DEFAULTS TO PATTERN 8
026504  012701  003560                          MOV     #PAT0,R1
026510  012021                          2$:     MOV     (R0)+,(R1)+
026512  020027  004220                          CMP     R0,#PAT9
026516  103774                                  BLO     2$
026520  032737  000001  001314                  BIT     #BIT00,C.SWR    ;16 BIT MODE ?
026526  001012                                  BNE     3$              ;BR IF 18 BIT MODE
026530  012737  000037  002456                  MOV     #31.,PRMLMT+24  ;SET 'FS' LIMIT TO 31.
026536  012737  000037  002460                  MOV     #31.,PRMLMT+26  ;SET 'LS' LIMIT TO 31.
026544  012737  160000  001450                  MOV     #-<256.*32.>,TRCKWC     ;WORD COUNT FOR A 16 BIT TRACK
026552  000411                                  BR      4$              ;CONTINUE
026554  012737  000035  002456          3$:     MOV     #29.,PRMLMT+24  ;SET 'FS' LIMIT TO 29.
026562  012737  000035  002460                  MOV     #29.,PRMLMT+26  ;SET 'LS' LIMIT TO 29.
026570  012737  161000  001450                  MOV     #-<256.*30.>,TRCKWC     ;WORD COUNT FOR AN 18 BIT TRACK
026576  012701  002362                  4$:     MOV     #PRMPT,R1       ;ADDRESS OF PARAMETER POINTER TABLE
026602  005711                          5$:     TST     (R1)            ;END OF PARAMETER POINTER TABLE ?
026604  001425                                  BEQ     8$              ;BR IF YES
026606  032731  004000                          BIT     #BIT11,@(R1)+   ;IS 'LS' SELECTED AS A VARIABLE IN THIS TEST ?
026612  001773                                  BEQ     5$              ;BR IF NO
026614  016102  177776                          MOV     -2(R1),R2       ;GET FIRST POSITION IN PARAMETER TABLE
026620  011246                                  MOV     (R2),-(SP)      ;AND SAVE VARIABLES BITS THAT ARE USED.
026622  012703  000014                          MOV     #12.,R3         ;POSITION OF 'LS' IN TEST PARAMATER TABLE
026626  006216                          6$:     ASR     (SP)            ;IS THIS PARAMETER A VARIABLE ?
026630  103002                                  BCC     7$              ;BR IF NO
026632  062702  000002                          ADD     #2,R2           ;YES, POINT TO NEXT PARAMETER IN TABLE
026636  005303                          7$:     DEC     R3              ;AT 'LS' PARAMETER YET ?
026640  001372                                  BNE     6$              ;BR IF NO
026642  005726                                  TST     (SP)+           ;ADJUST THE STACK
026644  021237  002456                          CMP     (R2),PRMLMT+24  ;IS 'LS' TOO LARGE FOR THE MODE SELECTED ?
026650  101754                                  BLOS    5$              ;BR IF NO
026652  013712  002456                          MOV     PRMLMT+24,(R2)  ;RESET VALUE FOR MODE USED
026656  000751                                  BR      5$              ;CONTINUE
026660                                  8$:
026660  012603                                  MOV     (SP)+,R3        ;;POP STACK INTO R3
026662  012602                                  MOV     (SP)+,R2        ;;POP STACK INTO R2
026664  012601                                  MOV     (SP)+,R1        ;;POP STACK INTO R1
026666  012600                                  MOV     (SP)+,R0        ;;POP STACK INTO R0
026670  000207                                  RTS     PC              ;RETURN

                                        ;THIS POUTINE FILLS THE PARAMETER TABLE THE CURRENT TEST.
```

```
                                ;CALL
                                ;       MOV     #TESTNUM,$TSTNM ;LOAD THE TEST NUMBER
                                ;       JSR     PC,LODPRM
                                ;       RETURN

026672                          LODPRM:
026672  010146                         MOV     R1,-(SP)        ;;PUSH R1 ON STACK
026674  010246                         MOV     R2,-(SP)        ;;PUSH R2 ON STACK
026676  010346                         MOV     R3,-(SP)        ;;PUSH R3 ON STACK
026700  010446                         MOV     R4,-(SP)        ;;PUSH R4 ON STACK
026702  005004                         CLR     R4              ;CLEAR R4
026704  113704  001116                 MOVB    $TSTNM,R4       ;GET THE TEST NUMBER
026710  006304                         ASL     R4              ;SETUP TO ADDRESS WORDS
026712  016401  002362                 MOV     PRMPT(R4),R1    ;GET THE TEST'S PARAMETER TABLE ADDRESS
026716  012702  002322                 MOV     #PRM,R2         ;PARAMETER EXECUTION TABLE
026722  005003                         CLR     R3              ;R3 IS USED AS A COUNTER
026724  013704  001350                 MOV     CHKDRV,R4       ;GET DRIVE ADDRESS
026730  012122                         MOV     (R1)+,(R2)+     ;LOAD PARAMETER SPECIFIER
026732  006237  002322         1$:     ASR     PRM             ;IS THIS PARAMETER USED IN THE TEST ?
026736  103002                         BCC     2$              ;BR IF NOT
026740  012122                         MOV     (R1)+,(R2)+     ;LOAD THE VALUE
026742  000401                         BR      3$              ;CONTINUE
026744  005022                 2$:     CLR     (R2)+           ;CLEAR THE UNUSED PARAMETER LOCATION
026746  005203                 3$:     INC     R3              ;COUNT THE POSITION IN THE OUTPUT TABLE
026750  022702  002350                 CMP     #PAT+2,R2       ;FINISHED ?
026754  001437                         BEQ     7$              ;BR IF YES
026756  022703  000007                 CMP     #7,R3           ;DOING TRACK PARAMETERS ?
026762  001363                         BNE     1$              ;BR IF NO
026764  122764  000007  036750         CMPB    #7,DRVTYP(R4)   ;IS DEVICE AN RM05 ?
026772  001422                         BEQ     6$              ;IF SO, OVERLAY FT, LT & IT WITH FT', LT' & IT'
026774  013737  002444  001372         MOV     PRMLMT+12,LSTRK ;GET LAST TRACK FOR AN RM03/2
027002  062703  000003                 ADD     #3,R3           ;ADJUST COUNTER
027006  006237  002322                 ASR     PRM             ;COUNT THE PARAMETER
027012  103001                         BCC     4$              ;BR IF FT' IS NOT USED
027014  005721                         TST     (R1)+           ;MOVE THE INPUT POINTER
027016  006237  002322         4$:     ASR     PRM             ;COUNT THE PARAMETER
027022  103001                         BCC     5$              ;BR IF LT' NOT USED
027024  005721                         TST     (R1)+           ;MOVE THE INPUT POINTER
027026  006237  002322         5$:     ASR     PRM             ;COUNT THE PARAMETER
027032  103337                         BCC     1$              ;BR IF IT' NOT USED
027034  005721                         TST     (R1)+           ;MOVE THE INPUT POINTER
027036  000735                         BR      1$              ;KEEP GOING
027040  013737  002452  001372 6$:     MOV     PRMLMT+20,LSTRK ;GET LAST TRACK FOR AN RM05
027046  162702  000006                 SUB     #6,R2           ;BACKUP THE OUTPUT POINTER
027052  000727                         BR      1$              ;KEEP GOING
027054                          7$:
027054  012604                         MOV     (SP)+,R4        ;;POP STACK INTO R4
027056  012603                         MOV     (SP)+,R3        ;;POP STACK INTO R3
027060  012602                         MOV     (SP)+,R2        ;;POP STACK INTO R2
027062  012601                         MOV     (SP)+,R1        ;;POP STACK INTO R1
027064  000207                         RTS     PC              ;RETURN

                                ;THIS ROUTINE LOADS A READ HEADER AND DATA COMMAND OR A SEEK COMMAND
                                ;INTO DPB.B+2 AND DPB.C+2, DEPENDING ON THE STATE OF "CONTROL SWITCH"
                                ;BIT07.
                                ;CALL
                                ;       JSR     PC,LDCMD
```

```
                                        ;       RETURN
                                        ;
027066  032737  000200  001314  LDCMD:  BIT     #SW07,C.SWR     ;DO EXPLICIT SEEKS?
027074  001007                          BNE     1$              ;YES--BRANCH
027076  012737  000173  045500          MOV     #READHD,DPB.B+2 ;NO--SET UP FOR READ HEADER AND
027104  012737  000173  045520          MOV     #READHD,DPB.C+2 ;DATA COMMAND
027112  000406                          BR      2$
027114  012737  000105  045500  1$:     MOV     #SEEK,DPB.B+2   ;SETUP FOR SEEK COMMAND
027122  012737  000105  045520          MOV     #SEEK,DPB.C+2
027130  000207                  2$:     RTS     PC
                                        ;THIS ROUTINE WILL CALL THE RM05 DRIVER AND THEN WAIT ON THE FUNCTION
                                        ;TO COMPLETE. IF AN ERROR OCCURS IT IS REPORTED.
                                        ;CALL
                                        ;       FILL "DPB" WITH COMMAND INFORMATION
                                        ;       JSR     R0,CALL.A
                                        ;       RETURN
027132  005037  001222          CALL.A: CLR     $ESCAPE ;NO ESCAPE ADDRESS
027136  004037  037654                  JSR     R0,RM05         ;CALL RM05 DRIVER
027142  045456                          DPB.A
027144  000772                          BR      CALL.A
027146  005737  045474          1$:     TST     DPB.A+16        ;DONE?
027152  001775                          BEQ     1$              ;NO--LOOP
027154  100042                          BPL     4$              ;BRANCH IF NO ERROR
027156  012737  027252  001222          MOV     #3$,$ESCAPE     ;;ESCAPE TO 3$ ON ERROR
027164  013737  045470  001364          MOV     DPB.A+12,CYL.DS ;CYLINDER
027172  113737  045467  001370          MOVB    DPB.A+11,TRK.DS ;TRACK
027200  113737  045466  001366          MOVB    DPB.A+10,SEC.DS ;SECTOR
027206  012746  045474                  MOV     #DPB.A+16,-(SP) ;STATUS/ERROR INDICATOR ADDRESS
027212  004737  030526                  JSR     PC,ERINDX       ;FORM DISPATCH INDEX
027216  062607                          ADD     (SP)+,PC        ;REPORT PROPER ERROR
027220  104041                          EMT     41              ;NON-EXIST DRIVE
027222  104042                          EMT     42              ;PARITY ERROR
027224  104043                          EMT     43              ;UNSAFE ERROR
027226  104044                          EMT     44              ;NON-I/O ERROR
027230  000240                          NOP                     ;TO SYNC THE CALLING SEQ OF ERINDX
027232  005737  045572                  TST     RM.REG+RMER1            ;ANY DRIVE ERROR
027236  001004                          BNE     2$              ;BRANCH IF SO
027240  032737  100000  045620          BIT     #BIT15,RM.REG+RMER2     ;BAD SPOT ERROR
027246  001005                          BNE     4$              ;BRANCH IF SO
027250                          2$:
027250  104045                          EMT     45
027252  013746  04 474          3$:     MOV     DPB.A+16,-(SP)  ;STATUS WORD
027256  004737   0466                  JSR     PC,LOP.CK       ;SEE IF LOOP, ABORT, OR CONTINUE
027262  000200                  4$:     RTS     R0              ;RETURN
                                        ;THIS ROUTINE IS THE SAME AS "CALL.A" EXCEPT FOR THE DPB USED AND IF
                                        ;THE COMMAND IS A READ HEADER AND DATA THE HEADER (CYLINDER, TRACK,
                                        ;AND SECTOR) READ IS CHECKED FOR VALIDITY.
                                        ;CALL
                                        ;       FILL DPB
                                        ;       JSR     R0,CALL.B
                                        ;       RETURN
027264  005037  001222          CALL.B: CLR     $ESCAPE ;NO ESCAPE ADDRESS
027270  004037  037654                  JSR     R0,RM05         ;CALL RM05 DRIVER
```

```
                                                DPB.B
                                        BR      CALL.B
                                1$:     TST     DPB.B+16        ;DONE?
                                        BEQ     1$              ;NO--BRANCH
                                        BPL     4$              ;BRANCH IF NO ERROR
                                        MOV     #3$,$ESCAPE     ;;ESCAPE TO 3$ ON ERROR
                                        MOV     DPB.B+12,CYL.DS ;CYLINDER
                                        MOVB    DPB.B+11,TRK.DS ;TRACK
                                        MOVB    DPB.B+10,SEC.DS ;SECTOR
                                        MOV     #DPB.B+16,-(SP) ;STATUS/ERROR INDICATOR ADDRESS
                                        JSR     PC,ERINDX       ;FORM DISPATCH INDEX
                                        ADD     (SP)+,PC        ;REPORT PROPER ERROR
                                        EMT     41              ;NON-EXIST DRIVE
                                        EMT     42              ;PARITY ERROR
                                        EMT     43              ;UNSAFE ERROR
                                        EMT     44              ;NON-I/O ERROR
                                        NOP                     ;TO SYNC THE CALLING SEQ OF ERINDX: RT.
                                        TST     RM.REG+RMER1    ;DRIVE ERROR ?
                                        BEQ     2$              ;BR IF NOT
                                        BIT     #^C100,RM.REG+RMER1     ;SEE IF ONLY 'HCE' SET
                                        BEQ     4$              ;BR IF IT IS
                                2$:
                                        BIT     #BIT15,RM.REG+RMER2     ;BSE ERROR
                                        BNE     6$                      ;BRANCH IF SO
                                        EMT     45
                                3$:     MOV     DPB.B+16,-(SP)  ;STATUS WORD
                                        JSR     PC,LOP.CK       ;SEE IF LOOP, ABORT, OR CONTINUE
                                        BR      5$              ;CHECK FOR STALL
                                4$:     CMPB    DPB.B+2,#READHD ;DOING IMPLIED SEEKS?
                                        BNE     5$              ;NO--BRANCH
                                        JSR     R0,VERIFY       ;YES--GO CHECK THE DATA
                                        DPB.B+10
                                        BR      6$              ;ERROR DURING VERIFY
                                5$:     BIT     #SW14,C.SWR     ;STALL?
                                        BEQ     6$              ;NO--BRANCH
                                        JSR     R0,STALL        ;YES--CALL STALL ROUTINE
                                        .WORD   STALL1          ;STALL TIME POINTER
                                6$:     RTS     R0              ;RETURN

                                ;THIS ROUTINE IS THE SAME AS "CALL.B" EXCEPT FOR THE DPB USED.
                                ;CALL
                                ;       FILL DPB
                                ;       JSR     R0,CALL.C
                                ;       RETURN

                                CALL.C: CLR     $ESCAPE ;NO ESCAPE ADDRESS
                                        JSR     R0,RM05         ;CALL RM05 DRIVER
                                        DPB.C
                                        BR      CALL.C
                                1$:     TST     DPB.C+16        ;DONE?
                                        BEQ     1$              ;NO--LOOP
                                        BPL     4$              ;YES--BRANCH IF NO ERROR
                                        MOV     #3$,$ESCAPE     ;;ESCAPE TO 3$ ON ERROR
                                        MOV     DPB.C+12,CYL.DS ;CYLINDER
                                        MOVB    DPB.C+11,TRK.DS ;TRACK
                                        MOVB    DPB.C+10,SEC.DS ;SECTOR
                                        MOV     #DPB.C+16,-(SP) ;STATUS/ERROR INDICATOR ADDRESS
```

```
                027546  004737  030526                  JSR     PC,ERINDX       ;FORM DISPATCH INDEX
                027552  062607                          ADD     (SP)+,PC        ;REPORT PROPER ERROR
                027554  104041                          EMT     41              ;NON-EXIST DRIVE
                027556  104042                          EMT     42              ;PARITY ERROR
                027560  104043                          EMT     43              ;UNSAFE ERRGR
                027562  104044                          EMT     44              ;NON-I/O ERROR
    415         027564  000240                          NOP                     ;TO SYNC THE CALLING SEQ OF ERINDX: RT.
    416         027566  005737  045572                  TST     RM.REG+RMER1    ;DRIVE ERROR ?
    417         027572  001404                          BEQ     2$              ;BR IF NOT
    418         027574  032737  177677  045572          BIT     #^C100,RM.REG+RMER1     ;SEE IF ONLY 'HCE' SET
    419         027602  001412                          BEQ     4$              ;BR IF IT IS
    420         027604                          2$:
    421         027604  032737  100000  045620          BIT     #BIT15,RM.REG+RMER2     ;BSE ERROR ONLY ?
    422         027612  001025                          BNE     6$              ;BRANCH IF SO
    423         027614  104045                          EMT     45
    424         027616  013746  045534          3$:     MOV     DPB.C+16,-(SP)  ;STATUS WORD
    425         027622  004737  030466                  JSR     PC,LOP.CK       ;SEE IF LOOP, ABORT, OR CONTINUE
    426         027626  000410                          BR      5$
    427         027630  123727  045520  000173  4$:     CMPB    DPB.C+2,#READHD ;DOING IMPLIED SEEK?
    428         027636  001004                          BNE     5$              ;NO--EXIT
    429         027640  004037  030746                  JSR     R0,VERIFY       ;YES--CHECK THE DATA
    430         027644  045526                          DPB.C+10
    431         027646  000407                          BR      6$              ;ERROR DURING VERIFY
    432         027650  032737  040000  001314  5$:     BIT     #SW14,C.SWR     ;STALL?
    433         027656  001403                          BEQ     6$              ;NO--BRANCH
    434         027660  004037  030664                  JSR     R0,STALL        ;YES--CALL STALL ROUTINE
    435         027664  001452                          .WORD   STALL1          ;STALL TIME POINTER
    436         027666  000200                  6$:     RTS     R0
    437
    438
    439                                         ;THIS ROUTINE IS THE SAME AS "CALL.A" EXCEPT FOR THE DPB USED AND
    440                                         ;ON AN ERROR LOCATION "ERR.CT" IS EXAMINED. IF ERR.CT IS EQUAL TO
    441                                         ;$ERFLG EXIT IS TO THE NEXT TEST.
    442                                         ;CALL
    443                                         ;       FILL DPB
    444                                         ;       JSR     R0,DRVCAL
    445                                         ;       RETURN
    446
    447 027670  005037  001222          DRVCAL: CLR     $ESCAPE ;NO ESCAPE ADDRESS
    448 027674  005037  001432                  CLR     WCEFLG  ;CLEAR WRITE CHECK ERROR FLAG
    449 027700  004037  037654                  JSR     R0,RM05         ;CALL RM05 DRIVER
    450 027704  045536                          DTADPB
    451 027706  000770                          BR      DRVCAL
    452
    453 027710  005737  045554          DRVCL1: TST     DTADPB+16       ;DONE
    454 027714  001775                          BEQ     DRVCL1          ;NO--LOOP
    455 027716  100402                          BMI     1$              ;BR IF ERRORS
    456 027720  000137  030446                  JMP     14$             ;NO ERRORS
    457 027724                          1$:
        027724  012737  030032  001222          MOV     #3$,$ESCAPE     ;;ESCAPE TO 3$ ON ERROR
    458 027732  013737  045550  001364          MOV     DTADPB+12,CYL.DS        ;CYLINDER
        027740  113737  045547  001370          MOVB    DTADPB+11,TRK.DS        ;TRACK
        027746  113737  045546  001366          MOVB    DTADPB+10,SEC.DS        ;SECTOR
        027754  012746  045554                  MOV     #DTADPB+16,-(SP)        ;STATUS/ERROR INDICATOR ADDRESS
        027760  004737  030526                  JSR     PC,ERINDX       ;FORM DISPATCH INDEX
        027764  062607                          ADD     (SP)+,PC        ;REPORT PROPER ERROR
        027766  104041                          EMT     41              ;NON-EXIST DRIVE
```

```
        027770  104042                          EMT     42              ;PARITY ERROR
        027772  104043                          EMT     43              ;UNSAFE ERROR
        027774  104044                          EMT     44              ;NON-I/O ERROR
    459 027776  000240                          NOP                     ;TO SYN THE CALLING SEQ OF THE ERINDX:
    460 030000  005737  045572                  TST     RM.REG+RMER1    ;ANY DRIVE ERROR ?
    461 030004  001011                          BNE     2$              ;REPORT THE I/O ERROR IF SO
    462 030006  032737  100000  045620          BIT     #BIT15,RM.REG+RMER2     ;BAD SPOT ERROR ?
    463 030014  001405                          BEQ     2$              ;BRANCH IF NOT
    464 030016  012737  177777  001464          MOV     #-1,BASFLG      ;SET BAD SECTOR ENCOUNTER FLAG
    465 030024  000137  030446                  JMP     14$             ;OTHERWISE ,DON'T REPORT THE BSE
    466 030030                          2$:
        030030  104045                          EMT     45
    467 030032  122737  000020  001116  3$:     CMPB    #20,$TSTNM      ;TEST 20?
    468 030040  001170                          BNE     11$             ;NO--BRANCH
    469 030042  013746  045554                  MOV     DTADPB+16,-(SP) ;STATUS WORD
    470 030046  004737  030466                  JSR     PC,LOP.CK       ;SEE IF LOOP, ABORT, OR CONTINUE
    471 030052  122737  000151  045540          CMPB    #WRCKD,DTADPB+2 ;DOING A WRITE CHECK?
    472 030060  001164                          BNE     12$             ;NO--BRANCH
    473 030062  032737  040000  045566          BIT     #BIT14,RM.REG+10        ;IS ''WCE''=1?
    474 030070  001560                          BEQ     12$             ;NO--BRANCH
    475 030072  032777  000020  151054          BIT     #SW04,@SWR      ;INHIBIT WRITES?
    476 030100  001154                          BNE     12$             ;YES--BRANCH
    477 030102  112737  000161  045540          MOVB    #WRITE,DTADPB+2 ;SETUP FOR A WRITE
    478 030110  005037  001222                  CLR     $ESCAPE ;NO ESCAPE ADDRESS
    479 030114  004037  037654                  JSR     R0,RM05         ;DO THE WRITE
    480 030120  045536                          DTADPB
    481 030122  000240                          NOP
    482 030124  005737  045554          4$:     TST     DTADPB+16       ;DONE?
    483 030130  001775                          BEQ     4$              ;NO--LOOP
    484 030132  100043                          BPL     6$              ;YES--BRANCH IF NO ERROR
    485 030134  012737  030422  001222          MOV     #11$,$ESCAPE    ;;ESCAPE TO 11$ ON ERROR
    486 030142  013737  045550  001364          MOV     DTADPB+12,CYL.DS        ;CYLINDER
        030150  113737  045547  001370          MOVB    DTADPB+11,TRK.DS        ;TRACK
        030156  113737  045546  001366          MOVB    DTADPB+10,SEC.DS        ;SECTOR
        030164  012746  045554                  MOV     #DTADPB+16,-(SP)        ;STATUS/ERROR INDICATOR ADDRESS
        030170  004737  030526                  JSR     PC,ERINDX       ;FORM DISPATCH INDEX
        030174  062607                          ADD     (SP)+,PC        ;REPORT PROPER ERROR
        030176  104041                          EMT     41              ;NON-EXIST DRIVE
        030200  104042                          EMT     42              ;PARITY ERROR
        030202  104043                          EMT     43              ;UNSAFE ERROR
        030204  104044                          EMT     44              ;NON-I/O ERROR
    487 030206  000240                          NOP                     ;TO SYN THE CALLING SEQ OF ERINDX
    488 030210  005737  045572                  TST     RM.REG+RMER1    ;ANY DRIVE ERROR ?
    489 030214  001011                          BNE     5$              ;BRANCH IF SO
    490 030216  032737  100000  045620          BIT     #BIT15,RM.REG+RMER2     ;BAD SOPT ERROR
    491 030224  001405                          BEQ     5$              ;BRANCH IF NOT
    492 030226  012737  177777  001464          MOV     #-1,BASFLG      ;SET BAD SECTOR ENCOUNTER FLAG
    493 030234  000137  030446                  JMP     14$             ;EXIT
    494 030240                          5$:
        030240  104045                          EMT     45
    495 030242  112737  000151  045540  6$:     MOVB    #WRCKD,DTADPB+2 ;COMMAND=WRITE CHECK DATA
    496 030250  004037  037654                  JSR     R0,RM05         ;DO THE WRITE CHECK
    497 030254  045536                          DTADPB
    498 030256  000240                          NOP
    499 030260  005737  045554          7$:     TST     DTADPB+16       ;DONE?
    500 030264  001775                          BEQ     7$              ;NO--LOOP
    501 030266  100410                          BMI     9$              ;YES--BRANCH IF ERROR
```

```
                                                JSR     R0,RM05         ;DO A 2ND WRITE CHECK
                                                DTADPB
                                                NOP
                                        8$:     TST     DTADPB+16       ;DONE?
                                                BEQ     8$              ;NO--LOOP
                                                BPL     14$             ;YES--BRANCH IF NO ERROR
                                        9$:     MOV     #1,WCEFLG       ;SET THE WRITE CHECK ERROR FLAG
                                                MOV     #11$,$ESCAPE    ;;ESCAPE TO 11$ ON ERROR
                                                MOV     DTADPB+12,CYL.DS        ;CYLINDER
                                                MOVB    DTADPB+11,TRK.DS        ;TRACK
                                                MOVB    DTADPB+10,SEC.DS        ;SECTOR
                                                MOV     #DTADPB+16,-(SP)        ;STATUS/ERROR INDICATOR ADDRESS
                                                JSR     PC,ERINDX       ;FORM DISPATCH INDEX
                                                ADD     (SP)+,PC        ;REPORT PROPER ERROR
                                                EMT     41              ;NON-EXIST DRIVE
                                                EMT     42              ;PARITY ERROR
                                                EMT     43              ;UNSAFE ERROR
                                                EMT     44              ;NON-I/O ERROR
                                                NOP                     ;SO SYN THE CALLING SEQ OF ERINDX:
                                                TST     RM.REG+RMER1    ;ANY DRIVE ERROR
                                                BNE     10$             ;BRANCH IF SO
                                                BIT     #BIT15,RM.REG+RMER2     ;BAD SOPT ERROR ?
                                                BEQ     10$             ;BRANCH IF NOT
                                                MOV     #-1,BASFLG      ;SET BAD SECTOR ENCOUNTER FLAG
                                                BR      14$             ;OTHERWISE EXIT
                                        10$:
                                                EMT     46
                                        11$:    MOV     DTADPB+16,-(SP) ;STATUS WORD
                                                JSR     PC,LOP.CK       ;SEE IF LOOP, ABORT, OR CONTINUE
                                        12$:    CMPB    ERR.CT,$ERFLG   ;GO TO NEXT TEST?
                                                BHI     14$             ;NO--BRANCH
                                        13$:    MOV     BYPASS,R0       ;YES--GET EXIT ADDRESS
                                        14$:    BIT     #SW14,C.SWR     ;STALL?
                                                BEQ     15$             ;NO--BRANCH
                                                JSR     R0,STALL        ;YES--CALL STALL ROUTINE
                                                .WORD   STALL2          ;STALL TIME POINTER
                                        15$:    RTS     R0

                                        ;THIS SUBROUTINE CHECK FOR LOOP, ABORT, OR CONTINUE SWITCHES AFTER
                                        ;ERRORS 41, 42, 43, 44, 45, AND 46.
                                        ;CALL
                                        ;       MOV     DTA+16,-(SP)    ;STATUS WORD FROM DPB IN USE
                                        ;       JSR     PC,LOP.CK
                                        ;       RETURN

                                        LOP.CK: BIT     #SW9,@SWR       ;LOOP ON ERROR
                                                BEQ     1$              ;BR IF NOT
                                                JMP     @$LPERR         ;START AT THE LOOP ADDRESS
                                        1$:     CLR     $ESCAPE         ;CLEAR ERROR ESCAPE FLAG
                                                BIT     #BIT14!BIT13!BIT12!BIT10!BIT02!BIT01,2(SP)      ;CHECK ERROR TYPE
                                                BEQ     2$                      ;BR IF DRIVE NOT OFFLINE, UNLOADED, OR
                                                                                ;PERSISTENT UNSAFE OR FATAL MASSBUS PARITY
                                                JMP     $EOP                    ;TERMINATE DRIVE
                                        2$:     MOV     (SP)+,(SP)      ;ADJUST RETURN ADDRESS
                                                RTS     PC

                                        ;THIS ROUTINE FORMS AN INDEX THAT WILL BE USED TO DISPATCH
```

```
                                ;TO THE PROPER ERROR CALL.  THE INDEX IS FORMED BY EXAMINING
                                ;THE STATUS/ERROR INDICATOR OF THE APPLICABLE DPB.
                                ;INDEX          STATUS/ERROR
                                ;-----          ------------
                                ;     0 BIT14!BIT13!BIT08!BIT01
                                ;     2 BIT11!BIT10!BIT02
                                ;     4 BIT12!BIT04
                                ;     6 BIT05!BIT03!<BIT09 & COMMAND=NON-I/O>
                                ;    10 BIT06!<BIT09 & COMMAND=I/O>
                                ;CALL
                                ;       JSR     #DPB+16,-(SP)   ;ADDRESS OF STATUS/ERROR INDICATOR
                                ;       JSR     PC,ERINDX       ;FORM INDEX
                                ;       RETURN                  ;INDEX IS ON THE STACK

030526  010046                  ERINDX: MOV     R0,-(SP)        ;SAVE R0
030530  010146                          MOV     R1,-(SP)        ;SAVE R1
030532  016600  000006                  MOV     6(SP),R0        ;GET STATUS/ERROR INDICATOR POINTER
030536  011037  001354                  MOV     (R0),SVSTAT     ;SAVE THE STATUS/ERROR INDICATOR
030542  005001                          CLR     R1              ;START INDEX AT ZERO
030544  032710                          BIT     (PC)+,(R0)      ;FORM INDEX OF 0?
030546  020402                          .WORD   BIT13!BIT08!BIT01
030550  001037                          BNE     5$              ;YES--BRANCH
030552  032710                          BIT     (PC)+,(R0)      ;FORM PARITY ERROR OR PORT REQUEST INDEX (2)?
030554  006004                          .WORD   BIT11!BIT10!BIT02
030556  001033                          BNE     4$              ;YES--BRANCH
030560  032710                          BIT     (PC)+,(R0)      ;FORM UNSAFE INDEX (4)?
030562  050020                          .WORD   BIT14!BIT12!BIT04
030564  001027                          BNE     3$              ;YES--BRANCH
030566  032710                          BIT     (PC)+,(R0)      ;FORM NON-I/O ERROR INDEX (6)?
030570  000050                          .WORD   BIT05!BIT03
030572  001023                          BNE     2$              ;YES--BRANCH
030574  032710                          BIT     (PC)+,(R0)      ;FORM I/O ERROR INDEX (10)?
030576  000100                          .WORD   BIT06
030600  001017                          BNE     1$              ;YES--BRANCH
030602  032710                          BIT     (PC)+,(R0)      ;SOFTWARE TIMEOUT?
030604  001000                          .WORD   BIT09
030606  001420                          BEQ     5$              ;NO--FORM INDEX OF 0
030610  122760  000150  177762          CMPB    #150,-16(R0)    ;YES--I/O?
030616  003011                          BGT     2$              ;NO--BRANCH
030620  005737  045572                  TST     RM.REG+RMER1    ;ANY DRIVE ERROR ?
030624  001005                          BNE     1$              ;BRANCH IF SO
030626  032737  100000  045620          BIT     #BIT15,RM.REG+RMER2     ;BSE ERROR
030634  001401                          BEQ     1$              ;BRANCH IF NOT
030636  005201                          INC     R1              ;SKIP , NOT REPORT BSE ERROR
030640  005201                  1$:     INC     R1              ;INDEX=10---ERROR=45 OR 46
030642  005201                  2$:     INC     R1              ;INDEX=6---ERROR=44
030644  005201                  3$:     INC     R1              ;INDEX=4---ERROR=43
030646  005201                  4$:     INC     R1              ;INDEX=2---ERROR=42
030650  006301                  5$:     ASL     R1              ;INDEX=0---ERROR=41
030652  010166  000006                  MOV     R1,6(SP)        ;RETURN INDEX TO USER
030656  012601                          MOV     (SP)+,R1        ;RESTORE R1
030660  012600                          MOV     (SP)+,R0        ;RESTORE R0
030662  000207                          RTS     PC              ;RETURN FROM CALL

                                ;THIS ROUTINE WILL PROVIDE A STALL IN MILLISECONDS FOR A SPECIFIC
                                ;AMOUNT OF TIME IF BIT13 OF C.SWR = 0 OR A RANDOM AMOUNT OF TIME
                                ;IF BIT 13 OF C.SWR = 1.
```

```
                                        ;STALL1 CONTAINS SPECIFIED TIME FOR TESTS 0 - 7, AND STALL2
                                        ;CONTAINS THE TIME FOR TESTS 16-21.
                                        ;CALL
                                        ;       JSR     R0,STALL
                                        ;       TIME POINTER            ;WHERE TO FIND THE STALL TIME

030664  013046                  STALL:  MOV     @(R0)+,-(SP)    ;PICKUP STALL TIME
030666  032737  020000  001314          BIT     #SW13,C.SWR     ;USE A RANDOM TIME?
030674  001406                          BEQ     1$              ;NO--BRANCH
030676  004737  025550                  JSR     PC,$RAND        ;YES--FORM RANDOM NUMBER
030702  013716  025650                  MOV     $LONUM,(SP)     ;AND USE IT FOR THE STALL TIME
030706  042716  177700                  BIC     #^C77,(SP)      ;BUT NEVER > 64 MILLISECONDS
030712  005046                  1$:     CLR     -(SP)           ;CLEAR TEMP. LOCATION
030714  162766  000001  000002  2$:     SUB     #1,2(SP)        ;MORE STALL REQUIRED?
030722  103407                          BLO     4$              ;NO--BRANCH
030724  012716  000144                  MOV     #100.,(SP)      ;STALL FOR ABOUT 1 MILLISECOND
030730  005700                  3$:     TST     R0              ;NOP TO KILL TIME
030732  005366  000000                  DEC     0(SP)           ;COUNT
030736  001374                          BNE     3$              ;LOOP IF MORE COUNTS NEEDED
030740  000765                          BR      2$
030742  022626                  4$:     CMP     (SP)+,(SP)+     ;CLEAN OFF THE STACK
030744  000200                          RTS     R0              ;EXIT

                                        ;ROUTINE TO SOFTWARE COMPARE HEADER ON IMPLIED SEEKS.
                                        ;CALL
                                        ;       JSR     R0,VERIFY
                                        ;       ADR POINTER             ;ADDRESS OF DPB+10 (SECTOR NUMBER)
                                        ;       RETURN

030746  010146                  VERIFY: MOV     R1,-(SP)        ;SAVE R1
030750  012001                          MOV     (R0)+,R1        ;GET ADDRESS OF DPB+10
030752  042737  150000  052776          BIC     #150000,BUFFER  ;STRIP FORMAT AND BAD SECTOR BITS FROM CYLINDER NUMBER
030760  023761  052776  000002          CMP     BUFFER,2(R1)    ;CYLINDER NUMBER OK?
030766  001003                          BNE     1$              ;NO--BRANCH
030770  023711  053000                  CMP     BUFFER+2,(R1)   ;YES--HOW ABOUT TRACK/SECTOR?
030774  001441                          BEQ     3$              ;BRANCH IF GOOD
030776  013737  052776  001356  1$:     MOV     BUFFER,CYL.RD   ;SAVE THE EXPECTED AND THE
031004  113737  053001  001360          MOVB    BUFFER+3,TRK.RD ;RECIEVED CYLINDER, TRACK,
031012  113737  053000  001362          MOVB    BUFFER+2,SEC.RD ;AND SECTOR
031020  112137  001366                  MOVB    (R1)+,SEC.DS
031024  112137  001370                  MOVB    (R1)+,TRK.DS
031030  011137  001364                  MOV     (R1),CYL.DS
031034  012737  031046  001222          MOV     #2$,$ESCAPE     ;;ESCAPE TO 2$ ON ERROR
031042  005740                          TST     -(R0)           ;MAKE IT TEST PC+4
031044  104012                          EMT     12
031046  012737  000107  045460  2$:     MOV     #RECAL,DPB.A+2  ;LOAD RECALIBRATE ORDER CODE
031054  004037  027132                  JSR     R0,CALL.A       ;GO EXECUTE THE COMMAND
031060  005037  001222                  CLR     $ESCAPE         ;CLEAR ERROR ESCAPE FLAG
031064  032777  001000  150062          BIT     #SW9,@SWR       ;LOOP ON ERROR ?
031072  001404                          BEQ     4$              ;BR IF NOT
031074  000177  150024                  JMP     @$LPERR         ;RETURN TO ERROR LOOP ADDRESS
031100  062700  000002          3$:     ADD     #2,R0           ;INCREMENT RETURN ADDRESS
031104  012601                  4$:     MOV     (SP)+,R1        ;RESTORE R1
031106  000200                          RTS     R0              ;EXIT

                                        ;THIS ROUTINE WILL PERFORM A "MASSBUS" INIT. FOLLOWED BY
                                        ;A "RECALIBRATE" ON THE DRIVE UNDER TEST.
```

```
                                        ;NOTE: THIS ROUTINE DESTROYS R1 AND R4
                                        ;CALL
                                        ;       JSR     R0,SRCH00       ;DO A MASSBUS INIT. AND RECAL
                                        ;       RETURN1                 ;RETURN HERE IF NO ERROR
                                        ;       RETURN2                 ;RETURN HERE ON ERROR

031110  005001                  SRCH00: CLR     R1              ;INCASE OF ERROR (TYPTIM)
031112  005037  177776                  CLR     PS
031116  012777  041740  005744          MOV     #ISR,@RMVEC     ;SETUP INTERRUPT VECTOR
031124  013704  037066                  MOV     RMADR,R4        ;PICKUP ADDRESS OF RMCS1
031130  012764  000040  000010          MOV     #CLR,RMCS2(R4)  ;MASSBUS INIT.
031136  005037  045546                  CLR     DTADPB+10       ;TRACK=0; SECTOR=0
031142  005037  045550                  CLR     DTADPB+12       ;CYLINDER =0
031146  012737  000107  045540          MOV     #RECAL,DTADPB+2 ;COMMAND = RECALIBRATE
031154  005037  001222                  CLR     $ESCAPE         ;NO ESCAPE ADDRESS
031160  004037  037654                  JSR     R0,RM05         ;CALL THE DRIVER
031164  045536                          DTADPB                  ;DPB POINTER
031166  000440                          BR      4$              ;QUEUE IS FULL
031170  005737  045554          1$:     TST     DTADPB+16       ;WAIT ON DONE
031174  001775                          BEQ     1$
031176  100030                          BPL     3$              ;TAKE NORMAL EXIT IF NO ERROR
031200  012737  031254  001222          MOV     #2$,$ESCAPE     ;;ESCAPE TO 2$ ON ERROR
031206  013737  045550  001364          MOV     DTADPB+12,CYL.DS        ;CYLINDER
031214  113737  045547  001370          MOVB    DTADPB+11,TRK.DS        ;TRACK
031222  113737  045546  001366          MOVB    DTADPB+10,SEC.DS        ;SECTOR
031230  012746  045554                  MOV     #DTADPB+16,-(SP)        ;STATUS/ERROR INDICATOR ADDRESS
031234  004737  030526                  JSR     PC,ERINDX       ;FORM DISPATCH INDEX
031240  062607                          ADD     (SP)+,PC        ;REPORT PROPER ERROR
031242  104041                          EMT     41              ;NON-EXIST DRIVE
031244  104042                          EMT     42              ;PARITY ERROR
031246  104043                          EMT     43              ;UNSAFE ERROR
031250  104044                          EMT     44              ;NON-I/O ERROR
031252  104045                          EMT     45              ;I/O ERROR
031254  005720                  2$:     TST     (R0)+           ;ADJUST FOR ERROR EXIT
031256  000404                          BR      4$              ;GO TO THE EXIT
031260  005064  000006          3$:     CLR     RMDA(R4)        ;TRACK AND SECTOR = 0
031264  005064  000034                  CLR     RMDC(R4)        ;CYLINDER = 0
031270  000200                  4$:     RTS     R0              ;RETURN

                                        ;THIS IS AN RTI WHICH IS USED BY THE TIMING TESTS & THE SERVO SETTLE DOWN TEST

031272  000002                  DORTI:  RTI                     ;RETURN FROM INTERRUPT

                                        ;THIS ROUTINE WILL INITIALIZE THE TIMERS USED BY THE ''TIMING ROUTI ES
                                        ;CALL
                                        ;       JSR     PC,STRTMR
                                        ;       RETURN

031274  104412                  STRTMR: SAVREG                  ;SAVE R0-R5
031276  012700  001374                  MOV     #TIM.UP,R0      ;START AT TIM.UP
031302  005020                  1$:     CLR     (R0)+           ;CLEAR THE TIME TABLES
031304  020027  001430                  CMP     R0,#TIM.PT      ;DONE?
031310  103774                          BLO     1$              ;NO--BRANCH
031312  012710  052776                  MOV     #BUFFER,(R0)    ;SETUP POINTER
031316  012737  077777  001374          MOV     #^CBIT15,TIM.UP ;SET MINIMUM TIME TO MAXIMUM
031324  012737  077777  001412          MOV     #^CBIT15,TIM.DN ;POSITIVE NUMBER
031332  104413                          RESREG                  ;RESTORE R0-R5
```

```
031334  000207                          RTS     PC              ;RETURN

                                ;THIS ROUTINE WILL ADD THE ELAPSED TIME TO THE AVERAGE COUNTER AND
                                ;MAINTAIN THE MINIMUM AND MAXIMUM TIMES.
                                ;NOTE: THIS ROUTINE DESTROYS R2
                                ;CALL
                                ;       MOV     #TP,R3          ;PARAMETER POINTER
                                ;       MOV     FLAG,R5         ;FLAG=0=COUNT UP
                                ;                               ;FLAG=-1=COUNT DOWN
                                ;       JSR     PC,COUNT
                                ;       RETURN

031336  012702  001374          COUNT:  MOV     #TIM.UP,R2      ;PICKUP THE "UP" POINTER
031342  005705                          TST     R5              ;USE IT?
031344  001402                          BEQ     1$              ;YES--BRANCH
031346  012702  001412                  MOV     #TIM.DN,R2      ;NO--PICKUP "DOWN" POINTER
031352  027722  150130          1$:     CMP     @PKC,(R2)+      ;LESS THAN PREVIOUS LOW?
031356  002003                          BGE     2$              ;NO--BRANCH
031360  017762  150122  177776          MOV     @PKC,-2(R2)     ;YES--SAVE IT
031366  027763  150114  000004  2$:     CMP     @PKC,4(R3)      ;LESS THAN THE LOW LIMIT?
031374  002001                          BGE     3$              ;NO--BRANCH
031376  005212                          INC     (R2)            ;YES--COUNT IT
031400  005722                  3$:     TST     (R2)+           ;ADVANCE THE POINTER
031402  027722  150100                  CMP     @PKC,(R2)+      ;GREATER THAN PREVIOUS HIGH?
031406  003403                          BLE     4$              ;NO--BRANCH
031410  017762  150072  177776          MOV     @PKC,-2(R2)     ;YES--SAVE IT
031416  027763  150064  000006  4$:     CMP     @PKC,6(R3)      ;GREATER THAN THE HIGH LIMIT?
031424  003401                          BLE     5$              ;NO--BRANCH
031426  005212                          INC     (R2)            ;YES--COUNT IT
031430  005722                  5$:     TST     (R2)+           ;ADVANCE THE POINTER
031432  067722  150050                  ADD     @PKC,(R2)+      ;ADD THIS COUNT TO THE .TOTAL
031436  005522                          ADC     (R2)+
031440  005212                          INC     (R2)            ;COUNT THIS READING
031442  022737  061326  001430          CMP     #BUFFER+<4*822.>,TIM.PT ;SAVE THIS COUNT?
031450  101406                          BLOS    6$              ;NO--BRANCH
031452  017777  150030  147750          MOV     @PKC,@TIM.PT    ;YES--WELL SAVE IT THEN
031460  062737  000002  001430          ADD     #2,TIM.PT       ;ADVANCE THE POINTER
031466  000207                  6$:     RTS     PC              ;RETURN

                                ;THIS ROUTINE PRINTS THE SPEC OF ALL TIMING TESTS
                                ;CALL
                                ;       JSR     R0,SPTYP
                                ;       TABLE ADDRESS
                                ;
                                ;TABLE: .WORD   ASCIZ MESSAGE POINTER
                                ;       .WORD   MIN VALUE
                                ;       .WORD   MAX VALUE

031470  012002                  SPTYP:  MOV     (R0)+,R2        ;THE TABLE ADDRESS
031472  032777  000100  147454          BIT     #SW06,@SWR      ;ALLOW PRINT
031500  001035                          BNE     3$              ;EXIT IF NOT
031502  104401  001231                  TYPE    ,$CRLF
031506  104401  001231                  TYPE    ,$CRLF
031512  012237  031520                  MOV     (R2)+,1$        ;
031516  104401                          TYPE
031520  000000                  1$:     .WORD   0
031522  012246                          MOV     (R2)+,-(SP)     ;LOAD MIN VALUE
```

```
                                        BEQ     2$              ;SKIP IF MIN VALUE IS 0
                                        TYPE    ,MSGMIN
                                        JSR     PC,$SB2D        ;CONVERT TO DECIMAL
                                        JSR     PC,$SUPRS       ;TYPE IT
                                        TYPE    ,MSGOUS         ;0 US
                                2$:     TYPE    ,MSGMAX         ;
                                        MOV     (R2),-(SP)      ;MAXMUM VALUE
                                        JSR     PC,$SB2D        ;
                                        JSR     PC,$SUPRS       ;
                                        TYPE    ,MSGOUS         ;
                                        TYPE    ,$CRLF          ;CR-LF
                                3$:     RTS     R0              ;


                                ;;THIS ROUTINE IS USED TO TYPE THE MINIMUM,
                                ;MAXIMUM, AND AVERAGE TIMES FOR THE TIMING TESTS
                                ;IT WILL ALSO CHECK THE TIMES TO ENSURE
                                ;THEY ARE WITHIN TOLERANCE AND IF NOT FLAG THE BAD TIMES.
                                ;NOTE: THIS ROUTINE DESTROYS R2-R5
                                ;CALL
                                ;       JSR     R0,TYPTIM       ;GO REPORT THE TIMES
                                ;       TABLE                   ;POINT TO THE PROPER TABLE
                                ;       RETURN
                                ;
                                ;TABLE: MSGADR1                 ;ADDRESS OF ASCIZ MESSAGE NUMBER 1
                                ;       MSGADR2                 ;ADDRESS OF ASCIZ MESSAGE NUMBER 2
                                ;       MIN.ALLOWED             ;MINIMUM TIME ALLOWED
                                ;       MAX.ALLOWED             ;MAXIMUM TIME ALLOWED

                                TYPTIM: MOV     (R0)+,R2        ;PICKUP THE TABLE POINTER
                                        BIT     #SW06,@SWR      ;INHIBIT TIME REPORTS?
                                        BNE     9$              ;YES--BRANCH
                                        MOV     (R2)+,2$        ;ADDRESS OF MESSAGE NUMBER 1
                                        MOV     (R2)+,R5        ;ADDRESS OF MESSAGE NUMBER 2
                                        MOV     (R2)+,R3        ;PICKUP THE LOW LIMIT
                                        MOV     (R2),R2         ;AND THE HIGH LIMIT
                                        MOV     #TIM.UP,R4      ;PARAMETER POINTER
                                1$:     TYPE                    ;TYPE THE MESSAGE
                                2$:     .WORD   0               ;ASCIZ MESSAGE POINTER GOES HERE
                                        TST     14(R4)          ;DID ANY COUNTS OCCUR?
                                        BEQ     8$              ;NO--BRANCH
                                        TYPE    ,MSGMIN         ;"MIN="
                                        MOV     (R4)+,-(SP)     ;PUT (R4)+ ON THE STACK
                                        JSR     PC,$SB2D        ;CHANGE TO DECIMAL ASCIZ
                                        JSR     PC,$SUPRS       ;TYPE WITHOUT LEADING ZEROS
                                        TYPE    ,MSGOUS         ;"0 US"
                                        TST     (R4)+           ;ANY SEEKS BELOW THE LOW LIMIT
                                        BEQ     3$              ;NO--BRANCH
                                        TYPE    ,BLNKS2         ;TYPE 2 SPACES
                                        MOV     -2(R4),-(SP)    ;PUT -2(R4) ON THE STACK
                                        JSR     PC,$SB2D        ;CHANGE TO DECIMAL ASCIZ
                                        JSR     PC,$SUPRS       ;TYPE WITHOUT LEADING ZEROS
                                        TYPE    ,MBELOW         ;"BELOW THE MINIMUM OF"
                                        MOV     R3,-(SP)        ;PUT R3 ON THE STACK
                                        JSR     PC,$SB2D        ;CHANGE TO DECIMAL ASCIZ
                                        JSR     PC,$SUPRS       ;TYPE WITHOUT LEADING ZEROS
```

```
                                                TYPE    ,MSGOUS
                                3$:             TYPE    ,MSGMAX         ;"MAX="
                                                MOV     (R4)+,-(SP)     ;PUT (R4)+ ON THE STACK
                                                JSR     PC,$SB2D        ;CHANGE TO DECIMAL ASCIZ
                                                JSR     PC,$SUPRS       ;TYPE WITHOUT LEADING ZEROS
                                                TYPE    ,MSGOUS
                                                TST     (R4)+           ;ANY SEEKS ABOVE THE HIGH LIMIT
                                                BEQ     4$              ;NO--BRANCH
                                                TYPE    ,BLNKS2         ;TYPE 2 SPACES
                                                MOV     -2(R4),-(SP)    ;PUT -2(R4) ON THE STACK
                                                JSR     PC,$SB2D        ;CHANGE TO DECIMAL ASCIZ
                                                JSR     PC,$SUPRS       ;TYPE WITHOUT LEADING ZEROS
                                                TYPE    ,MABOVE         ;"ABOVE THE MAXIMUM OF"
                                                MOV     R2,-(SP)        ;PUT R2 ON THE STACK
                                                JSR     PC,$SB2D        ;CHANGE TO DECIMAL ASCIZ
                                                JSR     PC,$SUPRS       ;TYPE WITHOUT LEADING ZEROS
                                                TYPE    ,MSGOUS
                                4$:             TYPE    ,MSGAVG         ;"AVG="
                                                MOV     (R4)+,-(SP)     ;FORM THE AVERAGE
                                                MOV     (R4)+,-(SP)
                                                MOV     (R4)+,-(SP)
                                                JSR     PC,$DIV
                                                ROL     (SP)+           ;IS THE REMAINDER OVER HALF?
                                                BPL     5$              ;NO--BRANCH
                                                INC     (SP)            ;YES--ROUND UP
                                5$:
                                                JSR     PC,$SB2D        ;CHANGE TO DECIMAL ASCIZ
                                                JSR     PC,$SUPRS       ;TYPE WITHOUT LEADING ZEROS
                                                TYPE    ,MSGOUS
                                                TYPE    ,BLNKS2         ;TYPE 2 SPACES
                                                MOV     -2(R4),-(SP)    ;PUT -2(R4) ON THE STACK
                                                JSR     PC,$SB2D        ;CHANGE TO DECIMAL ASCIZ
                                                JSR     PC,$SUPRS       ;TYPE WITHOUT LEADING ZEROS
                        001116                  CMP     #12,$TSTNM      ;TEST 12 ?
                                                BEQ     6$              ;BRANCH IF SO
                        001116                  CMP     #14,$TSTNM      ;TEST 14 ?
                                                BEQ     6$              ;BRANCH IF SO
                                                TYPE    ,MSGNUM         ;TYPE "SEEKS TIMED"
                                                BR      7$              ;TYPE IT
                                6$:             TYPE    ,MSGSEA         ;TYPE "SEARCH TIME"
                                7$:
                                                MOV     R5,2$           ;NEXT MESSAGE POINTER
                                                BEQ     9$              ;IF NONE EXIT
                                                CLR     R5              ;NO MORE THAN 2
                                                BR      1$
                                8$:             TYPE    ,MSGNON
                                9$:             RTS     R0              ;EXIT

                                ;THIS SUBROUTINE WILL INCREMENT THE TRACK
                                ;NUMBER (R2) BY THE AMOUNT SPECIFIED BY 'IT'.
                                ;CALL
                                ;               JSR     R0,INCTRK
                                ;               RETURN1                 ;TRACK NUMBER GREATER THAN LT15
                                ;               RETURN2                 ;TRACK NUMBER INCREMENTED

                                INCTRK: CMP     R2,LT           ;LAST TRACK COMPLETED?
                                                BEQ     2$              ;YES--EXIT
```

```
 865 032160  063702  002340                ADD     IT,R2           ;NO--UPDATE TRACK
 866 032164  020237  002336                CMP     R2,LT           ;TRACK TO BIG?
 867 032170  003402                        BLE     1$              ;NO--EXIT
 868 032172  013702  002336                MOV     LT,R2           ;YES--SET TRACK TO LAST TRACK
 869 032176  005720                1$:     TST     (R0)+           ;ADJUST FOR RETURN 2
 870 032200  000200                2$:     RTS     R0              ;RETURN
 871
 872
 873                                ;THIS SUBROUTINE WILL INCREMENT THE CYLINDER
 874                                ;NUMBER (R1) BY THE AMOUNT SPECIFIED BY 'IC'.
 875                                ;CALL
 876                                ;       JSR     R0,INCCYL
 877                                ;       RETURN1                 ;CYLINDER NUMBER GREATER THAN LC15
 878                                ;       RETURN2                 ;CYLINDER NUMBER INCREMENTED
 879
 880 032202  020137  002330         INCCYL: CMP     R1,LC           ;LAST CYLINDER COMPLETED?
 881 032206  001410                        BEQ     2$              ;YES--EXIT
 882 032210  063701  002332                ADD     IC,R1           ;NO--UPDATE CYLINDER
 883 032214  020137  002330                CMP     R1,LC           ;CYLINDER TO BIG?
 884 032220  003402                        BLE     1$              ;NO--EXIT
 885 032222  013701  002330                MOV     LC,R1           ;YES--SET CYLINDER TO LAST CYLINDER
 886 032226  005720                1$:     TST     (R0)+           ;ADJUST FOR RETURN 2
 887 032230  000200                2$:     RTS     R0              ;RETURN
 888
 889                                ;THIS ROUTINE DECREMENTS THE SECTOR ADDRESS.
 890                                ;CALL
 891                                ;       CLR     -(SP)           ;CLEAR THE STACK
 892                                ;       JSR     PC,DECSEC       ;SUBROUTINE ENTRY
 893                                ;       RETURN
 894
 895 032232  113766  045564  000002 DECSEC: MOVB    RM.REG+RMDA,2(SP)       ;PUT THE SECTOR ADDRESS ON THE STACK
 896 032240  005366  000002                DEC     2(SP)           ;DECREMENT THE ADDRESS
 897 032244  100003                        BPL     1$              ;BR IF NOT CORRECTION NEEDED
 898 032246  013766  002456  000002        MOV     PRMLMT+24,2(SP) ;OVERFLOW OCCURED, FORCE TO MAXIMUM ADDRESS
 899 032254  000207                1$:     RTS     PC              ;RETURN
 900
 901                                ;THIS SUBROUTINE IS USED TO FILL THE DATA BUFFER
 902                                ;WITH ADDRESSES FROM 0 TO 31 WITH EACH ADDRESS
 903                                ;BEING STORED IN 256 CONSECUTIVE LOCATIONS
 904                                ;CALL
 905                                ;       JSR     PC,FILBUF
 906                                ;       RETURN
 907
 908 032256  104412                 FILBUF: SAVREG                  ;SAVE R0 - R5
 909 032260  005000                        CLR     R0              ;FIRST DISK ADDRESS
 910 032262  012701  052776                MOV     #BUFFER,R1      ;START FILLING HERE
 911 032266  012702  000400         1$:    MOV     #256.,R2        ;DO 256 WORDS
 912 032272  010021                 2$:    MOV     R0,(R1)+        ;STORE
 913 032274  005302                        DEC     R2              ;MORE?
 914 032276  003375                        BGT     2$              ;YES--BRANCH
 915 032300  005200                        INC     R0              ;NO--UPDATE DISK ADDRESS
 916 032302  023700  002456                CMP     PRMLMT+24,R0    ;DONE?
 917 032306  103367                        BHIS    1$              ;NO--BRANCH
 918 032310  104413                        RESREG                  ;RESTORE R0 - R5
 919 032312  000207                        RTS     PC              ;RETURN
 920
 921                                ;THIS ROUTINE WILL CLEAR THE BUFFER BY
```

```
                                ;SETTING EACH WORD TO "177400".
                                ;CALL
                                ;       JSR     R0,CLRBUF
                                ;       RETURN

032314  104412                  CLRBUF: SAVREG                          ;SAVE R0 - R5
032316  012701  177400                  MOV     #177400,R1              ;WORD TO FILL BUFFER WITH
032322  012702  052776                  MOV     #BUFFER,R2              ;FIRST ADDRESS OF BUFFER
032326  012703  112776                  MOV     #BUFFER+<512.*32.>,R3   ;LAST ADDRESS+2 OF BUFFER
032332  010122                  1$:     MOV     R1,(R2)+                ;FILL WORDS 1, 9,...249,...5625
032334  010122                          MOV     R1,(R2)+                ;FILL WORDS 2,10,...250,...5626
032336  010122                          MOV     R1,(R2)+                ;FILL WORDS 3,11,...251,...5627
032340  010122                          MOV     R1,(R2)+                ;FILL WORDS 4,12,...252,...5628
032342  010122                          MOV     R1,(R2)+                ;FILL WORDS 5,13,...253,...5629
032344  010122                          MOV     R1,(R2)+                ;FILL WORDS 6,14,...254,...5630
032346  010122                          MOV     R1,(R2)+                ;FILL WORDS 7,15,...255,...5631
032350  010122                          MOV     R1,(R2)+                ;FILL WORDS 8,16,...256,...5632
032352  020203                          CMP     R2,R3                   ;DONE?
032354  103766                          BLO     1$                      ;NO--BRANCH
032356  104413                          RESREG                          ;RESTORE R0 - R5
032360  000200                          RTS     R0                      ;RETURN FROM CALL

                                ;THIS ROUTINE IS USED TO CHECK THE DATA BUFFER
                                ;FOR ADDRESSES 0 THROUGH 31 WITH EACH ADDRESS
                                ;BEING STORED IN 256 CONSECUTIVE LOCATIONS
                                ;CALL
                                ;       JSR     R0,CKSCTR
                                ;       RETURN

032362  104412                  CKSCTR: SAVREG                          ;SAVE R0 - R5
032364  162706  000004                  SUB     #4,SP                   ;RESERVE TEMP. STORAGE AREA
032370  005001                          CLR     R1                      ;FIRST SECTOR
032372  012716  052776                  MOV     #BUFFER,(SP)            ;FIRST ADDRESS OF DATA BUFFER
032376  005066  000002                  CLR     2(SP)                   ;NO ERRORS
032402  012702  000020          1$:     MOV     #16.,R2                 ;LOOP COUNT (16*16=256)
032406  011603                          MOV     (SP),R3                 ;GET 1ST ADDRESS OF THIS SECTORS DATA
032410                          2$:
032410  020123                          CMP     R1,(R3)+                ;WORD 1
032412  001063                          BNE     7$                      ;BRANCH IF BAD
032414  020123                          CMP     R1,(R3)+                ;WORD 2
032416  001061                          BNE     7$                      ;BRANCH IF BAD
032420  020123                          CMP     R1,(R3)+                ;WORD 3
032422  001057                          BNE     7$                      ;BRANCH IF BAD
032424  020123                          CMP     R1,(R3)+                ;WORD 4
032426  001055                          BNE     7$                      ;BRANCH IF BAD
032430  020123                          CMP     R1,(R3)+                ;WORD 5
032432  001053                          BNE     7$                      ;BRANCH IF BAD
032434  020123                          CMP     R1,(R3)+                ;WORD 6
032436  001051                          BNE     7$                      ;BRANCH IF BAD
032440  020123                          CMP     R1,(R3)+                ;WORD 7
032442  001047                          BNE     7$                      ;BRANCH IF BAD
032444  020123                          CMP     R1,(R3)+                ;WORD 8
032446  001045                          BNE     7$                      ;BRANCH IF BAD
032450  020123                          CMP     R1,(R3)+                ;WORD 9
032452  001043                          BNE     7$                      ;BRANCH IF BAD
032454  020123                          CMP     R1,(R3)+                ;WORD 10
032456  001041                          BNE     7$                      ;BRANCH IF BAD
```

```
                032460  020123                  CMP     R1,(R3)+        ;WORD 11
                032462  001037                  BNE     7$              ;BRANCH IF BAD
                032464  020123                  CMP     R1,(R3)+        ;WORD 12
                032466  001035                  BNE     7$              ;BRANCH IF BAD
                032470  020123                  CMP     R1,(R3)+        ;WORD 13
                032472  001033                  BNE     7$              ;BRANCH IF BAD
                032474  020123                  CMP     R1,(R3)+        ;WORD 14
                032476  001031                  BNE     7$              ;BRANCH IF BAD
                032500  020123                  CMP     R1,(R3)+        ;WORD 15
                032502  001027                  BNE     7$              ;BRANCH IF BAD
                032504  020123                  CMP     R1,(R3)+        ;WORD 16
                032506  001025                  BNE     7$              ;BRANCH IF BAD
    963         032510  005302                  DEC     R2              ;FINISHED WITH THIS SECTORS DATA?
    964         032512  001336                  BNE     2$              ;NO--BRANCH
    965         032514  062716  001000          3$:     ADD     #512.,(SP)      ;YES--FIRST ADDRESS OF NEXT SECTOR
    966         032520  005201                  INC     R1              ;MOVE TO NEXT SECTOR
    967         032522  023701  002456          CMP     PRMLMT+24,R1    ;DONE?
    968         032526  103325                  BHIS    1$              ;NO--BRANCH
    969         032530  005766  000002          4$:     TST     2(SP)           ;ERROR OCCUR?
    970         032534  001406                  BEQ     6$              ;NO--BRANCH
    971         032536  123737  001462  001117          CMPB    ERR.CT,$ERFLG   ;MAX. ERROR OCCURRED?
    972         032544  101002                  BHI     6$              ;NO--BRANCH
    973         032546  013700  001346          5$:     MOV     BYPASS,R0       ;TAKE ERROR EXIT
    974         032552  062706  000004          6$:     ADD     #4,SP           ;FREE TEMP. AREA
    975         032556  104413                  RESREG                  ;RESTORE R0 - R5
    976         032560  000200                  RTS     R0              ;RETURN FROM CALL
    977         032562  010304                  7$:     MOV     R3,R4           ;FORM WORD NUMBER AND
    978         032564  161604                  SUB     (SP),R4         ;ADDRESS TO CONTINUE FROM
    979         032566  010405                  MOV     R4,R5
    980         032570  006204                  ASR     R4              ;WORD NUMBER
    981         032572  042705  177740          BIC     #^C37,R5
    982         032576  001002                  BNE     8$              ;BRANCH IF NOT A MULTIPLE OF 16
    983         032600  012705  000040          MOV     #40,R5          ;SET TO WORD 16
    984         032604  006305                  8$:     ASL     R5
    985         032606  062705  032410          ADD     #2$,R5          ;ADDRESS
    986         032612  016337  177776  001142  MOV     -2(R3),$BDDAT   ;SAVE BAD DATA
    987         032620  005766  000002          TST     2(SP)           ;FIRST ERROR?
    988         032624  001015                  BNE     10$             ;NO--BRANCH
    989         032626  013737  045550  001364  MOV     DTADPB+12,CYL.DS        ;CYLINDER NUMBER
    990         032634  113737  045547  001370  MOVB    DTADPB+11,TRK.DS        ;TRACK NUMBER
    991         032642  012737  032652  001222  MOV     #9$,$ESCAPE     ;;ESCAPE TO 9$ ON ERROR
    992         032650  104021                  EMT     21
    993         032652  105166  000002          9$:     COMB    2(SP)           ;SET ERROR SWITCH
    994         032656  000404                  BR      11$
    995         032660                          10$:
                032660  012737  032670  001222  MOV     #11$,$ESCAPE    ;;ESCAPE TO 11$ ON ERROR
    996         032666  104022                  EMT     22
    997         032670  032777  001000  146256  11$:    BIT     #SW09,@SWR      ;LOOP ON ERROR?
    998         032676  001323                  BNE     5$              ;YES
    999         032700  032777  000002  146246  BIT     #SW01,@SWR      ;STOP DATA COMPARE?
   1000         032706  001310                  BNE     4$              ;YES--BRANCH
   1001         032710  123737  001462  001117  CMPB    ERR.CT,$ERFLG   ;MAX. ERRORS?
   1002         032716  101713                  BLOS    5$              ;YES--BRANCH
   1003         032720  032777  000040  146226  BIT     #SW05,@SWR      ;REPORT ONLY 1ST ERROR PER SECTOR?
   1004         032726  001272                  BNE     3$              ;YES--BRANCH
   1005         032730  000115                  JMP     (R5)
   1006
```

```
                                        ;THIS ROUTINE WILL MOVE THE 16 WORDS OF THE
                                        ;DESIRED PATTERN INTO THE DATA BUFFER.
                                        ;CALL
                                        ;       MOV     #NX,R0          ;PATTERN NUMBER INDEX TO R0
                                        ;       JSR     PC,SETBUF
                                        ;
032732  104412                  SETBUF: SAVREG                          ;SAVE R0 - R5
032734  012701  052776                  MOV     #BUFFER,R1              ;FIRST ADDRESS
032740  013702  045542                  MOV     DTADPB+4,R2             ;WORD COUNT
032744  016003  003520          1$:     MOV     PAT.PT(R0),R3           ;PICKUP PATTERN POINTER
032750  012321                          MOV     (R3)+,(R1)+             ;MOVE WORD 1 INTO DATA BUFFER
032752  012321                          MOV     (R3)+,(R1)+             ;MOVE WORD 2 INTO DATA BUFFER
032754  012321                          MOV     (R3)+,(R1)+             ;MOVE WORD 3 INTO DATA BUFFER
032756  012321                          MOV     (R3)+,(R1)+             ;MOVE WORD 4 INTO DATA BUFFER
032760  012321                          MOV     (R3)+,(R1)+             ;MOVE WORD 5 INTO DATA BUFFER
032762  012321                          MOV     (R3)+,(R1)+             ;MOVE WORD 6 INTO DATA BUFFER
032764  012321                          MOV     (R3)+,(R1)+             ;MOVE WORD 7 INTO DATA BUFFER
032766  012321                          MOV     (R3)+,(R1)+             ;MOVE WORD 8 INTO DATA BUFFER
032770  012321                          MOV     (R3)+,(R1)+             ;MOVE WORD 9 INTO DATA BUFFER
032772  012321                          MOV     (R3)+,(R1)+             ;MOVE WORD 10 INTO DATA BUFFER
032774  012321                          MOV     (R3)+,(R1)+             ;MOVE WORD 11 INTO DATA BUFFER
032776  012321                          MOV     (R3)+,(R1)+             ;MOVE WORD 12 INTO DATA BUFFER
033000  012321                          MOV     (R3)+,(R1)+             ;MOVE WORD 13 INTO DATA BUFFER
033002  012321                          MOV     (R3)+,(R1)+             ;MOVE WORD 14 INTO DATA BUFFER
033004  012321                          MOV     (R3)+,(R1)+             ;MOVE WORD 15 INTO DATA BUFFER
033006  012321                          MOV     (R3)+,(R1)+             ;MOVE WORD 16 INTO DATA BUFFER
033010  062702  000020                  ADD     #16.,R2 ;DONE?
033014  001353                          BNE     1$                      ;NO--BRANCH
033016  104413                          RESREG                          ;RESTORE R0 - R5
033020  000207                          RTS     PC                      ;RETURN

                                        ;THIS ROUTINE COMPARES A 16 WORD DATA PATTERN
                                        ;AGAINST THE DATA BUFFER
                                        ;CALL
                                        ;       MOV     #NX,R0          ;PATTERN NUMBER INDEX TO R0
                                        ;       JSR     PC,DATCMP
                                        ;       RETURN
                                        ;
033022  104412                  DATCMP: SAVREG                          ;SAVE R0 - R5
033024  012701  052776                  MOV     #BUFFER,R1              ;FIRST ADDRESS OF BUFFER
033030  013702  045542                  MOV     DTADPB+4,R2             ;WORD COUNT
033034  005046                          CLR     -(SP)                   ;NO ERROR
033036  016003  003520          1$:     MOV     PAT.PT(R0),R3           ;PATTERN POINTER
033042                          2$:
033042  162321                          SUB     (R3)+,(R1)+             ;CHECK WORD 1
033044  001044                          BNE     4$                      ;BRANCH IF DIFFERENT
033046  162321                          SUB     (R3)+,(R1)+             ;CHECK WORD 2
033050  001042                          BNE     4$                      ;BRANCH IF DIFFERENT
033052  162321                          SUB     (R3)+,(R1)+             ;CHECK WORD 3
033054  001040                          BNE     4$                      ;BRANCH IF DIFFERENT
033056  162321                          SUB     (R3)+,(R1)+             ;CHECK WORD 4
033060  001036                          BNE     4$                      ;BRANCH IF DIFFERENT
033062  162321                          SUB     (R3)+,(R1)+             ;CHECK WORD 5
033064  001034                          BNE     4$                      ;BRANCH IF DIFFERENT
033066  162321                          SUB     (R3)+,(R1)+             ;CHECK WORD 6
033070  001032                          BNE     4$                      ;BRANCH IF DIFFERENT
033072  162321                          SUB     (R3)+,(R1)+             ;CHECK WORD 7
```

```
                033074  001030                          BNE     4$              ;BRANCH IF DIFFERENT
                033076  162321                          SUB     (R3)+,(R1)+     ;CHECK WORD 8
                033100  001026                          BNE     4$              ;BRANCH IF DIFFERENT
                033102  162321                          SUB     (R3)+,(R1)+     ;CHECK WORD 9
                033104  001024                          BNE     4$              ;BRANCH IF DIFFERENT
                033106  162321                          SUB     (R3)+,(R1)+     ;CHECK WORD 10
                033110  001022                          BNE     4$              ;BRANCH IF DIFFERENT
                033112  162321                          SUB     (R3)+,(R1)+     ;CHECK WORD 11
                033114  001020                          BNE     4$              ;BRANCH IF DIFFERENT
                033116  162321                          SUB     (R3)+,(R1)+     ;CHECK WORD 12
                033120  001016                          BNE     4$              ;BRANCH IF DIFFERENT
                033122  162321                          SUB     (R3)+,(R1)+     ;CHECK WORD 13
                033124  001014                          BNE     4$              ;BRANCH IF DIFFERENT
                033126  162321                          SUB     (R3)+,(R1)+     ;CHECK WORD 14
                033130  001012                          BNE     4$              ;BRANCH IF DIFFERENT
                033132  162321                          SUB     (R3)+,(R1)+     ;CHECK WORD 15
                033134  001010                          BNE     4$              ;BRANCH IF DIFFERENT
                033136  162321                          SUB     (R3)+,(R1)+     ;CHECK WORD 16
                033140  001006                          BNE     4$              ;BRANCH IF DIFFERENT
   1041         033142  062702  000020                  ADD     #16.,R2         ;DONE ?
   1042         033146  001333                          BNE     1$              ;NO--BRANCH
   1043         033150  005726                  3$:     TST     (SP)+           ;YES -- CLEAN UP STACK
   1044         033152  104413                          RESREG                  ;RESTORE R0 - R5
   1045         033154  000207                          RTS     PC
   1046         033156  010104                  4$:     MOV     R1,R4           ;FORM THE WORD NUMBER
   1047         033160  162704  052776                  SUB     #BUFFER,R4
   1048         033164  006204                          ASR     R4              ;WORD NUMBER
   1049         033166  010305                          MOV     R3,R5           ;FORM ADDRESS TO CONTINUE FROM
   1050         033170  166005  003520                  SUB     PAT.PT(R0),R5
   1051         033174  006305                          ASL     R5
   1052         033176  062705  033042                  ADD     #2$,R5          ;ADDRESS
   1053         033202  064341                          ADD     -(R3),-(R1)     ;RECONSTRUCT THE BAD WORD
   1054         033204  010137  001136                  MOV     R1,$BDADR       ;SAVE THE ERROR INFORMATION
   1055         033210  010337  001134                  MOV     R3,$GDADR       ;
   1056         033214  012137  001142                  MOV     (R1)+,$BDDAT    ;
   1057         033220  012337  001140                  MOV     (R3)+,$GDDAT    ;
   1058         033224  005716                          TST     (SP)            ;1ST DATA COMPARE ERROR?
   1059         033226  001023                          BNE     6$              ;NO--BRANCH
   1060         033230  013737  045550  001364          MOV     DTADPB+12,CYL.DS        ;CYLINDER
   1061         033236  113737  045547  001370          MOVB    DTADPB+11,TRK.DS        ;TRACK
   1062         033244  113737  045546  001366          MOVB    DTADPB+10,SEC.DS        ;SECTOR
   1063         033252  016600  000016                  MOV     16(SP),R0       ;GET TEST PC+4
   1064         033256  012737  033266  001222          MOV     #5$,$ESCAPE     ;;ESCAPE TO 5$ ON ERROR
   1065         033264  104013                          EMT     13
   1066         033266  016600  000014          5$:     MOV     14(SP),R0       ;PATTERN NUMBER INDEX
   1067         033272  105116                          COMB    (SP)            ;SET THE ERROR SWITCH
   1068         033274  000404                          BR      7$
   1069         033276                          6$:
                033276  012737  033306  001222          MOV     #7$,$ESCAPE     ;;ESCAPE TO 7$ ON ERROR
   1070         033304  104014                          EMT     14
   1071         033306  032777  000002  145640  7$:     BIT     #SW01,@SWR      ;STOP DATA COMPARE?
   1072         033314  001315                          BNE     3$              ;YES--EXIT
   1073         033316  123737  001462  001117          CMPB    ERR.CT,$ERFLG   ;MAX. ERRORS?
   1074         033324  101004                          BHI     8$              ;NO--BRANCH
   1075         033326  013766  001346  000016          MOV     BYPASS,16(SP)   ;YES--ERROR EXIT
   1076         033334  000705                          BR      3$
   1077         033336  000115                  8$:     JMP     (R5)            ;NO--CONTINUE AT NEXT WORD
```

```
                                        ;THIS ROUTINE WILL FILL THE DATA BUFFER (256*22 WORDS) WITH
                                        ;A RANDOM PATTERN.  THE FIRST TWO WORDS OF EVERY 256 WILL
                                        ;BE THE BASE OF THE RANDOM NUMBER GENERATOR FOR THE
                                        ;NEXT 254 WORDS.
                                        ;NOTE: THIS ROUTINE DESTROYS R1 AND R2
                                        ;CALL
                                        ;       JSR     R0,FILRAN
                                        ;       RETURN

033340  012701  052776          FILRAN: MOV     #BUFFER,R1
033344  013702  002456                  MOV     PRMLMT+24,R2    ;MAXIMUM NUMBER OF SECTORS
033350  004037  033560          1$:     JSR     R0,RANPAT
033354  005302                          DEC     R2
033356  100374                          BPL     1$
033360  000200                          RTS     R0

                                        ;THIS ROUTINE USES THE FIRST TWO WORDS OF THE
                                        ;READ BUFFER TO GENERATED A RANDOM PATTERN.  THEN
                                        ;THE READ BUFFER IS COMPARED TO THE PATTERN GENERATED.
                                        ;NOTE: THIS ROUTINE DESTROYS R1-R4
                                        ;CALL
                                        ;       JSR     R0,RANCK
                                        ;       RETURN

033362  013746  025646          RANCK:  MOV     $HINUM,-(SP)    ;SAVE THE PRESENT RANDOM NUMBER
033366  013746  025650                  MOV     $LONUM,-(SP)
033372  012702  053776                  MOV     #BUFFER+512.,R2 ;READ BUFFER ADDRESS
033376  012701  054776                  MOV     #BUFFER+1024.,R1        ;RANDOM PATTERN ADDRESS
033402  010103                          MOV     R1,R3           ;COPY IT INTO R3 FOR LATER USE
033404  011237  025650                  MOV     (R2),$LONUM     ;PRIME THE RANDOM NUMBER GENERATOR
033410  016237  000002  025646          MOV     2(R2),$HINUM
033416  004037  033560                  JSR     R0,RANPAT       ;GENERATE A RANDOM PATTERN
033422  012637  025650                  MOV     (SP)+,$LONUM    ;RESTORE PRESENT RANDOM NUMBER
033426  012637  025646                  MOV     (SP)+,$HINUM
033432  005046                          CLR     -(SP)           ;NO ERRORS
033434  162322                  1$:     SUB     (R3)+,(R2)+     ;ARE THESE TWO WORDS DIFFERENT?
033436  001441                          BEQ     4$              ;NO--BRANCH
033440  012737  033512  001222          MOV     #3$,$ESCAPE     ;;ESCAPE TO 3$ ON ERROR
033446  064342                          ADD     -(R3),-(R2)     ;RECREATE THE BAD WORD
033450  010237  001136                  MOV     R2,$BDADR       ;ADDRESS OF BAD DATA
033454  010337  001134                  MOV     R3,$GDADR       ;ADDRESS OF GOOD DATA
033460  012237  001142                  MOV     (R2)+,$BDDAT    ;BAD DATA
033464  012337  001140                  MOV     (R3)+,$GDDAT    ;GOOD DATA
033470  010204                          MOV     R2,R4           ;FORM WORD NUMBER (1 TO 256)
033472  162704  053776                  SUB     #BUFFER+512.,R4
033476  006204                          ASR     R4
033500  005716                          TST     (SP)            ;FIRST ERROR
033502  001002                          BNE     2$              ;NO--BRANCH
033504  105116                          COMB    (SP)            ;YES--SET ERROR SWITCH
033506  104015                          EMT     15
033510                          2$:
033510  104016                          EMT     16
033512  032777  001000  145434  3$:     BIT     #SW09,@SWR      ;LOOP ON ERROR?
033520  001012                          BNE     5$              ;YES--BRANCH
033522  123737  001462  001117          CMPB    ERR.CT,$ERFLG   ;MAX. ERRORS OCCURRED?
033530  101406                          BLOS    5$              ;YES--BRANCH
```

```
1134 033532  032777  000002  145414          BIT     #SW01,@SWR      ;STOP COMPARING?
1135 033540  001002                          BNE     5$              ;YES--BRANCH
1136 033542  020103                  4$:     CMP     R1,R3           ;ALL DATA BEEN COMPARED?
1137 033544  101333                          BHI     1$              ;NO--BRANCH
1138 033546  005726                  5$:     TST     (SP)+           ;ERROR OCCUR?
1139 033550  001402                          BEQ     6$              ;NO--BRANCH
1140 033552  013700  001346                  MOV     BYPASS,R0       ;TAKE ERROR EXIT
1141 033556  000200                  6$:     RTS     R0              ;EXIT
1142
1143                                  ;THIS ROUTINE FILLS A 256 WORD BUFFER WITH A RANDOM
1144                                  ;PATTERN OF WHICH THE FIRST TWO WORDS ARE THE BASE
1145                                  ;OF THE PATTERN.
1146                                  ;CALL
1147                                  ;       MOV     #ADR,R1         ;ADDRESS OF THE BUFFER
1148                                  ;       JSR     R0,RANPAT
1149                                  ;       RETURN
1150
1151 033560  010246                  RANPAT: MOV     R2,-(SP)        ;SAVE R2
1152 033562  012702  000200                  MOV     #256./2.,R2     ;GENERATE 256 WORDS
1153 033566  000402                          BR      2$
1154 033570  004737  025550          1$:     JSR     PC,$RAND        ;GENERATE A RANDOM NUMBER
1155 033574  013721  025650          2$:     MOV     $LONUM,(R1)+    ;PUT LOW WORD IN BUFFER
1156 033600  013721  025646                  MOV     $HINUM,(R1)+    ;PUT HIGH WORD IN BUFFER
1157 033604  005302                          DEC     R2              ;DONE?
1158 033606  003370                          BGT     1$              ;NO--BRANCH
1159 033610  012602                          MOV     (SP)+,R2        ;RESTORE R2
1160 033612  000200                          RTS     R0              ;EXIT
1161
1162                                  ;THIS ROUTINE GENERATES RANDOM CYLINDER, TRACK, AND SECTOR
1163                                  ;ADDRESSES AND SAVES THEN IN THE DPB (DTADPB+10, 11 & DTADPB+12).
1164                                  ;NOTE: THIS ROUTINE DESTROYS R1-R3
1165                                  ;CALL
1166                                  ;       JSR     R0,RANADR
1167                                  ;       RETURN
1168
1169 033614  004737  025550          RANADR: JSR     PC,$RAND        ;GENERATE A RANDOM NUMBER
1170 033620  113701  025650                  MOVB    $LONUM,R1       ;FORM SECTOR IN R1
1171 033624  113702  025651                  MOVB    $LONUM+1,R2     ;FORM TRACK IN R2
1172 033630  013703  025646                  MOV     $HINUM,R3       ;FORM CYLINDER IN R3
1173 033634  105701                          TSTB    R1              ;ENSURE THE SECTOR IS BETWEEN 0 AND 31
1174 033636  002403                          BLT     2$
1175 033640  123701  002456          1$:     CMPB    PRMLMT+24,R1    ;CHECK MAXIMUM SECTOR ADDRESS
1176 033644  103003                          BHIS    3$
1177 033646  000241                  2$:     CLC
1178 033650  106001                          RORB    R1
1179 033652  000772                          BR      1$
1180 033654  105702                  3$:     TSTB    R2              ;ENSURE THE TRACK IS BETWEEN 0 AND LAST TRACK
1181 033656  002403                          BLT     5$
1182 033660  123702  001372          4$:     CMPB    LSTRK,R2
1183 033664  002003                          BGE     6$
1184 033666  000241                  5$:     CLC
1185 033670  106002                          RORB    R2
1186 033672  000772                          BR      4$
1187 033674  023703  002326          6$:     CMP     FC,R3           ;ENSURE THE CYLINDER IS BETWEEN FC AND LC
1188 033700  003413                          BLE     7$
1189 033702  000241                          CLC
1190 033704  006003                          ROR     R3
```

```
 1191 033706  005503                        ADC     R3
 1192 033710  001371                        BNE     6$
 1193 033712  010103                        MOV     R1,R3
 1194 033714  000303                        SWAB    R3
 1195 033716  060203                        ADD     R2,R3
 1196 033720  005203                        INC     R3
 1197 033722  003364                        BGT     6$
 1198 033724  005403                        NEG     R3
 1199 033726  000762                        BR      6$
 1200 033730  023703  002330          7$:   CMP     LC,R3
 1201 033734  002003                        BGE     8$
 1202 033736  000241                        CLC
 1203 033740  006003                        ROR     R3
 1204 033742  000772                        BR      7$
 1205 033744  023703  002326          8$:   CMP     FC,R3
 1206 033750  003403                        BLE     9$
 1207 033752  005203                        INC     R3
 1208 033754  000303                        SWAB    R3
 1209 033756  000764                        BR      7$
 1210 033760  110137  045546          9$:   MOVB    R1,DTADPB+10    ;SAVE SECTOR ADDRESS
 1211 033764  110237  045547                MOVB    R2,DTADPB+11    ;SAVE TRACK ADDRESS
 1212 033770  010337  045550                MOV     R3,DTADPB+12    ;SAVE CYLINDER ADDRESS
 1213 033774  000200                        RTS     R0              ;RETURN
 1214
 1215                               ;THIS ROUTINE IS USED TO INPUT THE "CONTROL SWITCHES".
 1216                               ;IF SWR<07>=1 THE PRESENT SETTING WILL BE TYPED AND THE NEW
 1217                               ;SETTING IS READ AND STORED.
 1218                               ;NOTE: THIS ROUTINE DESTROYS R3 AND R4
 1219                               ;CALL
 1220                               ;       JSR     PC,GETSWR
 1221                               ;       RETURN                  ;(C.SWR)=DESIRED CONTROL SWITCHES
 1222
 1223 033776  032777  000200  145150 GETSWR: BIT     #SW07,@SWR      ;READ CONTROL SWITCHES?
 1224 034004  001430                        BEQ     2$              ;NO--BRANCH
 1225 034006  104401  034014                TYPE    ,65$            ;;TYPE ASCIZ STRING
      034012  000410                        BR      64$             ;;GET OVER THE ASCIZ
                                   ;;65$:   .ASCIZ  <CRLF>/SET SWR<07>=0/
      034034                       64$:
 1226 034034  012703  045702          1$:   MOV     #MSG.CS,R3      ;"CONTROL SWITCHES="
 1227 034040  013704  001314                MOV     C.SWR,R4        ;PRESENT CONTROL SWITCH SETTINGS
 1228 034044  004037  034070                JSR     R0,GETNUM       ;GET THE NEW SWITCH SETTINGS
 1229 034050  000771                        BR      1$              ; COMMA
 1230 034052  000240                        NOP                     ;PERIOD
 1231 034054  013737  001314  001316        MOV     C.SWR,SAVCSW    ;SAVE PREVIOUS VALUE
 1232 034062  010437  001314                MOV     R4,C.SWR        ; DOUBLE PERIOD-SAVE NEW SWITCH SETTING
 1233 034066  000207                  2$:   RTS     PC              ;RETURN FROM CALL
 1234
 1235                               ;THIS ROUTINE WILL TYPE AN ASCIZ MESSAGE AND THEN
 1236                               ;INPUT AN ASCIZ STRING AND CHANGE THE STRING TO OCTAL
 1237                               ;IF REQUIRED.
 1238                               ;NOTE: THIS ROUTINE DESTROYS R1
 1239                               ;CALL
 1240                               ;       MOV     #ADR,R3         ;ADDRESS OF ASCIZ MESSAGE
 1241                               ;       MOV     #NUM,R4         ;OCTAL NUMBER
 1242                               ;       JSR     R0,GETNUM
 1243                               ;       RETURN1                 ;INPUT TERMINATED WITH A COMMA
 1244                               ;       RETURN2                 ;WITH A PERIOD
```

```
                                        ;       RETURN3                 ;WITH A DOUBLE PERIOD
                                        ;                               ;R4=INPUT NUMBER AND
                                        ;                               ;R2=R4*32 FOR ALL
                                        ;                               ;THREE RETURNS
                                        ;
034070  010337  034076                  GETNUM: MOV     R3,2$           ;SAVE MESSAGE POINTER
034074  104401                          1$:     TYPE                    ;TYPE THE MESSAGE
034076  000000                          2$:     .WORD   0               ;MESSAGE POINTER GOES HERE
034100  010446                                  MOV     R4,-(SP)        ;;SAVE R4 FOR TYPEOUT
034102  104402                                  TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
034104  104401  045725                          TYPE    ,SLASH          ;' /'
034110  104411                                  RDLIN                   ;READ AN ASCIZ STRING
034112  012601                                  MOV     (SP)+,R1        ;ADDRESS OF FIRST CHARACTER
034114  004037  036340                          JSR     R0,CK.CHR       ;CHECK ONE CHARACTER
034120  034074                                  1$                      ;ILLEGAL CHARACTER
034122  034074                                  1$                      ;CARRIAGE RETURN
034124  034136                                  4$                      ;"/"
034126  034162                                  8$                      ;":"
034130  034170                                  9$                      ;"."
034132  034134                                  3$                      ;DIGIT 0-9
034134  005301                          3$:     DEC     R1              ;DECREMENT THE INPUT POINTER
034136                                  4$:
034136  004037  036600                          JSR     R0,CK.NUM       ;CHECK THE NUMBER
034142  034074                                  1$                      ;ILLEGAL INPUT
034144  034156                                  7$                      ;TERMINATED WITH A "," OR "CR"
034146  034154                                  6$                      ;TERMINATED WITH A "."
034150  034152                                  5$                      ;TERMINATED WITH A ".."
034152  005720                          5$:     TST     (R0)+           ;DOUBLE PERIOD
034154  005720                          6$:     TST     (R0)+           ;SINGLE PERIOD
034156  010204                          7$:     MOV     R2,R4           ;COMMA--SAVE INPUT NUMBER
034160  000414                                  BR      11$             ;GO TO EXIT
034162  105711                          8$:     TSTB    (R1)            ;TERMINATOR AFTER A COMMA?
034164  001343                                  BNE     1$              ;NO--LOOP
034166  000411                                  BR      11$             ;YES--EXIT
034170  105711                          9$:     TSTB    (R1)            ;TERMINATOR AFTER A PERIOD?
034172  001406                                  BEQ     10$             ;YES--EXIT
034174  122721  000056                          CMPB    #'.,(R1)+       ;NO--DOUBLE PERIOD?
034200  001335                                  BNE     1$              ;NO--LOOP
034202  105711                                  TSTB    (R1)            ;YES--TERMINATOR?
034204  001333                                  BNE     1$              ;NO--LOOP
034206  005720                                  TST     (R0)+           ;DOUBLE PERIOD
034210  005720                          10$:    TST     (R0)+           ;PERIOD
034212  010402                          11$:    MOV     R4,R2           ;COMMA--POSITION THE
034214  000302                                  SWAB    R2              ;NUMBER IN CASE IT
034216  006202                                  ASR     R2              ;IS THE PRIORITY LEVEL
034220  006202                                  ASR     R2
034222  006202                                  ASR     R2
034224  000200                                  RTS     R0              ;EXIT

                                        ;THIS ROUTINE IS USED TO CHANGE OR MODIFY
                                        ;THE TEST PARAMETERS.  IT GIVES THE OPERATOR
                                        ;THE CAPABILITY OF SPECIFYING WHICH DRIVES TO TEST, WHICH
                                        ;TESTS TO RUN AND HOW MANY TIMES TO
                                        ;REPEAT EACH TEST

034226  104412                          GT.PRM: SAVREG                  ;SAVE R0 - R5
034230  005037  001326                  GT.PR1: CLR     DRVSEL          ;NO DRIVE SELECTED
```

```
 1290 034234  104401  034242                 TYPE    ,65$            ;;TYPE ASCIZ STRING
      034240  000406                         BR      64$             ;;GET OVER THE ASCIZ
                                     ;;65$:  .ASCIZ  <CRLF>/DRIVE(S)=/
      034256                         64$:
 1291 034256  104411                         RDLIN                   ;READ TTY
 1292 034260  012601                         MOV     (SP)+,R1        ;ADDRESS OF ASCIZ STRING
 1293 034262  004037  036340                 JSR     R0,CK.CHR       ;CHECK ONE CHARACTER
      034266  034230                         GT.PR1                  ;ILLEGAL CHARACTER
      034270  034230                         GT.PR1                  ;CARRIAGE RETURN
      034272  034230                         GT.PR1                  ;"/"
      034274  034230                         GT.PR1                  ;","
      034276  034230                         GT.PR1                  ;"."
      034300  034302                         1$                      ;DIGIT 0-9
 1294 034302  005301                 1$:     DEC     R1
 1295 034304                         2$:
      034304  012702  000007                 MOV     #7,R2           ;UPPER LIMIT OF INPUT
      034310  004037  036414                 JSR     R0,CK.DIG       ;CHECK THE DIGIT(S)
      034314  034230                         GT.PR1                  ;ILLEGAL INPUT
      034316  034230                         GT.PR1                  ;INPUT TO LARGE
      034320  034326                         3$                      ;TERMINATED WITH A "," OR "CR"
      034322  034352                         4$                      ;TERMINATED WITH A ":"
      034324  034352                         4$                      ;TERMINATED WITH A "."
 1296 034326  156237  037054  001326 3$:     BISB    ATABIT(R2),DRVSEL       ;SET THE DRIVE SELECTED BIT
 1297 034334  105741                         TSTB    -(R1)           ;WAS THE LINE TERMINATED?
 1298 034336  001362                         BNE     2$              ;NO-GET THE NEXT DRIVE
 1299 034340  005037  001330                 CLR     TSTNMS          ;DESELECT ALL TESTS
 1300 034344  005037  001332                 CLR     TSTNMS+2
 1301 034350  000405                         BR      GTTST1          ;YES--SELECT TEST
 1302 034352  156237  037054  001326 4$:     BISB    ATABIT(R2),DRVSEL       ;SET THE SELECTED DRIVE BITS
 1303 034360  104413                 GT.PR2: RESREG                  ;RESTORE R0 - R5
 1304 034362  000207                         RTS     PC              ;EXIT
 1305
 1306 034364                         GTTST1:
      034364  104401  034372                 TYPE    ,65$            ;;TYPE ASCIZ STRING
      034370  000403                         BR      64$             ;;GET OVER THE ASCIZ
                                     ;;65$:  .ASCIZ  /TEST=/
      034400                         64$:
 1307 034400  104411                         RDLIN                   ;READ AN ASCIZ STRING
 1308 034402  012601                         MOV     (SP)+,R1        ;POINTER TO R1
 1309 034404  122711  000056                 CMPB    #'.,(R1)        ;DOUBLE PERIOD?
 1310 034410  001007                         BNE     1$              ;NO--BRANCH
 1311 034412  122761  000056  000001         CMPB    #'.,1(R1)
 1312 034420  001003                         BNE     1$
 1313 034422  105761  000002                 TSTB    2(R1)           ;"CR"?
 1314 034426  001754                         BEQ     GT.PR2          ;YES--EXIT
 1315 034430  005037  001330         1$:     CLR     TSTNMS          ;NO TEST SELECTED
 1316 034434  005037  001332                 CLR     TSTNMS+2
 1317 034440  005037  001334                 CLR     OPNFLG          ;NO TESTS TO BE OPENED
 1318 034444  005037  001336                 CLR     OPNFLG+2
 1319 034450  121127  000123         GTTST2: CMPB    (R1),#'S        ;ALL SEEK TESTS?
 1320 034454  001004                         BNE     1$              ;NO--BRANCH
 1321 034456  052737  000777  001330         BIS     #777,TSTNMS     ;YES--SELECT TESTS 0-10
 1322 034464  000552                         BR      GTTST3
 1323 034466  121127  000124         1$:     CMPB    (R1),#'T        ;ALL TIMING TESTS?
 1324 034472  001004                         BNE     2$              ;NO--BRANCH
 1325 034474  052737  036000  001330         BIS     #36000,TSTNMS   ;YES--SELECT TESTS 12-15
 1326 034502  000543                         BR      GTTST3
```

```
1327 034504  121127  000101          2$:   CMPB    (R1),#'A          ;ALL ADDRESSING TESTS?
1328 034510  001004                        BNE     3$                ;NO--BRANCH
1329 034512  052737  140000  001330        BIS     #140000,TSTNMS    ;YES--SELECT TESTS 16 & 17
1330 034520  000534                        BR      GTTST3
1331 034522  121127  000104          3$:   CMPB    (R1),#'D          ;DATA TEST?
1332 034526  001004                        BNE     4$                ;NO--BRANCH
1333 034530  052737  000001  001332        BIS     #1,TSTNMS+2       ;YES--SELECT TEST 20
1334 034536  000525                        BR      GTTST3
1335 034540  121127  000105          4$:   CMPB    (R1),#'E          ;EXERCISER TEST?
1336 034544  001004                        BNE     5$                ;NO--BRANCH
1337 034546  052737  000002  001332        BIS     #2,TSTNMS+2       ;YES--SELECT TEST 21
1338 034554  000516                        BR      GTTST3
1339 034556  004037  036264          5$:   JSR     R0,CK.OCT         ;OCTAL DIGIT?
1340 034562  000514                        BR      GTTST4            ;NO--BRANCH
1341 034564  010205                        MOV     R2,R5             ;YES--SAVE IT
1342 034566  005201                        INC     R1                ;MOVE TO NEXT CHARACTER
1343 034570  004037  036264                JSR     R0,CK.OCT         ;OCTAL DIGIT
1344 034574  000405                        BR      6$                ;NO--BRANCH
1345 034576  005201                        INC     R1                ;MOVE TO NEXT CHARACTER
1346 034600  006305                        ASL     R5                ;SCALE HIGH DIGIT
1347 034602  006305                        ASL     R5
1348 034604  006305                        ASL     R5
1349 034606  060502                        ADD     R5,R2             ;COMBINE HIGH & LOW DIGITS
1350 034610                        6$:
1353 034610  020227  000022                CMP     R2,#22            ;VALID TEST NUMBER?
1354 034614  003263                        BGT     GTTST1            ;NO--BRANCH
1355 034616  010237  035010                MOV     R2,9$             ;SAVE THE TEST NUMBER
1356 034622  010204                        MOV     R2,R4             ;CONVERT TEST NUMBER INTO AN INDEX
1357 034624  042704  000017                BIC     #17,R4            ;CLEAR UNWANTED BITS
1358 034630  006204                        ASR     R4                ;SHIFT THE BITS
1359 034632  006204                        ASR     R4
1360 034634  006204                        ASR     R4
1361 034636  006302                        ASL     R2
1362 034640  056264  001526 . 001330      BIS     BITS(R2),TSTNMS(R4)     ;SELECT TEST
1363 034646  121127  000055                CMPB    (R1),#'-          ;TEST STRING?
1364 034652  001060                        BNE     GTTST4            ;NO--BRANCH
1365 034654  005201                        INC     R1                ;YES--MOVE TO NEXT CHARACTER
1366 034656  004037  036264                JSR     R0,CK.OCT         ;OCTAL DIGIT?
1367 034662  000640                        BR      GTTST1            ;NO--BRANCH
1368 034664  010205                        MOV     R2,R5             ;YES--SAVE IT
1369 034666  005201                        INC     R1                ;MOVE TO NEXT CHARACTER
1370 034670  004037  036264                JSR     R0,CK.OCT         ;OCTAL DIGIT?
1371 034674  000405                        BR      7$                ;NO--BRANCH
1372 034676  005201                        INC     R1                ;YES--MOVE TO NEXT CHARACTER
1373 034700  006305                        ASL     R5                ;SCALE HIGH DIGIT
1374 034702  006305                        ASL     R5
1375 034704  006305                        ASL     R5
1376 034706  060502                        ADD     R5,R2             ;COMBINE HIGH & LOW DIGIT
1377 034710                        7$:
1380 034710  020227  000022                CMP     R2,#22            ;VALID TEST NUMBER?
1381 034714  003223                        BGT     GTTST1            ;NO--BRANCH
1382 034716  023702  035010                CMP     9$,R2             ;IS THE FIRST NUMBER OF THE
1383                                                                  ;STRING SMALLER THAN THE LAST?
1384 034722  002220                        BGE     GTTST1            ;NO--BRANCH
1385 034724  010246                        MOV     R2,-(SP)          ;SAVE ENDING TEST NUMBER
1386 034726  013702  035010                MOV     9$,R2             ;GET STARTING TEST NUMBER
1387 034732  012637  035010                MOV     (SP)+,9$          ;STORE ENDING TEST NUMBER
```

```
1388 034736  006337  035010                ASL     9$                ;SHIFT ENDING TEST NUMBER
1389 034742  006302                        ASL     R2                ;SHIFT TEST NUMBER
1390 034744  010204              8$:       MOV     R2,R4             ;COPY TEST NUMBER INTO R4
1391 034746  042704  000037                BIC     #37,R4            ;CLEAR LOWER BITS
1392 034752  006204                        ASR     R4                ;SHIFT THE TEST NUMBER
1393 034754  006204                        ASR     R4
1394 034756  006204                        ASR     R4
1395 034760  006204                        ASR     R4
1396 034762  056264  001526  001330        BIS     BITS(R2),TSTNMS(R4)     ;SELECT THE TEST
1397 034770  062702  000002                ADD     #2,R2             ;INECREMENT THE TEST NUMBER
1398 034774  020237  035010                CMP     R2,9$             ;SEE IF FINISHED
1399 035000  101761                        BLOS    8$                ;BR IF NOT
1400 035002  162702  000002                SUB     #2,R2             ;CORRECT TEST NUMBER
1401 035006  000402                        BR      GTTST4            ;CONTINUE
1402 035010  000000              9$:       .WORD   0                 ;STORE TEST NUMBER HERE
1403 035012  005201              GTTST3:   INC     R1                ;MOVE TO NEXT CHARACTER
1404 035014  121127  000056      GTTST4:   CMPB    (R1),#'.          ;''PERIOD''?
1405 035020  001441                        BEQ     GTTST5            ;YES--BRANCH
1406 035022  005737  001330                TST     TSTNMS            ;ANY TEST SELECTED THIS CYCLE?
1407 035026  001005                        BNE     1$                ;BR IF YES
1408 035030  005737  001332                TST     TSTNMS+2          ;ANY TEST SELECTED THIS CYCLE ?
1409 035034  001002                        BNE     1$                ;BR IF YES
1410 035036  000137  034364                JMP     GTTST1            ;NO
1411 035042  121127  000057      1$:       CMPB    (R1),#'/          ;''OPEN''?
1412 035046  001004                        BNE     2$                ;NO--BRANCH
1413 035050  056264  001526  001334        BIS     BITS(R2), OPNFLG(R4)    ;YES--SET BITS FOR TEST TO OPEN
1414 035056  000405                        BR      3$
1415 035060  121127  000054      2$:       CMPB    (R1),#',          ;''COMMA''?
1416 035064  001402                        BEQ     3$                ;BR IF YES
1417 035066  000137  034364                JMP     GTTST1            ;NO
1418 035072  005201              3$:       INC     R1                ;MOVE TO NEXT CHARACTER
1419 035074  105711                        TSTB    (R1)              ;''CR''?
1420 035076  001402                        BEQ     4$                ;BR IF 'CR'
1421 035100  000137  034450                JMP     GTTST2            ;NO--GO GET NEXT CHARACTER
1422 035104  005737  001334      4$:       TST     OPNFLG            ;ANY TESTS TO OPEN ?
1423 035110  001042                        BNE     OPNTST            ;BR IF YES
1424 035112  005737  001336                TST     OPNFLG+2          ;ANY TEST? TO OPEN ?
1425 035116  001037                        BNE     OPNTST            ;BR IF YES
1426 035120  000137  034364                JMP     GTTST1            ;NO--START AGAIN
1427 035124  005201              GTTST5:   INC     R1                ;MOVE TO NEXT CHARACTER
1428 035126  121127  000056                CMPB    (R1),#'.          ;''PERIOD''?
1429 035132  001414                        BEQ     GTTST6            ;YES--BRANCH
1430 035134  105711                        TSTB    (R1)              ;''CR''?
1431 035136  001402                        BEQ     1$                ;YES--BRANCH
1432 035140  000137  034364                JMP     GTTST1
1433 035144  005737  001334      1$:       TST     OPNFLG            ;ANY TESTS TO OPEN ?
1434 035150  001022                        BNE     OPNTST            ;BR IF YES
1435 035152  005737  001336                TST     OPNFLG+2          ;ANY TESTS TO OPEN ?
1436 035156  001017                        BNE     OPNTST            ;BR IF YES
1437 035160  000137  034360                JMP     GT.PR2            ;NO--GO START TESTING
1438 035164  005201              GTTST6:   INC     R1                ;MOVE TO NEXT CHARACTER
1439 035166  105711                        TSTB    (R1)              ;''CR''?
1440 035170  001402                        BEQ     1$                ;YES--BRANCH
1441 035172  000137  034364                JMP     GTTST1            ;NO--GO ASK FOR TEST
1442 035176  005737  001334      1$:       TST     OPNFLG            ;ANY TESTS TO OPEN ?
1443 035202  001005                        BNE     OPNTST            ;BR IF YES
1444 035204  005737  001336                TST     OPNFLG+2          ;ANY TESTS TO OPEN ?
```

```
                                                BNE     OPNTST          ;BR IF YES
                                                JMP     GT.PR2          ;NO--GO START TESTING

                                        ;OPEN THE SELECTED TEST FOR CHANGES

                                        OPNTST: SAVREG                  ;SAVE R0 - R5
                                                CLR     (PC)+           ;START WITH TEST 0
                                        OPN.CT: .WORD   0               ;COUNT STORED HERE
                                                BR      OPN.2           ;SKIP THE INCREMENT

                                        OPN.1:  INC     OPN.CT          ;MOVE TO THE NEXT TEST
                                                CMP     #22,OPN.CT      ;TEST NUMBER TOO BIG?
                                                BGE     OPN.2           ;NO--OPEN THE NEXT TEST
                                                RESREG                  ;RESTORE R0 - R5
                                                JMP     GTTST1          ;YES--GO ASK FOR MORE TESTS

                                        OPN.2:  MOV     OPN.CT,R5       ;SETUP TO USE THE
                                                ASL     R5              ;TEST NUMBER AS AN INDEX
                                                MOV     OPN.CT,R3       ;GET INDEX
                                                BIC     #17,R3          ;CLEAR LOWER TEST BITS
                                                ASR     R3              ;SHIFT TEST NUMBER
                                                ASR     R3
                                                ASR     R3
                                                BIT     BITS(R5),OPNFLG(R3)     ;OPEN THIS TEST?
                                                BEQ     OPN.1           ;NO--MOVE TO NEXT TEST
                                                TYPE    ,65$            ;;TYPE ASCIZ STRING
                                                BR      64$             ;;GET OVER THE ASCIZ
                                        ;;65$:  .ASCIZ  / TEST /
                                        64$:
                                                MOV     OPN.CT,-(SP)    ;;SAVE OPN.CT FOR TYPEOUT
                                                TYPOS                   ;;GO TYPE--OCTAL ASCII
                                                .BYTE   2               ;;TYPE 2 DIGIT(S)
                                                .BYTE   0               ;;SUPPRESS LEADING ZEROS
                                                TYPE    ,$CRLF          ;CR-LF
                                                MOV     PRMPT(R5),R0    ;PICKUP PARAMETER POINTER
                                                MOV     (R0),-(SP)      ;SAVE THE VARIABLE INDICATOR
                                                MOV     #PRM,R2         ;FIRST ADDRESS OF TABLE
                                                BR      2$
                                        1$:     ASR     (SP)            ;CHECK FOR A VARIABLE
                                                BCS     2$              ;GO MOVE THIS ONE
                                                BEQ     OPNPRM          ;DONE
                                                TST     (R2)+           ;BUMP THE POINTER
                                                BR      1$
                                        2$:     MOV     (R0)+,(R2)+     ;MOVE THIS VARIABLE INTO THE
                                                BR      1$              ;COMMON AREA

                                        OPNPRM: MOV     PRM,(SP)        ;GET THE VARIABLE INDICATOR
                                                CLR     R4              ;ZERO THE INDEX
                                        1$:     ASR     (SP)            ;CHECK FOR A VARIABLE
                                                BCS     3$              ;GO GET IT
                                                BEQ     OPNPRM          ;OUT OF VARIABLES
                                        2$:     TST     (R4)+           ;UPDATE THE INDEX
                                                BR      1$
                                        3$:     TST     PRMLMT(R4)      ;IS THE MAX. MAGNITUDE NEG?
                                                BMI     OPNPAT          ;YES--THEN IT IS THE PATTERN
                                                TYPE    ,BLNKS2         ;TYPE 2 SPACES
                                                MOV     PRMMSG(R4),4$   ;TYP  THE NAME OF THIS VARIABLE
```

```
1498 035430  104401            TYPE
1499 035432  000000      4$:   .WORD   0
1500 035434  104401  045700  TYPE    ,MSG.EQ          ;TYPE "="
1501 035440  016446  002324  MOV     RPT(R4),-(SP)    ;PUT RPT(R4) ON THE STACK
     035444  004737  025260  JSR     PC,$SB2D         ;CHANGE TO DECIMAL ASCIZ
     035450  004737  025510  JSR     PC,$SUPRS        ;TYPE WITHOUT LEADING ZEROS
1502 035454  104401  045725  TYPE    ,SLASH           ;TYPE ' / '
1503 035460  104411          RDLIN
1504 035462  012601          MOV     (SP)+,R1         ;READ AN ASCIZ STRING
1505 035464  004037  036340  JSR     R0,CK.CHR        ;CHECK ONE CHARACTER
     035470  035410          3$                       ;ILLEGAL CHARACTER
     035472  035404          2$                       ;CARRIAGE RETURN
     035474  035542          9$                       ;"/"
     035476  035504          5$                       ;","
     035500  035512          6$                       ;"."
     035502  035540          8$                       ;DIGIT 0-9
1506 035504  105711      5$: TSTB    (R1)             ;"CR"?
1507 035506  001340          BNE     3$               ;NO--STAY ON THIS VARIABLE
1508 035510  000735          BR      2$               ;YES--MOVE TO NEXT VARIABLE
1509 035512  105711      6$: TSTB    (R1)             ;IS THERE A "CR" AFTER THE PERIOD?
1510 035514  001002          BNE     7$               ;NO
1511 035516  000137  036132  JMP     OPN.N2           ;YES--GO CLOSE THIS TEST
1512 035522  122721  000056  7$:   CMPB    #'.,(R1)+        ;DOUBLE PERIOD?
1513 035526  001330          BNE     3$               ;NO--GO ASK FOR THIS VARIABLE
1514 035530  105711          TSTB    (R1)             ;YES--IS A "CR" AFTER THE DOUBLE PERIOD?
1515 035532  001326          BNE     3$               ;NO--ASK FOR THIS VARIABLE AGAIN
1516 035534  000137  036150  JMP     OPN.X2           ;YES--CLOSE ALL TEST
1517
1518 035540  005301      8$: DEC     R1               ;BACK THE POINTER UP BY ONE
1519 035542          9$:
     035542  016402  002432  MOV     PRMLMT(R4),R2            ;UPPER LIMIT OF INPUT
     035546  004037  036414  JSR     R0,CK.DIG        ;CHECK THE DIGIT(S)
     035552  035410          3$                       ;ILLEGAL INPUT
     035554  035410          3$                       ;INPUT TO LARGE
     035556  035564          10$                      ;TERMINATED WITH A "," OR "CR"
     035560  036126          OPN.N1                   ;TERMINATED WITH A "."
     035562  036144          OPN.X1                   ;TERMINATED WITH A ".."
1520 035564  010264  002324  10$:  MOV     R2,RPT(R4)       ;SAVE THIS VARIABLE
1521 035570  000705          BR      2$               ;MOVE TO NEXT VARIABLE
1522
1523                        ;OPEN PATTERN FOR CHANGES
1524
1525 035572  104401  046657  OPNPAT: TYPE    ,BLNKS2          ;TYPE 2 SPACES
1526 035576  104401  045674          TYPE    ,MSG.PAT         ;TYPE "PAT"
1527 035602  104401  045700          TYPE    ,MSG.EQ          ;TYPE "="
1528 035606  016446  002324          MOV     RPT(R4),-(SP)    ;;SAVE RPT(R4) FOR TYPEOUT
     035612  104402                  TYPOC                    ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
1529 035614  104401  046660          TYPE    ,BLNKS1          ;TYPE ONE SPACE
1530 035620  104411                  RDLIN                    ;READ ASCIZ STRING
1531 035622  012601                  MOV     (SP)+,R1         ;PICKUP POINTER
1532 035624  004037  036340          JSR     R0,CK.CHR        ;CHECK ONE CHARACTER
     035630  035572                  OPNPAT                   ;ILLEGAL CHARACTER
     035632  035370                  OPNPRM                   ;CARRIAGE RETURN
     035634  035666                  3$                       ;"/"
     035636  035570                  OPNPRM                   ;","
     035640  035644                                           ;"."
     035642  035664                  <>                       ;DIGIT 0-9
```

```
 1533 035644  105711                1$:     TSTB    (R1)            ;"CR" AFTER THE PERIOD?
 1534 035646  001531                        BEQ     OPN.N2          ;YES--GO CLOSE THIS TEST
 1535 035650  122721  000056                CMPB    #'.,(R1)+       ;NO--PERIOD?
 1536 035654  001346                        BNE     OPNPAT          ;NO--LOOP
 1537 035656  105711                        TSTB    (R1)            ;"CR" AFTER A DOUBLE PERIOD?
 1538 035660  001533                        BEQ     OPN.X2          ;YES--GO START TESTING
 1539 035662  000743                        BR      OPNPAT          ;NO--LOOP
 1540 035664  005301                2$:     DEC     R1              ;BACKUP THE ASCII POINTER
 1541 035666                        3$:
      035666  004037  036600                JSR     R0,CK.NUM       ;CHECK THE NUMBER
      035672  035572                        OPNPAT                  ;ILLEGAL INPUT
      035674  035702                        4$                      ;TERMINATED WITH A ",." OR "CR"
      035676  036126                        OPN.N1                  ;TERMINATED WITH A "."
      035700  036144                        OPN.X1                  ;TERMINATED WITH A ".."
 1542 035702  010264  002324        4$:     MOV     R2,RPT(R4)      ;SAVE THE INPUT NUMBER
 1543 035706  006002                        ROR     R2              ;OPEN PATTERN 0?
 1544 035710  103227                        BCC     OPNPRM          ;NO--START AT BEGINNING OF PARAMETER TABLE
 1545 035712  104412                        SAVREG                  ;SAVE R0 - R5
 1546
 1547                               ;OPEN WORDS IN PATTERN #0 FOR CHANGES
 1548
 1549 035714  005000                OPNWDS: CLR     R0              ;START WITH WORD 0
 1550 035716  012704  003560                MOV     #PAT0,R4
 1551 035722                        1$:
      035722  104401  035730                TYPE    ,65$            ;;TYPE ASCIZ STRING
      035726  000403                        BR      64$             ;;GET OVER THE ASCIZ
                                   ;;65$:  .ASCIZ  /   WD/
      035736                       64$:
 1552 035736  010046                        MOV     R0,-(SP)        ;PUT R0 ON THE STACK
      035740  004737  025260                JSR     PC,$SB2D        ;CHANGE TO DECIMAL ASCIZ
      035744  004737  025510                JSR     PC,$SUPRS       ;TYPE WITHOUT LEADING ZEROS
 1553 035750  104401  045700                TYPE    ,MSG.EQ         ;TYPE "="
 1554 035754  011446                        MOV     (R4),-(SP)      ;;SAVE (R4) FOR TYPEOUT
      035756  104402                        TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 1555 035760  104411                        RDLIN                   ;READ ASCIZ STRING
 1556 035762  012601                        MOV     (SP)+,R1        ;PICKUP THE POINTER
 1557 035764  004037  036340                JSR     R0,CK.CHR       ;CHECK ONE CHARACTER
      035770  035722                        1$                      ;ILLEGAL CHARACTER
      035772  036024                        5$                      ;CARRIAGE RETURN
      035774  036006                        3$                      ;"/"
      035776  036024                        5$                      ;","
      036000  036040                        6$                      ;"."
      036002  036004                        2$                      ;DIGIT 0-9
 1558 036004  005301                2$:     DEC     R1              ;BACKUP THE ASCII POINTER
 1559 036006                        3$:
      036006  004037  036600                JSR     R0,CK.NUM       ;CHECK THE NUMBER
      036012  035722                        1$                      ;ILLEGAL INPUT
      036014  036022                        4$                      ;TERMINATED WITH A ",." OR "CR"
      036016  036060                        7$                      ;TERMINATED WITH A "."
      036020  036072                        9$                      ;TERMINATED WITH A ".."
 1560 036022  010214                4$:     MOV     R2,(R4)         ;SAVE THE INPUT
 1561 036024  005724                5$:     TST     (R4)+           ;MOVE TO NEXT WORD
 1562 036026  005200                        INC     R0              ;INCREMENT THE COUNT
 1563 036030  022700  000020                CMP     #16.,R0         ;COUNT TO LARGE?
 1564 036034  003332                        BGT     1$              ;NO--BRANCH
 1565 036036  000726                        BR      OPNWDS          ;YES--BRANCH
 1566 036040  105711                6$:     TSTB    (R1)            ;"CR" AFTER THE PERIOD?
```

```
                                        BEQ     8$              ;YES--GO CLOSE THIS TEST
                                        CMPB    #'.,(R1)+       ;NO--PERIOD?
                                        BNE     1$              ;NO--BRANCH ILLEGAL INPUT STRING
                                        TSTB    (R1)            ;"CR" AFTER THE "PERIOD-PERIOD"?
                                        BEQ     10$             ;YES--GO START TESTING
                                        BR      1$              ;NO--LOOP
                                7$:     MOV     R2,(R4)+        ;SAVE THE INPUT
                                8$:     JSR     PC,CLSWDS       ;CLOSE THE DATA PATTERN
                                        RESREG                  ;RESTORE R0 - R5
                                        BR      OPN.N2          ;MOVE TO NEXT TEST
                                9$:     MOV     R2,(R4)+        ;SAVE THE INPUT
                                10$:    JSR     PC,CLSWDS       ;CLOSE THE DATA PATTERN
                                        RESREG                  ;RESTORE R0 - R5
                                        BR      OPN.X2          ;START TESTING

                                ;CLOSE PATTERN #0 AND SAVE CHANGED WORDS

                                CLSWDS: MOV     #PAT0,R1        ;FIRST ADDRESS OF DATA PATTERN
                                1$:     INC     R0              ;COUNT THE LAST WORD THAT WAS STORED
                                        CMP     #15.,R0         ;END OF TABLE
                                        BLT     2$              ;YES--EXIT
                                        MOV     (R1)+,(R4)+     ;COPY
                                        BR      1$              ;LOOP
                                2$:     RTS     PC              ;RETURN

                                OPN.N1: MOV     R2,RPT(R4)      ;SAVE THIS VARIABLE
                                OPN.N2: TST     (SP)+           ;CLEAN OFF THE STACK
                                        JSR     PC,CLOSE        ;CLOSE THIS TEST
                                        JMP     OPN.1           ;GO OPEN THE NEXT TEST

                                OPN.X1: MOV     R2,RPT(R4)      ;SAVE THIS VARIABLE
                                OPN.X2: TST     (SP)+           ;CLEAN OFF THE STACK
                                1$:     JSR     PC,CLOSE        ;CLOSE THIS TEST
                                2$:     TST     (R5)+           ;UPDATE THE INDEX
                                        CMP     R5,#16*2        ;INDEX TO BIG?
                                        BLT     3$              ;NO--BRANCH
                                        RESREG                  ;RESTORE R0 - R5
                                        JMP     GT.PR2          ;GO TO EXIT
                                3$:     BIT     BITS(R5),R3     ;IS THIS TEST OPEN FOR CHANGE?
                                        BNE     1$              ;YES--GO CLOSE IT
                                        BR      2$              ;NO--MOVE TO NEXT TEST

                                ;CLOSE CURRENT TEST THAT WAS OPEN FOR CHANGES

                                CLOSE:  SAVREG                  ;SAVE R0 - R5
                                        MOV     #PRM,R0         ;"FROM" ADDRESS
                                        MOV     PRMPT(R5),R1    ;"TO" ADDRESS
                                        MOV     (R0)+,R2        ;"FROM" INDICATOR
                                        MOV     (R1)+,R3        ;"TO" INDICATOR
                                        MOV     #1,R4           ;TEST BIT START A "RPT"
                                1$:     BIT     R4,R2           ;PARAMETER TO BE MOVED?
                                        BEQ     2$              ;NO--BRANCH
                                        BIT     R4,R3           ;A PLACE TO PUT IT?
                                        BEQ     3$              ;NO--BRANCH
                                        MOV     (R0),(R1)       ;YES--MOVE "FROM" TO "TO"
                                2$:     BIT     R4,R3           ;"TO" PARAMETER?
                                        BEQ     3$              ;NO--BRANCH
```

```
1624 036244  005721                    TST     (R1)+           ;YES--UPDATE THE POINTER
1625 036246  005720             3$:    TST     (R0)+           ;UPDATE FROM POINTER
1626 036250  006304                    ASL     R4              ;POSITION THE TEST BIT
1627 036252  032704  002000            BIT     #BIT10,R4       ;DONE?
1628 036256  001763                    BEQ     1$              ;NO--BRANCH
1629 036260  104413                    RESREG                  ;RESTORE R0 - R5
1630 036262  000207                    RTS     PC              ;RETURN
1631
1632                                ;THIS ROUTINE IS USED TO CHECK IF AN
1633                                ;ASCII CHARACTER IS A DIGIT BETWEEN 0 AND 7.
1634                                ;CALL
1635                                ;       MOV     #ADR,R1         ;ADDRESS OF ASCII CHARACTER
1636                                ;       JSR     R0,CK.OCT       ;CHECK THE CHARACTER
1637                                ;       RETURN1                 ;CHARACTER IS NOT BETWEEN 0-7
1638                                ;       RETURN2                 ;CHARACTER IS IN R2 AS A
1639                                ;                               ;OCTAL DIGIT
1640
1641 036264  121127  000060        CK.OCT: CMPB    (R1),#'0        ;LESS THAN ZERO?
1642 036270  103407                    BLO     1$              ;YES -- BRANCH
1643 036272  121127  000067            CMPB    (R1),#'7        ;GREATER THAN SEVEN?
1644 036276  101004                    BHI     1$              ;YES -- BRANCH
1645 036300  111102                    MOVB    (R1),R2         ;GET THE CHARACTER
1646 036302  042702  177770            BIC     #^C7,R2         ;STRIP AWAY THE ASCII
1647 036306  005720                    TST     (R0)+           ;ADJUST FOR RETURN
1648 036310  000200             1$:    RTS     R0              ;RETURN
1649
1650                                ;THIS ROUTINE IS USED TO CHECK AN ASCII CHARACTER
1651                                ;AND DETERMINE IF IT IS A DIGIT BETWEEN 0 AND 9.
1652                                ;CALL
1653                                ;       MOV     #ADR,R1         ;ADDRESS OF ASCII CHARACTER
1654                                ;       JSR     R0,CK.DEC       ;CHECK THE CHARACTER
1655                                ;       RETURN1                 ;NOT BETWEEN 0 AND 9
1656                                ;       RETURN2                 ;BETWEEN 0 AND 9
1657                                ;                               ;R2 = DIGIT
1658
1659 036312  121127  000060        CK.DEC: CMPB    (R1),#'0        ;LESS THAN ZERO?
1660 036316  103407                    BLO     1$              ;YES -- BRANCH
1661 036320  121127  000071            CMPB    (R1),#'9                ;GREATER THAN NINE?
1662 036324  101004                    BHI     1$              ;YES -- BRANCH
1663 036326  111102                    MOVB    (R1),R2         ;GET THE CHARACTER
1664 036330  042702  000060            BIC     #'0,R2          ;STRIP AWAY THE ASCII
1665 036334  005720                    TST     (R0)+           ;ADJUST FOR RETURN
1666 036336  000200             1$:    RTS     R0              ;RETURN
1667
1668                                ;THIS ROUTINE WILL CHECK AN ASCII CHARACTER TO
1669                                ;DETERMINE WHAT IT IS.
1670                                ;CALL
1671                                ;       MOV     #ADR,R1         ;ADDRESS OF ASCII CHARACTER
1672                                ;       JSR     R0,CK.CHR       ;CHECK CHARACTER
1673                                ;       RETURN  ADR1            ;UNKNOWN CHARACTER
1674                                ;       RETURN  ADR2            ;CARRIAGE RETURN * (R1)=ADR+1
1675                                ;       RETURN  ADR3            ;SLASH * (R1)=ADR+1
1676                                ;       RETURN  ADR4            ;COMMA * (R1)=ADR+1
1677                                ;       RETURN  ADR5            ;PERIOD * (R1)=ADR+1
1678                                ;       RETURN  ADR6            ;DIGIT BETWEEN 0 AND 9.
1679                                ;                               ;R2 = DIGIT * (R1)=ADR+1
1680
```

```
1681 036340  105711                CK.CHR: TSTB    (R1)            ;"CARRIAGE RETURN"?
1682 036342  001420                        BEQ     4$              ;YES -- BRANCH
1683 036344  121127  000057                CMPB    (R1),#'/        ;"SLASH"?
1684 036350  001414                        BEQ     3$              ;YES -- BRANCH
1685 036352  121127  000054                CMPB    (R1),#',        ;"COMMA"?
1686 036356  001410                        BEQ     2$              ;YES -- BRANCH
1687 036360  121127  000056                CMPB    (R1),#'.        ;"PERIOD"?
1688 036364  001404                        BEQ     1$              ;YES -- BRANCH
1689 036366  004037  036312                JSR     R0,CK.DEC       ;"DIGIT"?
1690 036372  000406                        BR      5$              ;NO -- BRANCH
1691 036374  005720                        TST     (R0)+           ;DIGIT BETWEEN 0-9
1692 036376  005720                1$:     TST     (R0)+           ;PERIOD
1693 036400  005720                2$:     TST     (R0)+           ;COMMA
1694 036402  005720                3$:     TST     (R0)+           ;SLASH
1695 036404  005720                4$:     TST     (R0)+           ;CARRIAGE RETURN
1696 036406  005201                        INC     R1              ;MOVE POINTER TO NEXT CHARACTER
1697 036410  011000                5$:     MOV     (R0),R0         ;UNKNOWN CHARACTER
1698 036412  000200                        RTS     R0              ;RETURN
1699
1700                                ;THIS ROUTINE CHECKS AN ASCII STRING FOR LEGAL
1701                                ;CHARACTERS AND FORMS A DECIMAL VALUE BINARY NUMBER IN R2.
1702                                ;CALL
1703                                ;       MOV     #ADR,R1         ;ADDRESS OF ASCIZ STRING
1704                                ;       MOV     #NUM,R2         ;MAX. MAGNITUDE OF INPUT NUMBER
1705                                ;       JSR     R0,CK.DIG       ;CHECK DIGITS
1706                                ;       RETURN  ADR1            ;ILLEGAL CHARACTER -- R2=?
1707                                ;       RETURN  ADR2            ;INPUT NUMBER TO LARGE -- R2=?
1708                                ;       RETURN  ADR3            ;"COMMA" -- R2 = NUMBER
1709                                ;       RETURN  ADR4            ;"PERIOD" -- R2 = NUMBER
1710                                ;       RETURN  ADR5            ;"PERIOD-PERIOD" -- R2 = NUMBER
1711
1712 036414  010446                CK.DIG: MOV     R4,-(SP)        ;SAVE R4
1713 036416  010346                        MOV     R3,-(SP)        ;SAVE R3
1714 036420  010246                        MOV     R2,-(SP)        ;SAVE THE MAX. SIZE ON THE STACK
1715 036422  005002                        CLR     R2              ;START WITH 0
1716 036424  005003                        CLR     R3
1717 036426  005004                        CLR     R4
1718 036430  004037  036340                JSR     R0,CK.CHR       ;CHECK ONE CHARACTER
     036434  036564                        9$                      ;ILLEGAL CHARACTER
     036436  036564                        9$                      ;CARRIAGE RETURN
     036440  036564                        9$                      ;"/"
     036442  036564                        9$                      ;","
     036444  036564                        9$                      ;"."
     036446  036450                        1$                      ;DIGIT 0-9
1719 036450  006303                1$:     ASL     R3              ;2
1720 036452  010346                        MOV     R3,-(SP)        ;SAVE *2
1721 036454  006303                        ASL     R3              ;4
1722 036456  006303                        ASL     R3              ;8
1723 036460  062603                        ADD     (SP)+,R3        ;(*8)+(*2)=*10.
1724 036462  060203                        ADD     R2,R3           ;UPDATE THE INPUT NUMBER
1725 036464  004037  036340                JSR     R0,CK.CHR       ;CHECK ONE CHARACTER
     036470  036564                        9$                      ;ILLEGAL CHARACTER
     036472  036504                        2$                      ;CARRIAGE RETURN
     036474  036564                        9$                      ;"/"
     036476  036512                        4$                      ;","
     036500  036510                        3$                      ;"."
     036502  036450                        1$                      ;DIGIT 0-9
```

```
1726 036504 005301                2$:   DEC     R1              ;BACKUP THE CHARACTER POINTER
1727 036506 000401                      BR      4$              ;CONTINE
1728 036510 005724                3$:   TST     (R4)+           ;"PERIOD"
1729 036512 005724                4$:   TST     (R4)+           ;"COMMA" OR "CR"
1730 036514 004037 036340               JSR     R0,CK.CHR       ;CHECK ONE CHARACTER
     036520 036564                      9$                      ;ILLEGAL CHARACTER
     036522 036554                      7$                      ;CARRIAGE RETURN
     036524 036564                      9$                      ;"/"
     036526 036564                      9$                      ;"."
     036530 036534                      5$                      ;","
     036532 036544                      6$                      ;DIGIT 0-9
1731 036534 005724                5$:   TST     (R4)+           ;"PERIOD-PERIOD"
1732 036536 105711                      TSTB    (R1)            ;"CR"?
1733 036540 001405                      BEQ     7$              ;YES--BRANCH
1734 036542 000410                      BR      9$
1735 036544 126127 177776 000054  6$:   CMPB    -2(R1),#',      ;WAS CHARACTER BEFORE THE DIGIT A COMMA?
1736 036552 001004                      BNE     9$              ;NO--EXIT
1737 036554 020316                7$:   CMP     R3,(SP)         ;INPUT TO LARGE?
1738 036556 101001                      BHI     8$              ;YES -- BRANCH
1739 036560 060400                      ADD     R4,R0           ;ADJUST RETURN ADDRESS
1740 036562 005720                8$:   TST     (R0)+
1741 036564 010302                9$:   MOV     R3,R2           ;NUMBER TO R2
1742 036566 005726                      TST     (SP)+           ;CLEAN MAX. SIZE OFF OF STACK
1743 036570 012603                      MOV     (SP)+,R3        ;RESTORE R3
1744 036572 012604                      MOV     (SP)+,R4        ;RESTORE R4
1745 036574 011000                      MOV     (R0),R0         ;GET RETURN ADDRESS
1746 036576 000200                      RTS     R0              ;RETURN
1747
1748                                    ;THIS ROUTINE CHECKS AN ASCIZ STRING FOR LEGAL CHARACTERS
1749                                    ;AND FORMS AN OCTAL NUMBER IN R2
1750                                    ;CALL:
1751                                    ;       MOV     #ADR,R1         ;ADDRESS OF ASCIZ STRING
1752                                    ;       JSR     R0,CK.NUM       ;GO FORM THE NUMBER
1753                                    ;       RETURN  ADR1            ;ILLEGAL CHARACTER IN THE INPUT STRING
1754                                    ;       RETURN  ADR2            ;"COMMA" OR "CR"--R2=NUMBER
1755                                    ;       RETURN  ADR3            ;"PERIOD"--R2=NUMBER
1756                                    ;       RETURN  ADR4            ;"PERIOD-PERIOD"--R2=NUMBER
1757
1758 036600 010346               CK.NUM: MOV     R3,-(SP)        ;SAVE R3
1759 036602 005003                       CLR     R3              ;START NUMBER AT ZERO
1760 036604 004037 036264                JSR     R0,CK.OCT       ;OCTAL DIGIT?
1761 036610 000440                       BR      6$              ;NO--BRANCH
1762 036612 005201               1$:     INC     R1              ;MOVE TO NEXT CHARACTER
1763 036614 006303                       ASL     R3              ;FOR THE OCTAL NUMBER IN R3
1764 036616 103435                       BCS     6$              ;DON'T LET IT GET TO BIG
1765 036620 006303                       ASL     R3
1766 036622 103433                       BCS     6$
1767 036624 006303                       ASL     R3
1768 036626 103431                       BCS     6$
1769 036630 060203                       ADD     R2,R3
1770 036632 004037 036264                JSR     R0,CK.OCT       ;IS THIS AN OCTAL DIGIT?
1771 036636 000401                       BR      2$              ;NO--FIND OUT WHAT IT IS
1772 036640 000764                       BR      1$              ;YES--MAKE IT PART OF THE NUMBER
1773 036642 010302               2$:     MOV     R3,R2           ;SAVE THE OCTAL NUMBER
1774 036644 005003                       CLR     R3              ;START WITH ZERO INDEX
1775 036646 004037 036340                JSR     R0,CK.CHR       ;CHECK ONE CHARACTER
     036652 036712                       6$                      ;ILLEGAL CHARACTER
```

```
        036654  036702                          5$                      ;CARRIAGE RETURN
        036656  036712                          6$                      ;'/'
        036660  036702                          5$                      ;'-'
        036662  036666                          3$                      ;'.'
        036664  036712                          6$                      ;DIGIT 0-9
  1776  036666  005723                  3$:     TST     (R3)+           ;''PERIOD''
  1777  036670  121127  000056                  CMPB    (R1),#'.        ;''PERIOD-PERIOD''?
  1778  036674  001002                          BNE     5$              ;NO--BRANCH
  1779  036676  005201                          INC     R1              ;YES--ADVANCE THE POINTER
  1780  036700  005723                  4$:     TST     (R3)+           ;''PERIOD-PERIOD''
  1781  036702  005723                  5$:     TST     (R3)+           ;''COMMA''
  1782  036704  105711                          TSTB    (R1)            ;''CR''?
  1783  036706  001001                          BNE     6$              ;NO--BRANCH
  1784  036710  060300                          ADD     R3,R0           ;YES--SAVE THE OCTAL NUMBER
  1785  036712  012603                  6$:     MOV     (SP)+,R3        ;RESTORE R3
  1786  036714  011000                          MOV     (R0),R0         ;PICKUP EXIT ADDRESS
  1787  036716  000200                          RTS     R0              ;RETURN
  1788
  1795
```

```
                                .SBTTL  SINGLE/DUAL PORT RH/RM DRIVER (REV 6.2) 1980

                                ;NEW DRIVE TYPE ID FOR RM02 **********
                                ;10-AUG-77 ***********
                                ;10-MAR-78 THE SC, SC5 CHANGES
                                ;NEW DRIVE TYPE ID FOR RM05 **********
                                ;1980 ***********

                                ;COPYRIGHT (C) 1977
                                ;DIGITAL EQUIPMENT CORP.
                                ;MAYNARD, MA 01754
                                ;AUTHOR(S): JIM LACEY/CHUCK HESS/

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

                                ;STORAGE FOR RMDS, RMER1, RMER2, AND RMMR2 ON AN ERROR ''2''
                                        ;RMERRS   = RMDS
                                        ;RMERRS+2 = RMER1
                                        ;RMERRS+4 = RMER2
                                        ;RMERRS+6 = RMMR2

036720  000000  000000  000000  RMERRS: .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

036730      000                 DRVACT: .BYTE   0               ;DRIVE 0
036731      000                         .BYTE   0               ;DRIVE 1
036732      000                         .BYTE   0               ;DRIVE 2
036733      000                         .BYTE   0               ;DRIVE 3
036734      000                         .BYTE   0               ;DRIVE 4
036735      000                         .BYTE   0               ;DRIVE 5
036736      000                         .BYTE   0               ;DRIVE 6
036737      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

036740      000                 DRVSTA: .BYTE   0               ;DRIVE 0
036741      000                         .BYTE   0               ;DRIVE 1
036742      000                         .BYTE   0               ;DRIVE 2
036743      000                         .BYTE   0               ;DRIVE 3
036744      000                         .BYTE   0               ;DRIVE 4
036745      000                         .BYTE   0               ;DRIVE 5
036746      000                         .BYTE   0               ;DRIVE 6
036747      000                         .BYTE   0               ;DRIVE 7

                                ;TABLE OF DRIVE TYPES (DRVTYP=8 BYTES)
                                        ;DRVTYP=0 IF DRIVE IS NONEXISTENT (DRVSTA=0, ALSO)
                                        ;DRVTYP=7 IF DRIVE IS RM05 ******
                                        ;DRVTYP=5 IF DRIVE IS RM02 ******
                                        ;DRVTYP=4 IF DRIVE IS RM03
                                        ;DRVTYP=-1 IF NOT RM05/3/2
```

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

036760  000     DPINT:  .BYTE   0               ;DRIVE 0
036761  000             .BYTE   0               ;DRIVE 1
036762  000             .BYTE   0               ;DRIVE 2
036763  000             .BYTE   0               ;DRIVE 3
036764  000             .BYTE   0               ;DRIVE 4
036765  000             .BYTE   0               ;DRIVE 5
036766  000             .BYTE   0               ;DRIVE 6
036767  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

036770  000     DPRQS:  .BYTE   0               ;DRIVE 0
036771  000             .BYTE   0               ;DRIVE 1
036772  000             .BYTE   0               ;DRIVE 2
036773  000             .BYTE   0               ;DRIVE 3
036774  000             .BYTE   0               ;DRIVE 4
036775  000             .BYTE   0               ;DRIVE 5
036776  000             .BYTE   0               ;DRIVE 6
036777  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.

037000  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.

037002  000000  SRCHWT: .WORD   0

                        ;RM DRIVER ACTIVE FLAG (ACTDRV=1 BYTE)
                        ;ACTDRV=0 IF DRIVER IS INACTIVE
                        ;ACTDRV>0 IF DRIVER IS ACTIVE

037004  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

 037005  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

 037006  000                    ULDFLG: .BYTE   0               ;DRIVE 0
 037007  000                            .BYTE   0               ;DRIVE 1
 037010  000                            .BYTE   0               ;DRIVE 2
 037011  000                            .BYTE   0               ;DRIVE 3
 037012  000                            .BYTE   0               ;DRIVE 4
 037013  000                            .BYTE   0               ;DRIVE 5
 037014  000                            .BYTE   0               ;DRIVE 6
 037015  000                            .BYTE   0               ;DRIVE 7

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

 037016  000                    LACNT:  .BYTE   0               ;DRIVE 0
 037017  000                            .BYTE   0               ;DRIVE 1
 037020  000                            .BYTE   0               ;DRIVE 2
 037021  000                            .BYTE   0               ;DRIVE 3
 037022  000                            .BYTE   0               ;DRIVE 4
 037023  000                            .BYTE   0               ;DRIVE 5
 037024  000                            .BYTE   0               ;DRIVE 6
 037025  000                            .BYTE   0               ;DRIVE 7

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

 037026  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

 037030  177777                 SEEKFG: .WORD   -1

                                ;TIMEOUT TABLE (TIMER=8 WORDS)
                                        ;THIS TABLE CONTAINS THE TIME ALLOWED FOR AN OPERATION

 037032  177777                 TIMER:  .WORD   -1              ;DRIVE 0
 037034  177777                         .WORD   -1              ;DRIVE 1
 037036  177777                         .WORD   -1              ;DRIVE 2
 037040  177777                         .WORD   -1              ;DRIVE 3
 037042  177777                         .WORD   -1              ;DRIVE 4
 037044  177777                         .WORD   -1              ;DRIVE 5
 037046  177777                         .WORD   -1              ;DRIVE 6
```

```
                037050  177777                          .WORD   -1              ;DRIVE 7
    141
    142                                         ;DATA TRANSFER UNDERWAY INDICATOR (DTUW=1 WORD)
    143                                                 ;DTUW<0 IF NO DATA TRANSFER UNDERWAY
    144                                                 ;DTUW=+N (WHERE N=0 TO 7) IMPLIES DATA TRANSFER UNDERWAY ON DRIVE N
    145
    146 037052  177777                          DTUW:   .WORD   -1
    147
    148                                         ;ATTENTION BITS TABLE (ATABIT=8 BYTES)
    149                                                 ;THIS TABLE CONTAINS THE CORRESPONDING BIT TO EACH DRIVES
    150                                                 ;ATTENTION BIT
    151
    152 037054      001                         ATABIT: .BYTE   1       ;DRIVE 0
    153 037055      002                                 .BYTE   2       ;DRIVE 1
    154 037056      004                                 .BYTE   4       ;DRIVE 2
    155 037057      010                                 .BYTE   10      ;DRIVE 3
    156 037060      020                                 .BYTE   20      ;DRIVE 4
    157 037061      040                                 .BYTE   40      ;DRIVE 5
    158 037062      100                                 .BYTE   100     ;DRIVE 6
    159 037063      200                                 .BYTE   200     ;DRIVE 7
    160
    161                                         ;FSRM TO RH11/RH70 "MASSBUS CONTROL BUS PARITY ERRORS" (MCPE) ALLOWED BEFORE
    162                                         ;CALLING IT FATAL (MCPEMX=1 WORD)
    163
    164 037064  000003                          MCPEMX: .WORD   3
    165
    166                                         ;STORAGE FOR RMADR (THE FIRST ADDRESS (776700) OF THE RH/RM),
    167                                         ;RMVEC (THE VECTOR ADDRESS (254)), AND RMVEC+2 (THE BR LEVEL (5)).
    168
    169 037066  176700                          RMADR:  .WORD   176700
    170 037070  000254  000240                  RMVEC:  .WORD   254,5*32.
    171
    172                                         ;MAXIMUM NUMBER OF LOOK AHEADS ALLOWED IS 4 (MXLACT=1 WORD)
    173 037074  000004                          MXLACT: .WORD   4
    174
    175                                         ;MAXIMUM DELTA DELAY IS 8 SECTORS (MXDLTA=1 WORD)
    176 037076  001000                          MXDLTA: .WORD   8.*64.
    177
    178                                         ;MINIMUM DELTA DELAY IS 2 SECTORS (MNDLTA=1 WORD)
    179 037100  000200                          MNDLTA: .WORD   2*64.
    180
    181                                         ;MAXIMUM SEARCH FOR I/O WINDOW IS 5 SECTORS (MXWNDW=1 WORD)
    182 037102  000005                          MXWNDW: .WORD   5
    183
    184                                         ;DEFINITIONS OF THE RH/RM ADDRESS INDEXES
    185
    186         000000                          RMCS1=0                 ;CONTROL AND STATUS REGISTER #1 (DRIVE REG. 00)
    187         000002                          RMWC=2                  ;WORD COUNT REGISTER (NOT A DRIVE REG)
    188         000004                          RMBA=4                  ;UNIBUS ADDRESS REGISTER (NOT A DRIVE REG)
    189         000006                          RMDA=6                  ;DESIRED SECTOR/TRACK ADDRESS REGISTER (DRIVE REG. 05)
    190         000010                          RMCS2=10                ;CONTROL AND STATUS REGISTER #2 (NOT A DRIVE REG)
    191         000012                          RMDS=12                 ;DRIVE STATUS REGISTER (DRIVE REG 01)
    192         000014                          RMER1=14                ;ERROR REGISTER #1 (DRIVE REG. 02)
    193         000016                          RMAS=16                 ;ATTENTION SUMMARY PSEUDO REGISTER (DRIVE REG. 04)
    194         000020                          RMLA=20                 ;LOOK AHEAD REGISTER (DRIVE REG. 07)
    195         000022                          RMDB=22                 ;DATA BUFFER REGISTER (NOT A DRIVE REG.)
    196         000024                          RMMR1=24                ;MAINTAINABILITY REGISTER (DRIVE REG. 03)
```

```
                        RMDT=26                 ;DRIVE TYPE REGISTER (DRIVE REG. 06)
                        RMSN=30                 ;SERIAL NUMBER REGISTER (DRIVE REG. 10)
                        RMOF=32                 ;OFFSET REGISTER (DRIVE REG. 11)
                        RMDC=34                 ;DESIRED CYLINDER ADDRESS REGISTER (DRIVE REG. 12)
                        RMHR=36                 ;DUMMY  ADDRESS REGISTER (DRIVE REG. 13)
                        RMMR2=40                ;MAINTENANCE REGISTER #2
                        RMER2=42                ;ERROR REGISTER #2 (DRIVE REG. 15)
                        RMEC1=44                ;ECC POSITION REGISTER (DRIVE REG. 16)
                        RMEC2=46                ;ECC PATTERN REGISTER (DRIVE REG.  17)

                        .SBTTL  RH/RM DRIVER INITIALIZATION CODE

                        ;THIS ROUTINE WILL DETERMINE WHICH RM 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,RMINIT
                        ;       RETURN
                        ;
                        ;NOTE: THE 'P' OR 'L' CLOCK MUST BE STARTED
                        ;
037104  104412          RMINIT: SAVREG                  ;SAVE R0 - R5
037106  013746  177776          MOV     PS,-(SP)        ;SAVE THE PRESENT PROCESSOR STATUS
037112  012737  000240  177776  MOV     #<5*32.>,PS     ;CHANGE THE PRIORITY TO 5
037120  004737  044772          JSR     PC,CLRQUE       ;CLEAR ALL REQUEST QUEUES
037124  012701  036720          MOV     #RMERRS,R1      ;FIRST ADDRESS TO BE CLEARED
037130  012702  037030          MOV     #SEEKFG,R2      ;LAST ADDRESS TO BE CLEARED
037134  005021          1$:     CLR     (R1)+           ;CLEAR
037136  020102                  CMP     R1,R2           ;ARE WE DONE?
037140  103775                  BLO     1$              ;BR IF NO
037142  012702  037052          MOV     #DTUW,R2        ;LAST ADDRESS
037146  012721  177777  2$:     MOV     #-1,(R1)+       ;INITIALIZE
037152  020102                  CMP     R1,R2           ;DONE?
037154  101774                  BLOS    2$              ;LOOP IF NO
037156  005037  036740          CLR     DRVSTA          ;SET ALL DRIVES TO OFFLINE
037162  005037  036742          CLR     DRVSTA+2
037166  005037  036744          CLR     DRVSTA+4
037172  005037  036746          CLR     DRVSTA+6
037176  013703  037070          MOV     RMVEC,R3        ;SETUP THE RH/RM VECTOR
037202  012723  041740          MOV     #ISR,(R3)+
037206  013713  037072          MOV     RMVEC+2,(R3)
037212  013704  037066          MOV     RMADR,R4        ;FIRST ADDRESS OF RH/RM
037216  012764  000040  000010  MOV     #CLR,RMCS2(R4)  ;MASSBUS INIT
037224  005001                  CLR     R1              ;START WITH DRIVE 0
037226  004037  037316  3$:     JSR     R0,DRVINT       ;INIT THE DRIVE
037232  000401                  BR      4$              ;'DVA' NOT SET OR PARITY ERROR
037234  000402                  BR      5$              ;NORMAL RETURN
037236  105061  036740  4$:     CLRB    DRVSTA(R1)      ;SET DRIVE STATUS TO OFFLINE
037242  005201          5$:     INC     R1              ;GO TO NEXT DRIVE
037244  042701  177770          BIC     #^C7,R1         ;MASK OUT UNUSED BITS
037250  001366                  BNE     3$              ;BR IF MORE DRIVES TO GO
037252  012701  000007          MOV     #7,R1           ;START WITH DRIVE 7
037256  005037  177776          CLR     PS              ;CLEAR THE PROCESSOR STATUS
037262  105761  036760  6$:     TSTB    DPINT(R1)       ;WAITING FOR DRIVE TO SWITCH PORTS ?
```

```
037266  001405                          BEQ     8$              ;BR NOT WAITING
037270  004737  044426                  JSR     PC,SET.IE       ;SET INTERRUPT
037274  105761  036760          7$:     TSTB    DPINT(R1)       ;DRIVE SWITCHED PORTS ?
037300  001375                          BNE     7$              ;BR IF NOT
037302  005301                  8$:     DEC     R1              ;GO TO THE NEXT DRIVE
037304  100366                          BPL     6$              ;CHECK NEXT DRIVE
037306  012637  177776                  MOV     (SP)+,PS        ;RESTORE THE PROCESSOR STATUS
037312  104413                          RESREG                  ;RESTORE R0 - R5
037314  000207                          RTS     PC              ;BYE-BYE

                        ;DRIVE INITILIZATION ROUTINE
                        ;THIS ROUTINE DETERMINES IF A DRIVE EXIST AND IF IT IS
                        ;AN RM05/3/2. IF IT IS, A "READ-IN PRESET" IS ISSUED AND FMT16
                        ;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     RMADR,R4        ;UNIBUS ADDRESS OF RH/RM (RMCS1)
                        ;       JSR     R0,DRVINT       ;CALLED BY A JSR
                        ;       RETURN1                 ;ERROR OCCURRED (PARITY)
                        ;       RETURN2                 ;NORMAL RETURN
                        ;

037316  010546          DRVINT: MOV     R5,-(SP)        ;SAVE R5
037320  105061  036740  DULP:   CLRB    DRVSTA(R1)      ;START DRIVE STATUS AS OFFLINE
037324  105061  036750          CLRB    DRVTYP(R1)      ;CLEAR THE DRIVE TYPE INDICATOR
037330  105061  037006          CLRB    ULDFLG(R1)      ;CLEAR THE UNLOAD FLAG
037334  010164  000010          MOV     R1,RMCS2(R4)    ;SELECT A DRIVE
037340  112764  000111  000000  MOVB    #111,RMCS1(R4)  ;DO A DRIVE CLEAR COMMAND (& SEIZE DRIVE)
037346  032764  010000  000010  BIT     #BIT12,RMCS2(R4)        ;NONEXISTENT DRIVE?
037354  001403                  BEQ     1$              ;NO
037356  004737  044426          JSR     PC,SET.IE       ;GO SET "IE" WITHOUT A "TRE"
037362  000520                  BR      6$              ;LEAVE THIS ROUTINE

037364  105061  036740  1$:     CLRB    DRVSTA(R1)      ;SET DRIVE STATUS TO OFFLINE
037370  032764  004000  000000  BIT     #BIT11,RMCS1(R4)        ;SEE IF DRIVE AVAILABLE
037376  001750                  BEQ     DULP            ;BR IF DRIVE NOT AVAILABLE
037400  004037  043736          JSR     R0,RD.RM        ;READ THE DRIVE TYPE REG.
037404  000026                  RMDT
037406  037650                  8$                      ;ERROR RETURN ADDRESS
037410  012605                  MOV     (SP)+,R5        ;PUT DRIVE TYPE IN R5
037412  112761  000004  036750  MOVB    #4,DRVTYP(R1)   ;SET RM03 INDICATOR
037420  022705  020024          CMP     #20024,R5       ;SINGLE PORT RM03 ?
037424  001431                  BEQ     2$              ;BR IF YES
037426  022705  024024          CMP     #24024,R5       ;DUAL PORT RM03 ?
037432  001426                  BEQ     2$              ;BR IF YES
037434  112761  000005  036750  MOVB    #5,DRVTYP(R1)   ;SET RM02 INDICATOR
037442  022705  020025          CMP     #20025,R5       ;SINGLE PORT RM02 ?
037446  001420                  BEQ     2$              ;BR IF SO
037450  022705  024025          CMP     #24025,R5       ;DUAL PORT RM02 ?
037454  001415                  BEQ     2$              ;BR IF SO
037456  112761  000007  036750  MOVB    #7,DRVTYP(R1)   ;SET RM05 INDICATOR
037464  022705  020027          CMP     #20027,R5       ;SINGLE PORT RM05 ?
037470  001407                  BEQ     2$              ;BR IF YES
037472  022705  024027          CMP     #24027,R5       ;DUAL PORT RM05 ?
037476  001404                  BEQ     2$              ;BR IF YES
```

```
                                        MOVB    #-1,DRVTYP(R1)  ;SET INDICATOR TO 'OTHER'
                                        BR      6$              ;EXIT

                                2$:     MOV     #121,-(SP)      ;DO A "READ-IN PRESET"
                                        JSR     R0,WRT.RM
                                        RMCS1
                                        8$
                                        MOV     #BIT12,-(SP)    ;SET FMT16=1
                                        JSR     R0,WRT.RM
                                        RMOF
                                        8$
                                        JSR     R0,RD.RM        ;READ RMDS
                                        RMDS
                                        8$
                                        MOV     (SP)+,R5        ;AND SAVE IT IN R5
                                        BPL     4$              ;BR IF ATA=0
                                        MOVB    ATABIT(R1),RMAS(R4)     ;CLEAR ATTENTION BIT
                                        JSR     R0,RD.RM        ;FIND OUT WHY ATA=1
                                        RMER1
                                        8$
                                        ROL     (SP)+           ;IS IT UNSAFE?
                                        BPL     4$              ;BR IF NOT
                                        MOVB    #-1,DRVSTA(R1)  ;SET UNSAFE INDICATOR
                                        BR      6$              ;EXIT

                                4$:     COM     R5              ;CHECK MOL, DPR, DRY, AND VV
                                        BIC     #^C<BIT12!BIT08!BIT07!BIT06>,R5
                                        BNE     6$              ;BR IF MOL, DPR, DRY, OR VV IS CLEAR
                                        MOVB    #1,DRVSTA(R1)   ;SET DRIVE STATUS TO ONLINE
                                6$:     TST     (R0)+           ;STEP OVER THE ERROR RETURN
                                        BR      8$              ;EXIT
                                7$:     ASL     R1              ;CHANGE INDEX TO ADDRESS WORDS
                                        MOV     #60000,TIMER(R1)        ;START 2 SEC TIMER
                                        ASR     R1              ;RESTORE R1
                                        MOVB    #-1,DPINT(R1)   ;SET PORT INITIALIZE INIDICATOR
                                8$:     MOV     (SP)+,R5        ;RESTORE R5
                                        RTS     R0              ;EXIT

                                ;REQUEST PRE-PROCESSOR-HANDLES SUBSYSTEM REQUEST
                                ;
                                ;CALL
                                ;
                                ;       JSR     R0,RM05         ;CALL THE RM05 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

                                RM05:   MOV     PS,-(SP)        ;SAVE THE CALLING STATUS
                                        MOV     RMVEC+2,PS      ;DON'T ALLOW ANY RM INTERRUPTS
                                        MOVB    #1,ACTDRV       ;SET "ACTIVE DRIVER" FLAG
                                        SAVREG                  ;SAVE R0 - R5
                                        MOV     (R0),R2         ;PICKUP THE DRIVE PARAMETER BLOCK POINTER
                                        CLR     16(R2)          ;CLEAR THE STATUS/ERROR INDICATOR
                                        MOVB    (R2),R1         ;PICKUP THE DRIVE NUMBER
                                        MOV     RMADR,R4        ;UNIBUS ADDRESS OF RMCS1
                                        TSTB    DRVSTA(R1)      ;CHECK DRIVES STATUS
```

```
                                        BGT     1$              ;BR IF ONLINE
037716  003014
                                        TSTB    ULDFLG(R1)      ;UNLOAD COMMAND IN QUEUE?
037720  105761  037006
                                        BNE     3$              ;BR IF YES
037724  001036
                                        TSTB    DPINT(R1)       ;TRYING TO INIT THE DRIVE
037726  105761  036760
                                        BNE     5$              ;BR IF YES
037732  001042
                                        JSR     R0,DRVINT       ;GO INIT. THE DRIVE
037734  004037  037316
                                        BR      4$              ;ERROR RETURN
037740  000434
                                        TSTB    DRVSTA(R1)      ;IS DRIVE STATUS ONLINE?
037742  105761  036740
                                        BLE     6$              ;BR IF NOT
037746  003445
1$:                                     TSTB    DPRQS(R1)       ;OUTSTANDING PORT REQUEST FOR THE DRIVE ?
037750  105761  036770
                                        BNE     5$              ;BR IF YES
037754  001031
                                        MOV     R1,RMCS2(R4)    ;SELECT THE DRIVE
037756  010164  000010
                                        JSR     R0,DRVQUE       ;PUT THIS REQUEST IN QUEUE
037762  004037  045070
                                        BR      9$              ;QUEUE IS FULL
037766  000460
                                        CMPB    #103,2(R2)      ;IS THIS REQ. FOR AN UNLOAD?
037770  122762  000103  000002
                                        BNE     2$              ;BR IF NO
037776  001003
                                        MOVB    #-1,ULDFLG(R1)  ;SET THE "UNLOAD IN QUEUE" FLAG
040000  112761  177777  037006
2$:                                     TSTB    DRVACT(R1)      ;IS THIS DRIVE ACTIVE?
040006  105761  036730
                                        BNE     8$              ;BR IF YES
040012  001043
                                        JSR     PC,OPT          ;CALL THE OPTIMIZER
040014  004737  040146
                                        BR      8$
040020  000440
3$:                                     MOV     #BIT15!BIT13,16(R2)     ;SET THE "UNLOAD IN QUEUE" ERROR FLAG
040022  012762  120000  000016
                                        BR      8$              ;EXIT
040030  000434
4$:                                     JSR     PC,CI7          ;GO HANDLE THE PARITY ERROR
040032  004737  041226
                                        BR      8$
040036  000431
5$:                                     JSR     R0,DRVQUE       ;PUT REQUEST IN QUEUE
040040  004037  045070
                                        BR      9$              ;QUEUE IS FULL
040044  000431
                                        BIT     #BIT06,(R4)     ;IE BIT SET ?
040046  032714  000100
                                        BNE     8$              ;YES
040052  001023
                                        JSR     PC,SET.IE       ;SET THE INTERRUPT
040054  004737  044426
                                        BR      8$              ;RETURN
040060  000420
6$:                                     TSTB    DRVSTA(R1)      ;SEE IF DRIVE OFFLINE OR UNSAFE
040062  105761  036740
                                        BLT     7$              ;BR IF UNSAFE
040066  002412
                                        MOV     #BIT15!BIT14,16(R2)     ;SET OFFLINE ERROR INDICATOR
040070  012762  140000  000016
                                        TSTB    DRVTYP(R1)      ;SEE IF OFFLINE OR NONEXISTENT
040076  105761  036750
                                        BNE     8$              ;BR IF OFFLINE
040102  001007
                                        MOV     #BIT15!BIT01,16(R2)     ;REPORT DRIVE NONEXISTENT
040104  012762  100002  000016
                                        BR      8$              ;GO TO EXIT
040112  000403
7$:                                     MOV     #BIT15!BIT12,16(R2)     ;DRIVE IS UNSAFE
040114  012762  110000  000016
8$:                                     RESREG                  ;RESTORE R0 - R5
040122  104413
                                        TST     (R0)+           ;SETUP FOR NORMAL RETURN
040124  005720
                                        BR      10$             ;FINISH UP, THEN EXIT
040126  000401
9$:                                     RESREG                  ;RESTORE R0 - R5
040130  104413
10$:                                    TST     (R0)+           ;CORRECT THE RETURN ADDRESS
040132  005720
                                        CLRB    ACTDRV          ;CLEAR "ACTIVE DRIVER" FLAG
040134  105037  037004
                                        MOV     (SP)+,PS        ;RETURN "PS" TO USER LEVEL
040140  012637  177776
                                        RTS     R0              ;RETURN TO CALLER
040144  000200

                                ;OPTIMIZER-CALLED FOR A PARTICULAR DRIVE
                                ;
                                ;CALL
                                ;       MOV     #DRVNUM,R1      ;DRIVE NUMBER TO R1
                                ;       JSR     PC,OPT          ;SETUP A COMMAND
                                ;
OPT:                                    SAVREG                  ;SAVE R0 - R5
040146  104412
                                        MOV     PS,-(SP)        ;SAVE PROC. STATUS
040150  013746  177776
                                        BICB    ATABIT(R1),SRCHWT       ;CLEAR LA SEACH FLAG
040154  146137  037054  037002
```

```
040162  105061  036770                  CLRB    DPRQS(R1)           ;RESET THE PORT REQ FLAG ****
040166  004737  045144                  JSR     PC,GETREQ           ;GET "DPB" POINTER OF REQUEST
040172  005702                          TST     R2                  ;IS THERE A REQUEST IN QUEUE?
040174  001472                          BEQ     7$                  ;NO--BR TO EXIT
040176  010164  000010                  MOV     R1,RMCS2(R4)        ;LOAD THE DRIVE ADDRESS *******
040202  012764  000111  000000          MOV     #111,RMCS1(R4)      ;CLEAR THE DRIVE
040210  032764  004000  000000          BIT     #BIT11,RMCS1(R4)            ;DVA SET ?
040216  001446                          BEQ     5$                  ;TO PROT REQUEST ,IF NOT
040220  105761  036740          10$:    TSTB    DRVSTA(R1)          ;IS DRIVE ONLINE?
040224  003014                          BGT     1$                  ;YES
040226  004737  045166                  JSR     PC,POPQUE           ;NO--REMOVE REQUEST FROM QUEUE
040232  012762  140000  000016          MOV     #BIT15!BIT14,16(R2)     ;SET OFFLINE STATUS/ERROR INDICATOR
040240  105761  036740                  TSTB    DRVSTA(R1)          ;IS DRIVE UNSAFE ?
040244  100053                          BPL     8$                  ;BR TO EXIT IF NOT
040246  012762  110000  000016          MOV     #BIT15!BIT12,16(R2)     ;SET UNSAFE STATUS/ERROR INDICATOR
040254  000447                          BR      8$                  ;BR TO EXIT
040256                          1$:
                                ;       MOV     #111,-(SP)          ;LOAD COMMAND ONTO THE STACK
                                ;       JSR     R0,WRT.RM           ;LOAD THE REGISTER
                                ;       RMCS1                       ;REGISTER INCREMENT
                                ;       6$                          ;ERROR RETURN ADDRESS
                                ;       BIT     #BIT11,(R4)         ;DRIVE AVAILABLE ?
                                ;       BEQ     9$                  ;BR IF NOT
040256  122762  000150  000002          CMPB    #150,2(R2)          ;IS THE REQUEST FOR I/O?
040264  002403                          BLT     2$                  ;YES
040266  004737  040612                  JSR     PC,CI4              ;CALL THE COMMAND INITIATOR
040272  000440                          BR      8$                  ;BR TO EXIT
040274  005737  037052          2$:     TST     DTUW                ;DATA TRANSFER UNDERWAY?
040300  002012                          BGE     4$                  ;YES--GO START A SEARCH
040302  005737  037030                  TST     SEEKFG              ;DO IMPLIED SEEKS?
040306  100404                          BMI     3$                  ;YES
040310  004037  041564                  JSR     R0,LA               ;NO--DO LOOK AHEAD
040314  000427                          BR      8$                  ;RETURN HERE ON A PARITY ERROR
040316  000403                          BR      4$                  ;GO START A SEARCH
040320  004737  040404          3$:     JSR     PC,CI1              ;START A DATA TRANSFER
040324  000423                          BR      8$
040326  004737  040512          4$:     JSR     PC,CI3              ;START A SEARCH
040332  000420                          BR      8$                  ;GO TO THE EXIT
040334  112761  177777  036770  5$:     MOVB    #-1,DPRQS(R1)       ;SET PORT REQUEST INDICATOR
040342  010103                          MOV     R1,R3               ;SET UP TO ADDRESS WORDS
040344  006303                          ASL     R3                  ;CONVERT TO WORD INDEX
040346  012763  060000  037032          MOV     #60000,TIMER(R3)            ;START 10 SEC TIMER
040354  000402                          BR      7$                  ;EXIT
040356  004737  041226          6$:     JSR     PC,CI7              ;PROCESS THE PARITY ERROR
040362  032714  000100          7$:     BIT     #BIT06,(R4)         ;SEE IF 'IE' ALREADY SET
040366  001002                          BNE     8$                  ;BR IF SET
040370  004737  044426                  JSR     PC,SET.IE           ;SET "IE" WITHOUT A "TRE"
040374  012637  177776          8$:     MOV     (SP)+,PS            ;RESTORE PROC. STATUS
040400  104413                          RESREG                      ;RESTORE R0 - R5
040402  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
                                        ;
040404  004737  045166          CI1:   JSR     PC,POPQUE                ;REMOVE REQUEST FROM "DRIVES WAIT" QUEUE
040410  010237  037000                 MOV     R2,TRNSWT                ;PUT REQ. IN TRANSFER WAIT QUEUE
040414  010203                         MOV     R2,R3                    ;DPB ADDRESS TO R3
040416  013704  037066                 MOV     RMADR,R4                 ;RMCS1 ADDRESS
040422  010164  000010                 MOV     R1,RMCS2(R4)             ;SELECT DRIVE
040426  062703  000004                 ADD     #4,R3                    ;DESIRED WORD COUNT
040432  062704  000002                 ADD     #2,R4                    ;RMWC ADDRESS
040436  012324                         MOV     (R3)+,(R4)+              ;LOAD WORD COUNT
040440  012324                         MOV     (R3)+,(R4)+              ;LOAD BUFFER ADDRESS
040442  012346                         MOV     (R3)+,-(SP)              ;LOAD SECTOR AND TRACK
040444  004037  044116                 JSR     R0,WRT.RM                ;CALL THE LOAD(WRITE) ROUTINE
040450  000006                         RMDA                             ;INDEX OF REGISTER TO LOAD
040452  041226                         CI7                              ;ERROR RETURN ADDRESS
040454  012346                         MOV     (R3)+,-(SP)              ;LOAD CYLINDER ADDRESS
040456  004037  044116                 JSR     R0,WRT.RM
040462  000034                         RMDC
040464  041226                         CI7
040466  016246  000002                 MOV     2(R2),-(SP)              ;LOAD "COMMAND+GO", "A17&A16", AND "PSEL"
040472  004037  044116                 JSR     R0,WRT.RM
040476  000000                         RMCS1
040500  041226                         CI7
040502  010137  037052                 MOV     R1,DTUW                  ;SET "DATA TRANSFER UNDERWAY"
040506  000137  041170                 JMP     CI5
040512  013704  037066          CI3:   MOV     RMADR,R4                 ;RMCS1 ADDRESS
040516  010164  000010                 MOV     R1,RMCS2(R4)             ;SELECT DRIVE
040522  016246  000012                 MOV     12(R2),-(SP)             ;DESIRED CYLINDER ADDRESS
040526  004037  044116                 JSR     R0,WRT.RM
040532  000034                         RMDC
040534  041226                         CI7
040536  116203  000010                 MOVB    10(R2),R3                ;PICKUP SECTOR ADDRESS
                                ;       SUB     MXWNDW,R3                ;BACKUP BY MAX. SEARCH FOR I/O WINDOW
                                ;       BGE     1$
                                ;       ADD     #32.,R3
040542  010346                  1$:    MOV     R3,-(SP)                 ;COMBINE THE ADJUSTED SECTOR WITH
040544  042716  177740                 BIC     #177740,(SP)             ;CHOP OF ALL EXENTED BITS ,IF ANY
040550  116266  000011  000001         MOVB    11(R2),1(SP)             ;THE DESIRED TRACK
040556  004037  044116                 JSR     R0,WRT.RM                ;LOAD DESIRED TRACK & SECTOR
040562  000006                         RMDA
040564  041226                         CI7
040566  012746  000131                 MOV     #131,-(SP)               ;START A SEARCH
040572  004037  044116                 JSR     R0,WRT.RM
040576  000000                         RMCS1
040600  041226                         CI7
040602  156137  037054  037002         BISB    ATABIT(R1),SRCHWT        ;SET "SEARCH WAIT" KEY
040610  000567                         BR      CI5
040612  013704  037066          CI4:   MOV     RMADR,R4                 ;RMCS1 ADDRESS
040616  010164  000010                 MOV     R1,RMCS2(R4)             ;SELECT DRIVE
040622  116203  000002                 MOVB    2(R2),R3                 ;PICKUP THE REQUESTED COMMAND
040626  122703  000131                 CMPB    #131,R3                  ;IS IT A SEARCH COMMAND?
040632  001007                         BNE     1$                       ;BR IF NO
040634  016246  000010                 MOV     10(R2),-(SP)             ;LOAD DESIRED TRACK & SECTOR
040640  004037  044116                 JSR     R0,WRT.RM
```

```
040644  000006                          RMDA
040646  041226                          CI7
040650  000403                          BR      2$                      ;GO LOAD CYLINDER
040652  122703  000105          1$:     CMPB    #105,R3                 ;IS IT A SEEK COMMAND
040656  001007                          BNE     3$                      ;BR IF NO
040660  016246  000012          2$:     MOV     12(R2),-(SP)            ;LOAD DESIRED CYLINDER
040664  004037  044116                  JSR     R0,WRT.RM
040670  000034                          RMDC
040672  041226                          CI7
040674  000546                          BR      C16
040676  122703  000115          3$:     CMPB    #115,R3                 ;IS IT AN "OFFSET" COMMAND?
040702  001013                          BNE     4$                      ;BR IF NO
040704  004037  043736                  JSR     R0,RD.RM                ;MERGE THE OFFSET VALUE INTO RMOF
040710  000032                          RMOF                            ;BUT DON'T CHANGE THE UPPER
040712  041226                          CI7
040714  116216  000001                  MOVB    1(R2),(SP)              ;BYTE WHEN LOADING THE
040720  004037  044116                  JSR     R0,WRT.RM               ;REGISTER (RMOF)
040724  000032                          RMOF
040726  041226                          CI7
040730  000530                          BR      C16                     ;GO START THE COMMAND
040732  122703  000107          4$:     CMPB    #107,R3                 ;IS IT A "RECALIBRATE" COMMAND?
040736  001525                          BEQ     C16                     ;BR IF YES
040740  122703  000117                  CMPB    #117,R3                 ;IS IT A RETURN TO CENTER?
040744  001522                          BEQ     C16                     ;BR IF YES
040746  122703  000103                  CMPB    #103,R3                 ;IS IT AN "UNLOAD" COMMAND?
040752  001016                          BNE     5$                      ;BR IF NO
040754  112761  000001  036730          MOVB    #1,DRVACT(R1)           ;SET THE DRIVE ACTIVE INDICATOR
040762  105061  036740                  CLRB    DRVSTA(R1)              ;PUT DRIVE STATUS TO OFFLINE
040766  112761  000001  037006          MOVB    #1,ULDFLG(R1)           ;SET "UNLOAD IN PROGRESS" FLAG
040774  010346                          MOV     R3,-(SP)                ;START THE "UNLOAD" COMMAND
040776  004037  044116                  JSR     R0,WRT.RM
041002  000000                          RMCS1
041004  041226                          CI7
041006  000207                          RTS     PC                      ;RETURN TO USER
041010  122703  000143          5$:     CMPB    #143,R3                 ;IS IT A "SET FORMAT" COMMAND?
041014  001014                          BNE     6$                      ;BR IF NO
041016  004037  043736                  JSR     R0,RD.RM                ;READ THE OFFSET REGISTER
041022  000032                          RMOF
041024  041226                          CI7
041026  116266  000001  000001          MOVB    1(R2),1(SP)             ;COMBINE "FMT16","ECI", AND "HCI"
041034  004037  044116                  JSR     R0,WRT.RM               ;LOAD "FMT16", "ECI", AND/OR "HCI".
041040  000032                          RMOF
041042  041226                          CI7
041044  000436                          BR      12$
041046  122703  000141          6$:     CMPB    #141,R3                 ;IS IT A "GET REGISTER" COMMAND?
041052  001023                          BNE     10$                     ;BR IF NO
041054  016203  000006          7$:     MOV     6(R2),R3                ;POINTS TO 1ST ADDRESS OF WHERE
                                                                        ;TO PUT THE REGISTER(S)
041060  116237  000010  041076          MOVB    10(R2),9$               ;INIT. THE INDEX FOR THE FIRST REG.
041066  116205  000011                  MOVB    11(R2),R5               ;INDEX OF LAST REG. TO MOVE
041072  004037  043736          8$:     JSR     R0,RD.RM                ;READ RH/RM REGISTER
041076  000000                  9$:     RMCS1                           ;INDEX OF REG. TO READ
041100  041226                          CI7
041102  012623                          MOV     (SP)+,(R3)+             ;GET THE CONTENTS OF RH/RM REG.
041104  023705  041076                  CMP     9$,R5                   ;LAST REG. BEEN READ?
041110  001414                          BEQ     12$                     ;GET OUT IF YES
041112  062737  000002  041076          ADD     #2,9$                   ;INCREASE THE INDEX BY 2
```

```
                                                BR      8$              ;LOOP--MORE TO READ
                                        10$:    CMPB    #145,R3         ;IS IT A "SELECT DRIVE" COMMAND?
                                                BEQ     12$             ;BR IF YES
                                        11$:    MOV     R3,-(SP)        ;LOAD THE COMMAND
                                                JSR     R0,WRT.RM
                                                RMCS1
                                                CI7
                                        12$:    JSR     PC,POPQUE       ;REMOVE REQ. FROM QUEUE
                                                BIS     #BIT07,16(R2)   ;SET THE "DONE" BIT
                                                TST     SAVEFG          ;SAVE THE RH/RM REGISTERS?
                                                BPL     13$             ;BR IF NO
                                                JSR     PC,SVRH70       ;YES--GO SAVE THE REGISTERS
                                        13$:    RTS     PC              ;RETURN TO USER
                                        CI5:    ASL     R1
                                                MOV     #60000,TIMER(R1)        ;SET A ONE SECOND TIMER
                                                ASR     R1
                                                MOVB    #1,DRVACT(R1)   ;SET THE DRIVE ACTIVE
                                                RTS     PC              ;RETURN TO THE USER
                                        CI6:    MOV     R3,-(SP)        ;LOAD THE COMMAND
                                                JSR     R0,WRT.RM
                                                RMCS1
                                                CI7
                                                BR      CI5
                                        CI7:    BIT     #BIT12,RMCS2(R4)        ;DRIVE NON-EXISTENT ?
                                                BNE     CI8             ;BR IF YES
                                        1$:     TST     R2              ;ANYTHING IN QUEUE ?
                                                BEQ     CI7B            ;BR IF NOT
                                                BNE     2$              ;BR IF QUEUE IS THERE
                                                RTS     PC              ;OTHERWISE EXIT
                                        2$:     MOV     #BIT15!BIT11,16(R2)     ;SET "PARITY" ERROR INDICATOR
                                                JSR     PC,SVRH70       ;GO SAVE THE RH/RM REGISTERS
                                        CI7B:   MOV     #111,-(SP)      ;DO A "DRIVE CLEAR"
                                                JSR     R0,WRT.RM
                                                RMCS1
                                                CI8
                                        2$:     JSR     PC,EMPTYQ       ;EMPTY THE QUEUE
                                                CLRB    DPRQS(R1)       ;CLEAR THE PORT REQUEST FLAG
                                                CLRB    ULDFLG(R1)      ;CLEAR THE UNLOAD IN QUEUE FLAG
                                                CLRB    DRVACT(R1)      ;DRIVE IS IDLE
                                                CMP     R2,TRNSWT       ;IF THIS DRIVE HAD AN I/O REQUEST
                                                CMP     R1,DTUW         ;IF THIS DRIVE HAD AN I/O REQUEST
                                                BNE     1$              ;IN PROGRESS CLEAR ALL OF THE FLAGS
                                                CLR     TRNSWT
                                                MOV     #-1,DTUW
                                        1$:     RTS     PC
                                        CI8:    SAVREG                  ;SAVE R0 - R5
                                                BIT     #BIT12,RMCS2(R4)        ;IS 'NED' SET ?
                                                BNE     1$              ;BR IF YES
                                                CLR     R1
                                                CLR     R3
                                        1$:     TSTB    DRVACT(R1)      ;DRIVE ACTIVE?
                                                BEQ     5$              ;BR IF NO
                                                BNE     22$             ;BR IF IN ACTIVE
                                                TSTB    DPRQS(R1)       ;PORT REQUEST
                                                BEQ     5$              ;BR IF NOT
                                        22$:    MOV     TRNSWT,R2       ;GET THE "TRANSFER WAIT" QUEUE
                                                CMP     R1,DTUW         ;DID THIS DRIVE HAVE AN I/O IN PROGRESS?
```

```
                                                BEQ     2$              ;BR IF YES
                                                JSR     PC,GETREQ       ;GET THE DPB POINTER
                                        2$:     TST     R2              ;QUEUE ENTRY FOR DRIVE ?
                                                BEQ     4$              ;BR IF NOT
                                                BIT     #BIT12,RMCS2(R4)        ;'NED' SET ?
                                                BEQ     3$              ;BR IF NOT
                                                MOV     #BIT15!BIT01,16(R2)     ;SET 'DRIVE NON-EXISTENT' INDICATOR
                                                BR      4$              ;CONTINUE
                                        3$:     MOV     #BIT15!BIT10,16(R2)     ;SET "NON-CLEARABLE PARITY" ERROR INDICATOR
                                                ;
                                        4$:     JSR     PC,SVRH70       ;SAVE RH/RM REGISTERS
                                                MOV     #-1,TIMER(R3)   ;STOP THE TIMER
                                                CLRB    DRVACT(R1)      ;SET "DRIVE ACTIVE" TO IDLE
                                                CLRB    DPRQS(R1)       ;CLEAR PORT REQUEST FLAG
                                                CMP     R1,DTUW         ;IS THIS DRIVE SETUP FOR A TRANSFER
                                                BNE     5$              ;BR IF NOT
                                                MOV     #-1,DTUW        ;RESET THE INDICATOR
                                                CLR     TRNSWT          ;CLEAR THE TRANSFER QUEUE
                                        5$:     CLRB    ULDFLG(R1)      ;CLEAR UNLOAD FLAG
                                                BIT     #BIT12,RMCS2(R4)        ;'NED' SET ?
                                        ;       BNE     6$              ;BR IF YES
                                                INC     R1              ;MOVE TO THE NEXT DRIVE
                                                ADD     #2,R3
                                                BIC     #^C7,R1
                                                BNE     1$              ;BR IF MORE DRIVES
                                                MOV     #-1,DTUW        ;NO DATA TRANSFERS UNDERWAY
                                                CLR     TRNSWT          ;CLEAR THE 'TRANSFER WAIT' QUEUE
                                                JSR     PC,CLRQUE       ;CLEAR ALL OF THE REQUEST QUEUES
                                                MOV     #CLR,RMCS2(R4)  ;DO A MASSBUS INIT.
                                                BR      7$              ;CONTINUE
                                        6$:     JSR     PC,EMPTYQ       ;CLEAR THE DRIVE'S QUEUE
                                                CLRB    DRVSTA(R1)      ;SET DRIVE TO OFFLINE
                                                CLRB    DRVTYP(R1)      ;CLEAR THE DRIVE TYPE INDICATOR
                                        7$:     JSR     PC,SET.IE       ;SET "IE" WITHOUT "TRE"
                                                RESREG                  ;RESTORE R0 - R5
                                                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

                                        LA:     MOV     RMADR,R4        ;GET RMCS1'S ADDRESS
                                                MOV     R1,RMCS2(R4)    ;SELECT DRIVE
                                                JSR     R0,RD.RM        ;READ DRIVE STATUS
                                                RMDS
                                                4$                      ;ERROR RETURN ADDRESS
                                                BIC     #^C020200,(SP)  ;ON CYLINDER ?
                                                CMP     #200,(SP)+      ;PIP=0,DRY=1?
                                                BNE     3$              ;NO
                                                INCB    LACNT(R1)       ;INCREMENT THE LOOK AHEAD COUNT
                                                CMPB    LACNT(R1),MXLACT        ;EXCEED MAX?
                                                BGT     2$              ;BR IF YES
```

```
                                        MOVB    10(R2),R3       ;GET DESIRED SECTOR ADDRESS AND
                                        SWAB    R3              ;MULT. BY 64--ALIGN WITH
                                        ASR     R3              ;LOOK AHEAD REGISTER
                                        ASR     R3
                                        MOV     #340,PS         ;PRIORITY LEVEL "7"
                                6$:     JSR     R0,RD.RM        ;READ LOOK AHEAD REGISTER
                                        RMLA
                                        4$
                                        CMP     (SP),RMLA(R4)   ;CORRECT LA NUMBER ?
                                        BEQ     7$              ;YES
                                        TST     (SP)+           ;NO,CLEAR STACK
                                        BR      3$
                                7$:     SUB     (SP)+,R3        ;CALCULATE THE DELTA
                                        BGE     1$
                                        ADD     #<32.*64.>,R3   ;MAKE THE DELTA POSITIVE
                                1$:     CMP     MXDLTA,R3       ;CHECK THE DELTA TO SEE
                                        BLT     3$              ;IF IT IS WITHIN THE
                                        CMP     MNDLTA,R3       ;WINDOW---IF YES, ZERO
                                        BGE     3$              ;THE LOOK AHEAD COUNT
                                2$:     CLRB    LACNT(R1)       ;AND TAKE THE I/O EXIT
                                        TST     (R0)+
                                3$:     TST     (R0)+           ;ADJUST THE RETURN ADDRESS
                                        BR      5$              ;EXIT
                                4$:     JSR     PC,C17          ;PROCESS THE ERROR
                                5$:     RTS     R0              ;RETURN

                                ;INTERRUPT SERVICE ROUTINE

                        ISR:    MOVB    #1,ACTDRV       ;SET "ACTIVE DRIVER" FLAG
                                SAVREG                  ;SAVE R0 - R5
                                MOV     RMADR,R4        ;ADDRESS OF RHSCS1
                                MOV     DTUW,R1         ;GET "DATA TRANSFER UNDERWAY" INDICATOR
                                BLT     1$              ;BR IF NO DATA TRANSFER UNDERWAY
                                JSR     PC,TD           ;CALL TRANSFER DONE
                                BR      2$              ;EXIT
                                JSR     PC,SC           ;CALL SPECIAL CONDITIONS
                                RESREG                  ;RESTORE R0 - R5
                                CLRB    ACTDRV          ;CLEAR "ACTIVE DRIVER" FLAG
                                RTI                     ;RETURN

                                ;TRANSFER DONE ROUTINE

                        TD:     CLRB    DRVACT(R1)      ;SET DRIVE ACTIVE INDICATOR TO IDLE
                                MOV     #-1,DTUW        ;NO DATA TRANSFERS UNDERWAY
                                ASL     R1
                                MOV     #-1,TIMER(R1)   ;CANCEL TIMEOUT
                                ASR     R1
                                MOV     TRNSWT,R2       ;GET "DPB" ADDRESS FROM THE
                                CLR     TRNSWT          ;TRANSFER WAIT QUEUE--CLEAR QUEUE
                                BIS     #BIT07,16(R2)   ;SET DONE
                                MOV     R1,RMCS2(R4)    ;SELECT THE DRIVE
                                JSR     R0,RD.RM        ;TRANSFER ERROR(TRE=1)?
                                RMCS1
                                C17
                                ROL     (SP)+
                                BMI     3$              ;BR IF YES
                                TST     SAVEFG          ;SAVE THE RH/RM REGISTERS?
```

```
042072  100002                          BPL     1$              ;BR IF NO
042074  004737  044310                  JSR     PC,SVRH70       ;YES--SAVE THE REGISTERS
042100                          1$:
042100  004737  045144                  JSR     PC,GETREQ       ;GET DPB POINTER
042104  005702                          TST     R2              ;ENTRY FOR DRIVE ?
042106  001403                          BEQ     2$              ;BR IF NOT
042110  004737  040146                  JSR     PC,OPT          ;CALL OPTIMIZER
042114  000417                          BR      SC              ;CHECK OTHER DRIVES
                                ;THE RELEASE DRIVE COMMAND IS FORECD TO ENTER FOR DUAL PORT OPERATION
042116  012714  000113          2$:     MOV     #113,(R4)       ;RELEASE THE DRIVE
042122  000414                          BR      SC              ;CHECK FOR OTHER DRIVES
042124  052762  100100  000016  3$:     BIS     #BIT15!BIT06,16(R2)     ;SET DATA ERROR FLAG
042132  004737  045050                  JSR     PC,EMPTYQ       ;EMPTY THE "DRIVE'S WAIT" QUEUE
042136  004737  044310                  JSR     PC,SVRH70       ;SAVE THE RH/RM REGISTERS
042142  012714  040111                  MOV     #40111,(R4)     ;ISSUE A "DRIVE CLEAR"
042146  012714  000113                  MOV     #113,(R4)       ;ISSUE A RELEASE TO THE DRIVE
042152  000400                          BR      SC              ;CHECK FOR OTHER DRIVES


                                ;SPECIAL CONDITION ROUTINE

042154  116403  000016          SC:     MOVB    RMAS(R4),R3     ;READ "RMAS"
042160  001014                          BNE     2$              ;BR IF ANY 'ATA' BITS SET
042162  004037  043736                  JSR     R0,RD.RM        ;READ CONTROL AND STATUS REGISTER
042166  000000                          RMCS1
042170  041326                          CI8
                                ;       1$                      ;EXIT IF FAIL TO READ
042172  106126                          ROLB    (SP)+           ;IS "IE"=1?
042174  100405                          BMI     1$              ;YES, NO DRIVES TO CHECK
042176  004037  045234                  JSR     R0,ES.SAV       ;SAVE THE ADDRESS IN '$ESCAPE'
042202  104001                          EMT     1               ;REPORT AN ILLEGAL INTERRUPT
042204  004737  044426                  JSR     PC,SET.IE       ;SET INTERRUPT ENABLE
042210  000207                  1$:     RTS     PC              ;RETURN
042212  005046                  2$:     CLR     -(SP)           ;PROCESS ALL DRIVES THAT HAVE
042214  110316                          MOVB    R3,(SP)         ;AN "ATA"=1
042216  012703  000001                  MOV     #1,R3
042222  005001                          CLR     R1
042224  030316                  SC3:    BIT     R3,(SP)         ;ATA=1?
042226  001005                          BNE     SC5             ;YES
042230  005201                  SC4:    INC     R1              ;MOVE TO THE NEXT DRIVE
042232  106303                          ASLB    R3
042234  001373                          BNE     SC3             ;BR IF MORE TO CHECK?
042236  005726                          TST     (SP)+           ;CLEAN OFF THE STACK
042240  000207                          RTS     PC              ;RETURN TO USER
042242  105761  036760          SC5:    TSTB    DPINT(R1)       ;INITIALIZING THE DRIVE ?
042246  001402                          BEQ     1$              ;BR IF NOT
042250  000137  043166                  JMP     SC13            ;PROCESS THE DRIVE
042254  105761  036770          1$:     TSTB    DPRQS(R1)       ;PORT REQUEST OUTSTANDING ?
042260  001402                          BEQ     2$              ;BR IF NOT
042262  000137  043166                  JMP     SC13            ;START THE OUTSTANDING COMMAND
042266  105761  036740          2$:     TSTB    DRVSTA(R1)      ;CHECK THE DRIVE STATUS
042272  003023                          BGT     5$              ;BR IF ONLINE
042274  105761  037006                  TSTB    ULDFLG(R1)      ;UNLOAD IN PROGRESS?
042300  003420                          BLE     5$              ;BR IF NOT
042302  004737  045144                  JSR     PC,GETREQ       ;GET DPB POINTER
042306  004737  044310                  JSR     PC,SVRH70       ;SAVE THE RH/RM REGISTERS
```

```
                                                JSR     PC,SC12         ;SAVE RMDS, RMER1, RMER2, AND RMMR2
                                                                        ;ALSO DO A DRIVE INIT (DRVINT)
                                                TSTB    DRVSTA(R1)      ;DID DRIVE COME ONLINE?
                                                BLE     6$              ;NO
                                                BIT     #BIT14,RMERRS   ;WAS THERE AN ERROR?
                                                BNE     3$              ;BR IF ERROR
                                                JMP     SC11            ;NO ERROR
                                        3$:     MOV     RMERRS+2,R5     ;YES -- PICKUP RMER1 AND
                                                BR      SC6A            ;GO PROCESS THE ERROR
                                        5$:     TSTB    DRVACT(R1)      ;DRIVE ACTIVE WITH COMMAND OR ERROR RECOVERY ?
                                                BNE     SC6             ;BR IF EITHER
                                                JSR     PC,SC12         ;SAVE RMDS, RMER1, RMER2, AND RMMR2
                                                                        ;ALSO DO A DRVINT
                                        6$:     TSTB    DPINT(R1)       ;TRYING TO INIT THE DRIVE ?
                                                BNE     SC4             ;BR IF YES, CHECK ON MORE DRIVES
                                                TSTB    DRVSTA(R1)      ;CHECK ON DRIVE'S STATUS
                                                BMI     7$              ;BR IF UNSAFE
                                                BIT     #BIT13,RMERRS+4 ;ADDRESS PLUG CHANGED ?
                                                BNE     8$              ;BR IF YES
                                        ;       MOV     #113,-(SP)      ;RELEASE COMMAND
                                                MOV     #111,-(SP)      ;DRIVE CLEAR
                                                JSR     RO,WRT.RM       ;WRITE THE COMMAND INTO RMCS1
                                                RMCS1                   ;REGISTER INDEX
                                                SC8                     ;PARITY EXIT ADDRESS
                                        7$:     MOV     (SP),R5         ;PICKUP (RMAS) BEFORE THE ERROR CALL
                                                JSR     RO,ES.SAV       ;SAVE THE ADDRESS IN '$ESCAPE'
                                                EMT     2               ;REPORT THE UNEXPECTED ATTENTION
                                                BR      SC4             ;GO CHECK FOR MORE ATA'S
                                        8$:
                                                JSR     RO,ES.SAV       ;SAVE THE ADDRESS IN '$ESCAPE'
                                                EMT     5               ;REPORT THE ADDRESS PLUG CHANGE
                                                BR      SC4             ;CHECK FOR MORE DRIVES
                                        SC6:    ASL     R1              ;SETUP TO ADDRESS WORDS
                                                MOV     #-1,TIMER(R1)   ;STOP THE TIMER
                                                ASR     R1              ;RESTORE THE DRIVE ADDRESS
                                                JSR     PC,GETREQ       ;GET THE DPB POINTER FROM THE QUEUE
                                                MOV     R1,RMCS2(R4)    ;SELECT DRIVE
                                                JMP     SC11            ;PROCESS THE SEARCH
                                                JSR     RO,RD.RM        ;READ THE RM'S STATUS REG.
                                                RMDS
                                                SC8
                                                MOV     (SP),R5         ;AND PUT IT IN R5
                                                ROL     (SP)+           ;WAS THERE AN ERROR?
                                                BMI     1$              ;BR IF ERROR
                                                TSTB    DRVACT(R1)      ;CHECK DRIVE'S STATE
                                                BGT     SC11            ;BR IF DRIVE ACTIVE WITH ORDER
                                                BIS     #BIT15!BIT07!BIT03,16(R2)       ;INFORM USER OF ERROR RECOVER COMPLETION
                                                BR      SC7
                                        1$:     JSR     RO,RD.RM        ;READ ERROR REGISTER #1
                                                RMER1
                                                SC8
                                                MOV     (SP)+,R5        ;AND SAVE IT IN R5
                                                JSR     PC,SVRH70       ;SAVE RH/RM REGISTERS
                                                MOV     #111,-(SP)      ;ISSUE A DRIVE CLEAR
                                                JSR     RO,WRT.RM
                                                RMCS1
                                                SC8
```

```
                                        SC6A:   ROL     R5              ;WAS "UNSAFE" CONDITION =1?
                                                BMI     1$              ;BR IF YES
                                                TST     R2              ;ANYTHING IN QUEUE ?
                                                BEQ     SC7             ;BR IF NOT
                                                BIS     #BIT15!BIT07!BIT05,16(R2)       ;INFORM USER OF ERROR
                                                BR      SC7
                                        1$:     JSR     R0,RD.RM        ;READ DRIVE STATUS REG. #1
                                                RMDS
                                                SC8
                                                MOV     (SP),R5         ;SAVE RMDS IN R5
                                                ROL     (SP)+           ;"ERR"=1?
                                                BPL     2$              ;BR IF NO--UNSAFE CLEARED
                                                MOVB    #-1,DRVSTA(R1)  ;DRIVE IS UNSAFE
                                                JSR     PC,SVRH70       ;SAVE RH/RM REGISTERS
                                                BIS     #BIT15!BIT12,16(R2)     ;INFORM USER OF UNSAFE ERROR
                                                BR      SC7
                                        2$:     BIT     #BIT12,R5       ;"MOL" = 1 ?
                                                BNE     3$              ;BR IF YES
                                                MOVB    #-1,DRVACT(R1)  ;ACTIVE ERROR RECOVER
                                                MOVB    #1,DRVSTA(R1)   ;ONLINE
                                                ASL     R1
                                                MOV     #30000.,TIMER(R1)       ;START 30 SECOND TIMER
                                                ASR     R1
                                                JMP     SC4
                                        3$:     BIS     #BIT15!BIT07!BIT04,16(R2)       ;INFORM USER OF ERROR
                                        SC7:    CLRB    DRVACT(R1)      ;DRIVE IS IDLE
                                        ;       JSR     PC,EMPTYQ       ;DUMP THE QUEUE
                                                JSR     PC,POPQUE       ;REMOVE THE QUEUE
                                                TSTB    ULDFLG(R1)      ;UNLOAD IN RMOGRESS OR QUEUE?
                                                BGT     1$              ;BR IF NOT
                                                CLRB    ULDFLG(R1)      ;CLEAR UNLOAD FLAG
                                        1$:     MOVB    ATABIT(R1),RMAS(R4)     ;CLEAR ATTENTION BIT
                                                TSTB    DRVSTA(R1)      ;IS THE DRIVE UNSAFE ?
                                                BMI     2$              ;BR IF IT IS
                                        ;       MOV     #113,-(SP)      ;RELEASE COMMAND
                                                MOV     #111,-(SP)      ;DRIVE CLEAR COMMAND
                                                JSR     R0,WRT.RM       ;WRITE THE COMMAND INTO RPCS1
                                                RMCS1                   ;REGISTER INDEX
                                                SC8                     ;PARITY EXIT ADDRESS
                                        2$:     JMP     SC4             ;CHECK FOR MORE DRIVES
                                        SC8:    TSTB    DRVACT(R1)      ;IS DRIVE IDLE?
                                                BEQ     1$              ;YES
                                                JSR     PC,GETREQ       ;GET DPB POINTER
                                                JSR     PC,C17          ;PROCESS THE PARITY ERROR
                                                BR      2$              ;CONTINUE
                                        1$:
                                        ;       JSR     PC,C17          ;PROCESS THE PARITY ERROR
                                                JSR     PC,C17B         ;PROCESS THE UNCORRECTABLE PARITY ERROR
                                        2$:     JMP     SC4             ;CHECK MORE DRIVES
                                        SC11:   TSTB    ULDFLG(R1)      ;"UNLOAD IN PROGRESS"?
                                                BLE     1$              ;BR IF NO
                                                CLRB    ULDFLG(R1)      ;CLEAR UNLOAD FLAG
                                        1$:     CLRB    DRVACT(R1)      ;SET DRIVE IDLE
                                                BITB    ATABIT(R1),SRCHWT       ;DOING A SEARCH OPERATION FOR
                                                                        ;AN I/O COMMAND?
                                                BNE     2$              ;BR IF YES
                                                JSR     PC,POPQUE       ;REMOVE REQUEST FROM QUEUE
```

```
                                        BIS     #BIT07,16(R2)   ;SET "DONE" BIT
                                        TST     SAVEFG          ;SAVE THE REGISTERS?
                                        BPL     2$              ;BR IF NO
                                        JSR     PC,SVRH70       ;YES--SAVE ALL OF THE RH/RM REG'S
                                2$:     MOVB    ATABIT(R1),RMAS(R4)     ;CLEAR ATTENTION BIT
                                        BICB    ATABIT(R1),SRCHWT       ;CLEAR IMPLIED SEEK SET
                                        ASL     R1              ;WORD INDEX
                                        MOV     #-1,TIMER(R1)   ;STOP CLOCK
                                        ASR     R1              ;RESTORE R1
                                        JSR     PC,OPT          ;START A REQUEST
                                        JMP     SC4             ;CHECK FOR MORE DRIVES
                                SC12:   MOV     R1,RMCS2(R4)    ;SELECT DRIVE
                                        MOV     RMDS(R4),RMERRS ;SAVE THE FOUR REGISTERS THAT
                                        MOV     RMER1(R4),RMERRS+2      ;WILL TELL US SOMETHING
                                        MOV     RMER2(R4),RMERRS+4
                                        MOV     RMMR2(R4),RMERRS+6
                                        JSR     R0,DRVINT       ;INIT. THE STATE OF THE DRIVE
                                        BR      1$              ;TAKE ERROR EXIT
                                        RTS     PC              ;RETURN
                                1$:     TST     (SP)+           ;POP PC OFF OF THE STACK
                                        BR      SC8             ;PROCESS THE PARITY ERROR
                                SC13:   ASL     R1              ;SETUP TO ADDRESS WORDS
                                        MOV     #-1,TIMER(R1)   ;STOP THE TIMER
                                        ASR     R1              ;
                                        MOV     R1,RMCS2(R4)    ;SELECT THE DRIVE
                                        MOVB    ATABIT(R1),RMAS(R4)     ;CLEAR THE ATTENTION BIT
                                1$:     TSTB    DPINT(R1)       ;INITIALIZING THE DRIVE ?
                                        BEQ     2$              ;BR IF NOT
                                        CLRB    DPINT(R1)       ;CLEAR THE INIT INDICATOR
                                        JSR     R0,DRVINT       ;GO INIT THE DRIVE
                                        NOP                     ;DUMMY PARITY ERROR RETURN
                                        TSTB    DRVSTA(R1)      ;DRIVE ONLINE ?
                                        BGT     2$              ;BR IF YES -- START ORDER
                                        TST     R2              ;QUEUE ENTRY FOR THE DRIVE
                                        BEQ     3$              ;BR IF NOT
                                        JSR     PC,GETREQ       ;GET DPB ADDRESS
                                        BIS     #BIT15!BIT14,16(R2)     ;INFORM USER THAT DRIVE OFFLINE
                                        JSR     PC,SVRH70       ;SAVE THE REGISTERS
                                ;       JSR     PC,EMPTYQ       ;EMPTY THE REQUEST QUEUE
                                        JSR     PC,POPQUE       ;REMOVE THE QUEUE
                                        BR      3$
                                2$:     BIT     #BIT11,RMCS1(R4)        ;DVA SET ?
                                        BNE     4$              ;SET THEN CALL OPT
                                        ASL     R1
                                        MOV     #60000,TIMER(R1)
                                        ASR     R1
                                        BR      3$
                                4$:     JSR     PC,OPT          ;START THE PENDING REQUEST
                                3$:     JMP     SC4             ;PROCESS OTHER DRIVES

                                ;RM TIMER ROUTINE
                                ;CALL
                                ;       MOV     #TIME,-(SP)     ;ELASPED TIME IN MILLISECONDS ON THE STACK
                                ;       JSR     PC,RMTMR        ;CALL RM05 TIME ROUTINE

                                RMTMR:  TST     ACTDRV          ;CHECK "ACTDRV & ACTSTR"
                                        BNE     4$              ;IF NON ZERO EXIT
```

```
1011 043332  112737  000001  037005          MOVB    #1,ACTSTR       ;SET "ACTSTR"
1012 043340  104412                          SAVREG                  ;SAVE R0 - R5
1013 043342  005001                          CLR     R1              ;START WITH DRIVE 0
1014 043344  005003                          CLR     R3
1015 043346  005763  037032          1$:     TST     TIMER(R3)       ;IS THE TIMER RUNNING?
1016 043352  002406                          BLT     2$              ;BR IF NO
1017 043354  166663  000002  037032          SUB     2(SP),TIMER(R3) ;COUNT THE INTERVAL
1018 043362  003002                          BGT     2$              ;BR IF NO SOFTWARE TIMEOUT
1019 043364  004737  043414                  JSR     PC,STO          ;CALL SOFTWARE TIMEOUT ROUTINE
1020 043370  005201                  2$:     INC     R1              ;MOVE TO NEXT DRIVE
1021 043372  005723                          TST     (R3)+
1022 043374  022701  000010                  CMP     #8.,R1          ;OUT OF DRIVES?
1023 043400  003362                          BGT     1$              ;BR IF NO
1024 043402  104413                  3$:     RESREG                  ;RESTORE R0 - R5
1025 043404  105037  037005                  CLRB    ACTSTR          ;ZERO ACTIVE SOFTWARE TIMEOUT ROUTINE FLAG
1026 043410  012616                  4$:     MOV     (SP)+,(SP)      ;ADJUST THE STACK
1027 043412  000207                          RTS     PC              ;RETURN
1028
1029                                 ;SOFTWAPE TIMEOUT ROUTINE
1030                                 ;
1031                                 ;NOTE: THIS ROUTINE MUST BE ENTERED AT PRIORITY 6
1032                                 ;       OR GREATER
1033                                 ;
1034                                 ;CALL:  STO
1035                                 ;       MOV     #DRVNUM,R1      ;DRIVE NUMBER
1036                                 ;       JSR     PC,STO          ;CALL
1037                                 ;       RETURN
1038
1039 043414  010146                  STO:    MOV     R1,-(SP)        ;SAVE R1
1040 043416  010246                          MOV     R2,-(SP)        ;SAVE R2
1041 043420  010346                          MOV     R3,-(SP)        ;SAVE R3
1042 043422  010446                          MOV     R4,-(SP)        ;SAVE R4
1043 043424  013704  037066                  MOV     RMADR,R4        ;GET ADDRESS OF "RMCS1"
1044 043430  010164  000010                  MOV     R1,RMCS2(R4)    ;SELECT THE DRIVE
1045 043434  004037  043736                  JSR     R0,RD.RM        ;READ "DRIVE STATUS REG"
1046 043440  000012                          RMDS
1047                                 ;       STO5
1048 043442  043724                          STO9
1049 043444  105726                          TSTB    (SP)+           ;IS "DRY"=1?
1050 043446  100436                          BMI     STO2            ;BR IF YES
1051 043450  105761  036760          STO1:   TSTB    DPINT(R1)       ;TRYING TO INTIALIZE THE DRIVE ?
1052 043454  001033                          BNE     STO2            ;BR IF YES
1053 043456  105761  036770                  TSTB    DPRQS(R1)       ;OUTSTANDING PORT REQUEST FOR THE DRIVE ?
1054 043462  001030                          BNE     STO2            ;BR IF YES
1055 043464  013702  037000                  MOV     TRNSWT,R2       ;PICKUP TRANSFER WAIT QUEUE
1056 043470  020137  037052                  CMP     R1,DTUW         ;TRANSFER UNDERWAY ON THIS DRIVE?
1057 043474  001404                          BEQ     1$              ;BR IF YES
1058 043476  000137  043724                  JMP     STO9            ;IF NOT DON'T BOTHER DRIVES
1059 043502  004737  045144                  JSR     PC,GETREQ       ;GET DPB ADDRESS
1060 043506  052762  101000  000016  1$:     BIS     #BIT15!BIT09,16(R2)     ;SET THE ERROR FLAGS
1061 043514  004737  044310                  JSR     PC,SVRH70       ;SAVE RH/RM REGISTERS
1062                                 ;       MOV     #CLR,RMCS2(R4)  ;"INIT" THE MASS BUS
1063 043520  105061  036730                  CLRB    DRVACT(R1)      ;DRIVE IS IDLE
1064 043524  105061  037006                  CLRB    ULDFLG(R1)      ;CLEAR THE UNLOAD FLAG
1065 043530  005037  037000                  CLR     TRNSWT          ;CLEAR DPB ADDRESS
1066 043534  012737  177777  037052          MOV     #-1,DTUW        ;CLEAR THE TRANSFER DRIVE #
1067 043542  000470                          BR      STO9            ;DON'T BOTHER OTHER DRIVES
```

```
1068 043544  116405  000016          ST02:   MOVB    RMAS(R4),R5     ;READ ATTENTION REG
1069 043550  136105  037054                  BITB    ATABIT(R1),R5   ;IS ATTENTION FOR THIS DRIVE UP ?
1070 043554  001007                          BNE     ST03            ;YES
1071 043556  105761  036760                  TSTB    DPINT(R1)       ;TRYING TO INTIALIZE THE DRIVE ?
1072 043562  001021                          BNE     ST06            ;BR IF YES - DRIVE NOT ONLINE
1073 043564  105761  036770                  TSTB    DPRQS(R1)       ;OUTSTANDING PORT REQUEST FOR THE DRIVE ?
1074 043570  001035                          BNE     ST07            ;BR IF YES - NO RESPONSE TO REQUEST
1075 043572  000454                          BR      ST09            ;OTHER WISE EXIT
1076 043574  105761  036760          ST03:   TSTB    DPINT(R1)       ;INITIALIZING THE DRIVE ?
1077 043600  001003                          BNE     1$              ;BR IF INIT PENDING
1078 043602  105761  036770                  TSTB    DPRQS(R1)       ;PORT REQUEST PENDING ?
1079 043606  001446                          BEQ     ST09            ;BR IF NOT
1080 043610  012763  177777  037032  1$:     MOV     #-1,TIMER(R3)   ;STOP THE TIMER
1081 043616  000442                          BR      ST09            ;EXIT
1082 043620  004737  041326          ST05:   JSR     PC,C18          ;GO HANDLE THE PARITY ERROR
1083 043624  000437                          BR      ST09
1084 043626  105061  036760          ST06:   CLRB    DPINT(R1)       ;CLEAR THE INITIALIZE INDICATOR
1085 043632  105061  036740                  CLRB    DRVSTA(R1)      ;SET UNIT OFFLINE
1086 043636  012763  177777  037032          MOV     #-1,TIMER(R3)   ;STOP THE TIMER
1087 043644  004737  045144                  JSR     PC,GETREQ       ;GET THE DPB ADDRESS
1088 043650  005702                          TST     R2              ;REQUEST IN QUEUE ?
1089 043652  001424                          BEQ     ST09            ;BR IF NOT
1090 043654  052762  140000  000016          BIS     #BIT15!BIT14,16(R2)     ;INFORM THE USER DRIVE NOT AVAILABLE
1091 043662  000414                          BR      ST08            ;FINISH
1092 043664  012763  177777  037032  ST07:   MOV     #-1,TIMER(R3)   ;STOP THE TIMER
1093 043672  105061  036770                  CLRB    DPRQS(R1)       ;CLEAR PORT REQUEST INDICATOR
1094 043676  004737  045144                  JSR     PC,GETREQ       ;GET DPB ADDRESS
1095 043702  005702                          TST     R2              ;QUEUE ENTRY FOR DRIVE ?
1096 043704  001407                          BEQ     ST09            ;BR IF NONE
1097 043706  012762  100004  000016          MOV     #BIT15!BIT2,16(R2)      ;INFORM USER OF PORT REQUEST ERROR
1098 043714  004737  045050          ST08:   JSR     PC,EMPTYQ       ;CLEAR THE QUEUE FOR THE DRIVE
1099 043720  004737  044310                  JSR     PC,SVRH70       ;SAVE THE REGISTERS
1100 043724  012604                  ST09:   MOV     (SP)+,R4        ;RESTORE R4
1101 043726  012603                          MOV     (SP)+,R3        ;RESTORE R3
1102 043730  012602                          MOV     (SP)+,R2        ;RESTORE R2
1103 043732  012601                          MOV     (SP)+,R1        ;RESTORE R1
1104 043734  000207                          RTS     PC              ;RETURN
1105
1106                                 ;ROUTINE TO READ A RH/RM REGISTER
1107                                 ;
1108                                 ;CALL
1109                                 ;       JSR     R0,RD.RM        ;GO READ A REGISTER
1110                                 ;       INDEX                   ;REG. INDEX FROM BASE
1111                                 ;       ERRADR                  ;ERROR ADDRESS--PROCESS ERROR STARTING
1112                                 ;                               ;AT THIS ADDRESS
1113                                 ;       RETURN                  ;CONTENTS OF REG. IS ON THE STACK
1114                                 ;
1115 043736  013737  037064  044104  RD.RM:   MOV     MCPEMX,RD.RM2   ;MAX. RETRYS ALLOWED
1116 043744  011646                          MOV     (SP),-(SP)      ;SAVE R0 FOR RETURN
1117 043746  013737  037066  043762          MOV     RMADR,RD.ADR    ;FORM THE DESIRED ADDRESS
1118 043754  062037  043762                  ADD     (R0)+,RD.ADR    ;USING THE BASE AND THE INDEX
1119 043760  013727                  RD.RM1: MOV     @(PC)+,(PC)+    ;READ THE DESIRED REGISTER OF THE RM DRIVE
1120 043762  000000                  RD.ADR: .WORD   0               ;ADDRESS IS FORMED HERE
1121 043764  000000                  RD.WRD: .WORD   0               ;REG. CONTENTS PUT HERE
1122 043766  013766  043764  000002          MOV     RD.WRD,2(SP)    ;RETURN IT TO THE USER
1123 043774  013746  037066                  MOV     RMADR,-(SP)     ;PUT THE ADDRESS ON THE STACK
1124 044000  062716  000010                  ADD     #RMCS2,(SP)     ;FORM THE ADDRESS OF RMCS2
```

```
1125 044004  032736  010000            BIT     #BIT12,@(SP)+   ;CHECK THE 'NED' BIT
1126 044010  001037                    BNE     RD.RM3          ;BR IF DRIVE NON-EXISTENT
1127 044012  017746  173050            MOV     @RMADR,-(SP)    ;READ RMCS1
1128 044016  032716  020000            BIT     #BIT13,(SP)     ;DID MCPE SET?
1129 044022  001002                    BNE     1$              ;BR IF YES
1130 044024  022620                    CMP     (SP)+,(R0)+     ;ADJUST FOR RETURN
1131 044026  000432                    BR      RD.RM4          ;EXIT
1132 044030                    1$:
     044030  004037  045234            JSR     R0,ES.SAV       ;SAVE THE ADDRESS IN '$ESCAPE'
     044034  104003                    EMT     3               ;REPORT "MCPE" ERROR
1133 044036  005737  037052            TST     DTUW            ;DATA TRANSFER UNDERWAY?
1134 044042  100405                    BMI     2$              ;NO
1135 044044  032716  040000            BIT     #BIT14,(SP)     ;"TRE" = 1 ?
1136 044050  001402                    BEQ     2$              ;NO
1137 044052  005726                    TST     (SP)+           ;YES--CLEAN OFF THE STACK AND
1138 044054  000415                    BR      RD.RM3          ;TAKE THE FATAL ERROR EXIT
1139 044056  052716  040000    2$:     BIS     #BIT14,(SP)     ;CLEAR "MCPE" BY SENDING A "1" TO "TRE"
1140 044062  000316                    SWAB    (SP)            ;POSITION BEFORE WRITING
1141 044064  013737  037066  044100    MOV     RMADR,3$        ;FORM ADDRESS OF HIGH BYTE
1142 044072  005237  044100            INC     3$
1143 044076  112637                    MOVB    (SP)+,@(PC)+    ;WRITE THE HIGH BYTE OF RMCS1
1144 044100  000000            3$:     .WORD   0               ;ADDRESS STORAGE
1145 044102  005327                    DEC     (PC)+           ;EXCEEDED MAX. RETRYS
1146 044104  000003            RD.RM2: .WORD   3
1147 044106  002324                    BGE     RD.RM1          ;BR IF NO
1148 044110  011000            RD.RM3: MOV     (R0),R0         ;FATAL ERROR EXIT
1149 044112  012616                    MOV     (SP)+,(SP)
1150 044114  000200            RD.RM4: RTS     R0
1151
1152                           ;ROUTINE TO WRITE A REGISTER
1153                           ;
1154                           ;CALL
1155                           ;       MOV     DATA,-(SP)      ;DATA TO BE LOADED ON THE STACK
1156                           ;       JSR     R0,WRT.RM       ;CALL THE ROUTINE TO LOAD(WRITE) THE REG.
1157                           ;       INDEX                   ;INDEX OF THE REGISTER TO BE LOADED
1158                           ;       ERRADR                  ;ADDRESS TO RETURN TO ON AN ERROR
1159                           ;       RETURN                  ;ERROR FREE RETURN
1160
1161 044116  013737  037064  044274  WRT.RM: MOV     MCPEMX,WRT.R2   ;MAX RETRYS ALLOWED
1162 044124  016637  000002  044204          MOV     2(SP),WRT.WD    ;SAVE THE WORD TO WRITE
1163 044132  012616                    MOV     (SP)+,(SP)      ;ADJUST THE STACK
1164 044134  012037  044206            MOV     (R0)+,WRT.AD    ;GET INDEX OF REGISTER TO BE WRITTEN
1165 044140  001015                    BNE     1$              ;BR IF NOT RMCS1
1166 044142  122737  000150  044204    CMPB    #150,WRT.WD     ;IS THE COMMAND FOR DATA TRANSFERS?
1167 044150  002411                    BLT     1$              ;YES--DON'T GET THE OLD A16 & A17, & PSEL
1168 044152  004037  043736            JSR     R0,RD.RM        ;NO---COMBINE A16&A17, & PSEL WITH
1169 044156  000000                    RMCS1                   ;THE COMMAND BEFORE SENDING IT TO
1170 044160  044300                    WRT.R3                  ;THE RH/RM
1171 044162  000316                    SWAB    (SP)
1172 044164  042716  177770            BIC     #^C7,(SP)
1173 044170  112637  044205            MOVB    (SP)+,WRT.WD+1
1174 044174  063737  037066  044206  1$:     ADD     RMADR,WRT.AD    ;FORM THE ADDRESS OF THE DISK REG.
1175 044202  012737                    WRT.R1: MOV     (PC)+,@(PC)+    ;LOAD THE DESIRED REG.
1176 044204  000000                    WRT.WD: .WORD   0               ;WORD TO WRITE GOES HERE
1177 044206  000000                    WRT.AD: .WORD   0               ;ADDRESS IS FORMED HERE
1178 044210  013746  037066                    MOV     RMADR,-(SP)     ;PUT THE ADDRESS ON THE STACK
1179 044214  062716  000010                    ADD     #RMCS2,(SP)     ;FORM THE ADDRESS OF RMCS2
```

```
1180 044220  032736  010000                    BIT     #BIT12,@(SP)+   ;CHECK THE 'NED' BIT
1181 044224  001025                            BNE     WRT.R3          ;BR IF DRIVE NON-EXISTENT
1182 044226  004037  043736                    JSR     R0,RD.RM        ;CHECK FOR PARITY ERROR ON WRITE
1183 044232  000014                            RMER1
1184 044234  044300                            WRT.R3
1185 044236  032726  000010                    BIT     #BIT03,(SP)+
1186 044242  001420                            BEQ     WRT.R4          ;BR IF "PAR=0"
1187 044244  016037  177776  044256            MOV     -2(R0),1$       ;PICKUP THE INDEX
1188 044252  004037  043736                    JSR     R0,RD.RM        ;READ THE REG.
1189 044256  000000                    1$:     .WORD   0               ;REG. INDEX
1190 044260  044300                            WRT.R3                  ;RETURN TO THIS ADDRESS ON ERROR
1191 044262  004037  045234                    JSR     R0,ES.SAV       ;SAVE THE ADDRESS IN '$ESCAPE'
     044266  104004                            EMT     4               ;REPORT THE PARITY ON WRITE ERROR
1192 044270  005726                            TST     (SP)+           ;CLEAR OFF THE STACK
1193 044272  005327                            DEC     (PC)+           ;DECREMENT THE ERROR COUNT
1194 044274  000003                    WRT.R2: .WORD   3               ;RETRY COUNTER
1195 044276  002341                            BGE     WRT.R1          ;TRY AGAIN IF NOT FINISHED
1196 044300  011000                    WRT.R3: MOV     (R0),R0         ;TAKE THE "PARITY ON WRITE" ERROR EXIT
1197 044302  000401                            BR      WRT.R5          ;EXIT
1198 044304  005720                    WRT.R4: TST     (R0)+           ;ADJUST FOR ERROR FREE EXIT
1199 044306  000200                    WRT.R5: RTS     R0
1200
1201                                   ;ROUTINE TO SAVE THE RH/RM REGISTERS AS PER DPB+14
1202                                   ;
1203                                   ;CALL
1204                                   ;       MOV     #DPBNUM,R2      ;DPB POINTER TO R2
1205                                   ;       JSR     PC,SVRH70       ;SAVE THE DRIVES REG'S
1206
1207 044310  104412                    SVRH70: SAVREG                  ;SAVE R0 - R5
1208 044312  005702                            TST     R2              ;QUEUE ENTRY FOR THE DRIVE ?
1209 044314  001442                            BEQ     6$              ;BR IF NONE
1210 044316  013704  037066                    MOV     RMADR,R4
1211 044322  111264  000010                    MOVB    (R2),RMCS2(R4)  ;SELECT DRIVE
1212 044326  016203  000014                    MOV     14(R2),R3       ;GET THE ERROR TABLE POINTER
1213 044332  001433                            BEQ     6$              ;EXIT IF NO ADDRESS
1214 044334  005037  044370                    CLR     3$              ;COUNTER & POINTER
1215 044340  023727  044370  000022    1$:     CMP     3$,#RMDB        ;REACHED THE BUFFER REGISTER ?
1216 044346  001006                            BNE     2$              ;BR IF NOT
1217 044350  032764  000200  000010            BIT     #BIT07,RMCS2(R4)        ;'OR' SET ?
1218 044356  001002                            BNE     2$              ;BR IF SET
1219 044360  005023                            CLR     (R3)+           ;STORE RMDB AS ZEROES
1220 044362  000405                            BR      4$              ;CONTINUE
1221 044364  004037  043736            2$:     JSR     R0,RD.RM        ;READ THE SELECTED REGISTER
1222 044370  000000                    3$:     .WORD   0               ;REGISTER INDEX
1223 044372  044416                            5$                      ;ERROR RETURN ADDRESS
1224 044374  012623                            MOV     (SP)+,(R3)+     ;STORE THE REGISTER CONTENTS
1225 044376  023727  044370  000046    4$:     CMP     3$,#RMEC2       ;REACHED THE END ?
1226 044404  001406                            BEQ     6$              ;BR IF YES
1227 044406  062737  000002  044370            ADD     #2,3$           ;INCREMENT THE REGISTER INDEX
1228 044414  000751                            BR      1$              ;CONTINUE READING THE REGISTERS
1229 044416  004737  041226            5$:     JSR     PC,CI7          ;PROCESS THE UNCORRECTABLE PARITY ERROR
1230 044422  104413                    6$:     RESREG                  ;RESTORE R0 - R5
1231 044424  000207                            RTS     PC              ;RETURN
1232
1233                                   ;ROUTINE TO SET THE INTERRUPT WITHOUT GETTING A "TRE"
1234                                   ;CALL
1235                                   ;       MOV     #DRVNUM,R1      ;DRIVE NUMBER TO R1
```

```
                                        ;       JSR     PC,SET.IE       ;SET "IE"
                                        ;       RETURN
                                        SET.IE: MOV     R4,-(SP)        ;SAVE R4
                                                MOV     RMADR,R4        ;PICKUP ADDRESS OF RMCS1
                                                MOV     R1,RMCS2(R4)    ;SELECT DRIVE
                                                MOV     (R4),-(SP)      ;READ RMCS1
                                                BIS     #BIT14,(SP)     ;SET THE "TRE" BIT OF THE WORD READ
                                                SWAB    (SP)            ;ADJUST FOR DATO
                                                MOVB    #BIT06,(R4)     ;SET "IE"
                                                BIT     #BIT12,RMCS2(R4)        ;IS "NED"=1?
                                                BNE     1$              ;YES--CLEAR "TRE"
                                                TST     (SP)+           ;CLEAN OFF THE STACK
                                                BR      2$
                                        1$:     MOVB    (SP)+,1(R4)     ;CLEAR "TRE"
                                        2$:     MOV     (SP)+,R4        ;RESTORE R4
                                                RTS     PC              ;RETURN TO CALLER

                                        ;QUEUE COUNT
                                        QCNT:   .BYTE   0               ;DRIVE 0
                                                .BYTE   0               ;DRIVE 1
                                                .BYTE   0               ;DRIVE 2
                                                .BYTE   0               ;DRIVE 3
                                                .BYTE   0               ;DRIVE 4
                                                .BYTE   0               ;DRIVE 5
                                                .BYTE   0               ;DRIVE 6
                                                .BYTE   0               ;DRIVE 7

                                        ;QUEUE INPUT POINTERS

                                        QINPT:  .WORD   QDRV0           ;DRIVE 0
                                                .WORD   QDRV1           ;DRIVE 1
                                                .WORD   QDRV2           ;DRIVE 2
                                                .WORD   QDRV3           ;DRIVE 3
                                                .WORD   QDRV4           ;DRIVE 4
                                                .WORD   QDRV5           ;DRIVE 5
                                                .WORD   QDRV6           ;DRIVE 6
                                                .WORD   QDRV7           ;DRIVE 7

                                        ;QUEUE OUTPUT POINTERS

                                        QOUTPT: .WORD   QDRV0           ;DRIVE 0
                                                .WORD   QDRV1           ;DRIVE 1
                                                .WORD   QDRV2           ;DRIVE 2
                                                .WORD   QDRV3           ;DRIVE 3
                                                .WORD   QDRV4           ;DRIVE 4
                                                .WORD   QDRV5           ;DRIVE 5
                                                .WORD   QDRV6           ;DRIVE 6
                                                .WORD   QDRV7           ;DRIVE 7

                                        QSTART: .WORD   QDRV0           ;DRIVE 0 START ADDRESS
                                        QSTOP:  .WORD   QDRV1           ;DRIVE 0 STOP ADDRESS & DRIVE 1 START ADDRESS
                                                .WORD   QDRV2           ;STOP DRIVE 1--START DRIVE 2
                                                .WORD   QDRV3           ;STOP DRIVE 2--START DRIVE 3
                                                .WORD   QDRV4           ;STOP DRIVE 3--START DRIVE 4
                                                .WORD   QDRV5           ;STOP DRIVE 4--START DRIVE 5
                                                .WORD   QDRV6           ;STOP DRIVE 5--START DRIVE 6
```

```
1293 044566  044752                        .WORD   QDRV7           ;STOP DRIVE 6--START DRIVE 7
1294 044570  044772                        .WORD   QTERM           ;STOP DRIVE 7
1295
1296                                ;DRIVE REQUEST QUEUES
1297
1298 044572                         QDRV0:  .BLKW   10
1299 044612                         QDRV1:  .BLKW   10
1300 044632                         QDRV2:  .BLKW   10
1301 044652                         QDRV3:  .BLKW   10
1302 044672                         QDRV4:  .BLKW   10
1303 044712                         QDRV5:  .BLKW   10
1304 044732                         QDRV6:  .BLKW   10
1305 044752                         QDRV7:  .BLKW   10
1306         044772                 QTERM=.
1307
1308                                ;ROUTINE TO CLEAR ALL OF THE REQUEST QUEUES
1309                                ;
1310                                ;CALL
1311                                ;       JSR     PC,CLRQUE
1312
1313 044772  104412                 CLRQUE: SAVREG                  ;SAVE R0 - R5
1314 044774  012702  044500                 MOV     #QCNT,R2        ;ZERO THE QUEUE COUNTS
1315 045000  005022                         CLR     (R2)+           ;DRIVES 0 & 1
1316 045002  005022                         CLR     (R2)+           ;DRIVES 2 & 3
1317 045004  005022                         CLR     (R2)+           ;DRIVES 4 & 5
1318 045006  005022                         CLR     (R2)+           ;DRIVES 6 & 7
1319 045010  012703  000010                 MOV     #8.,R3          ;MOVE THE STARTING
1320 045014  012701  044550                 MOV     #QSTART,R1      ;ADDRESS OF THE QUEUE INTO
1321 045020  012122                 1$:     MOV     (R1)+,(R2)+     ;THE QUEUE INPUT POINTER
1322 045022  005303                         DEC     R3
1323 045024  001375                         BNE     1$
1324 045026  012703  000010                 MOV     #8.,R3          ;MOVE THE STARTING ADDRESS
1325 045032  012701  044550                 MOV     #QSTART,R1      ;OF THE QUEUE INTO THE
1326 045036  012122                 2$:     MOV     (R1)+,(R2)+     ;QUEUE OUTPUT POINTER
1327 045040  005303                         DEC     R3
1328 045042  001375                         BNE     2$
1329 045044  104413                         RESREG                  ;RESTORE R0 - R5
1330 045046  000207                         RTS     PC
1331
1332                                ;EMPTY THE QUEUE SPECIFIED BY R1
1333                                ;
1334                                ;CALL
1335                                ;       MOV     DRVNUM,R1       ;DRIVE NUMBER TO R1
1336                                ;       JSR     PC,EMPTYQ
1337
1338 045050  105061  044500         EMPTYQ: CLRB    QCNT(R1)        ;CLEAR NUMBER OF ITEMS IN QUEUE
1339 045054  006301                         ASL     R1
1340 045056  016161  044510  044530         MOV     QINPT(R1),QOUTPT(R1)    ;SET OUTPUT QUEUE POINTER=INPUT POINTER
1341 045064  006201                         ASR     R1
1342 045066  000207                         RTS     PC
1343
1344                                ;ROUTINE TO PUT A REQUEST IN QUEUE
1345                                ;
1346                                ;CALL
1347                                ;       MOV     #DRVNUM,R1      ;DRIVE NUMBER
1348                                ;       MOV     #DPB,R2         ;ADDRESS OF PARAMETER BLOCK
1349                                ;       JSR     R0,DRVQUE       ;GO PUT REQUEST IN QUEUE
```

```
                                        ;       RETURN1                 ;RETURN HERE IF QUEUE IS FULL
                                        ;       RETURN2                 ;RETURN HERE IF REQUEST IS IN QUEUE
                                        ;
045070  122761  000010  044500  DRVQUE: CMPB    #10,QCNT(R1)            ;IS QUEUE FULL?
045076  001421                          BEQ     2$                      ;BR IF YES-TAKE RETURN1
045100  105261  044500                  INCB    QCNT(R1)                ;INCREMENT QUEUE COUNT
045104  006301                          ASL     R1
045106  010271  044510                  MOV     R2,@QINPT(R1)           ;PUT THIS REQUEST IN QUEUE
045112  062761  000002  044510          ADD     #2,QINPT(R1)            ;UPDATE THE QUEUE POINTER
045120  026161  044510  044552          CMP     QINPT(R1),QSTOP(R1)     ;TIME TO RESET THE POINTER
045126  001003                          BNE     1$                      ;BR IF NO
045130  016161  044550  044510          MOV     QSTART(R1),QINPT(R1)    ;YES--RESET POINTER
045136  006201                  1$:     ASR     R1
045140  005720                          TST     (R0)+                   ;TAKE RETURN 2
045142  000200                  2$:     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
045144  005002                  GETREQ: CLR     R2
045146  105761  044500                  TSTB    QCNT(R1)                ;IS THERE ANY REQUEST IN QUEUE?
045152  001404                          BEQ     2$                      ;NO
045154  006301                  1$:     ASL     R1
045156  017102  044530                  MOV     @QOUTPT(R1),R2          ;PICKUP "DPB" POINTER FOR THIS DRIVE
045162  006201                          ASR     R1
045164  000207                  2$:     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
                                        ;       RETURN                  ;R2=ADDRESS OF DPB REMOVED
045166  105361  044500          POPQUE: DECB    QCNT(R1)                ;DECREMENT QUEUE COUNT
045172  006301                          ASL     R1
045174  017102  044530                  MOV     @QOUTPT(R1),R2          ;GET THE "DPB" POINTER
045200  005021  044530                  CLR     @QOUTPT(R1)             ;REMOVE DPB ADDRESS FROM THE QUEUE
045204  062761  000002  044530          ADD     #2,QOUTPT(R1)           ;UPDATE THE QUEUE POINTER
045212  026161  044530  044552          CMP     QOUTPT(R1),QSTOP(R1)    ;TIME TO RESET THE POINTER?
045220  001003                          BNE     1$                      ;NO--BR TO EXIT
045222  016161  044550  044530          MOV     QSTART(R1),QOUTPT(R1)   ;YES--RESET THE POINTER
045230  006201                  1$:     ASR     R1
045232  000207                          RTS     PC                      ;RETURN TO USER
                                        ;ROUTINE TO SAVE THE CONTENTS OF '$ESCAPE' WHEN THE DRIVER
                                        ;REPORTS AN ERROR DIRECTLY.
                                        ;
                                        ;CALL
                                        ;       JSR     R0,ES.SAV
                                        ;       ERROR   N               ;;THE ERROR CALL
                                        ;       RETURN                  ;THE RETURN IS PAST THE ERROR CALL
```

```
    1408
    1409 045234  012037  045250          ES.SAV: MOV     (R0)+,1$        ;GET THE ERROR CALL
    1410 045240  013746  001222                  MOV     $ESCAPE,-(SP)   ;SAVE THE ADDRESS IN '$ESCAPE'
    1411 045244  005037  001222                  CLR     $ESCAPE         ;CLEAR THE ESCAPE RETURN
    1412 045250  000000                  1$:     .WORD   0               ;THE ERROR CALL IS MOVED HERE
    1413 045252  012637  001222                  MOV     (SP)+,$ESCAPE   ;RESTORE THE ESCAPE ADDRESS
    1414 045256  000200                          RTS     R0              ;RETURN
    1416
```

```
                                .SBTTL  GETADR - GET BUS ADDRESS AND VECTOR ADDRESS
                                ;
                                ;THIS ROUTINE IS USED TO ENSURE THE BUS ADDRESS
                                ;OF THE RH/RM IS SETUP TO READ THE PROPER VALUE.
                                ;IT WILL ALSO READ THE ADDRESS FROM THE TTY IF
                                ;REQUIRED.
                                ;NOTE: THIS ROUTINE DESTROYS R0-R4
                                ;CALL
                                ;
                                ;       JSR     PC,GETADR
                                ;       RETURN
                                ;
045260  005737  001322          GETADR: TST     BUSADR          ;INPUT FROM TTY REQUESTED?
045264  001427                          BEQ     5$              ;NO--BRANCH
045266  005037  001322                  CLR     BUSADR          ;YES--CLEAR THE REQUEST FLAG
045272  012700  001470          1$:     MOV     #RH.ADR,R0      ;FIRST ADDRESS
045276  012703  045436                  MOV     #MRMCS1,R3      ;"RMCS1="
045302  011004                          MOV     (R0),R4         ;PRESENT RMCS1 ADDRESS
045304  004037  034070                  JSR     R0,GETNUM       ;GET NEW RMCS1
045310  000402                          BR      2$              ;COMMA
045312  000767                          BR      1$              ;PERIOD
045314  000412                          BR      4$              ;DOUBLE PERIOD
045316  010420          2$:             MOV     R4,(R0)+        ;SAVE NEW RMCS1
045320  012703  045446                  MOV     #MRMVEC,R3      ;"RMVEC="
045324  011004                          MOV     (R0),R4         ;PRESENT RH/RM VECTOR ADDRESS
045326  004037  034070                  JSR     R0,GETNUM       ;GET NEW RMVEC
045332  000402                          BR      3$              ;COMMA
045334  000756                          BR      1$              ;PERIOD
045336  000401                          BR      4$              ;DOUBLE PERIOD
045340  010420                  3$:     MOV     R4,(R0)+        ;SAVE NEW RMVEC
045342  010410                  4$:     MOV     R4,(R0)         ;SAVE INPUT
045344  013701  000004          5$:     MOV     ERRVEC,R1       ;SAVE THE ERROR VECTOR
045350  012737  045404  000004          MOV     #6$,ERRVEC      ;SETUP FOR TRAP
045356  005777  134106                  TST     @RH.ADR         ;CHECK FOR RH/RM
045362  010137  000004                  MOV     R1,ERRVEC       ;RESTORE ERROR VECTOR
045366  012700  001470                  MOV     #RH.ADR,R0      ;FIRST ADDRESS OF NEW PARAMETERS
045372  012701  037066                  MOV     #RMADR,R1       ;FIRST ADDRESS OF WHERE TO PUT THEM
045376  012021                          MOV     (R0)+,(R1)+     ;BUS ADDRESS
045400  012021                          MOV     (R0)+,(R1)+     ;VECTOR ADDRESS
045402  000207                          RTS     PC              ;RETURN
045404  010137  000004          6$:     MOV     R1,ERRVEC       ;RESTORE ERROR VECTOR
045410  022626                          CMP     (SP)+,(SP)+     ;CLEAN OFF THE STACK
045412  104010                          EMT     10
045414  005737  000042                  TST     @#42            ;IS THERE A MONITOR?
045420  001724                          BEQ     1$              ;NO--GO ASK FOR ADDRESS
045422  005037  001326                  CLR     DRVSEL          ;YES--NO DRIVES SELECTED
045426  005037  021172                  CLR     $EOPCT          ;NO PASSES
045432  000137  021020                  JMP     $EOP            ;GO TO END OF PROGRAM
                                ;
045436      200     122     115 MRMCS1: .ASCIZ   <CRLF>/RMCS1=/
045446      200     122     115 MRMVEC: .ASCIZ   <CRLF>/RMVEC=/
                                ;
```

```
                                .SBTTL  DPB (DATA PARAMETER BLOCKS)

045456      000         DPB.A:  .BYTE   0               ;(0) DRIVE NUMBER
045457      000                 .BYTE   0               ;(1) OFFSET VALUE OR FMT16, ECI, AND HCI
045460      000                 .BYTE   0               ;(2) COMMAND
045461      000                 .BYTE   0               ;(3) PSEL AND A17 AND A16
045462   000000                 .WORD   0               ;(4) WORD COUNT (MUST BE NEG.)
045464   052776                 .WORD   BUFFER          ;(6) BUFFER ADDRESS OR
                                                        ;REGISTER TABLE POINTER
045466      000                 .BYTE   0               ;(10) SECTOR ADDRESS OR
                                                        ;FIRST REG. INDEX
045467      000                 .BYTE   0               ;(11) TRACK ADDRESS OR
                                                        ;LAST REG. INDEX
045470   000000                 .WORD   0               ;(12) CYLINDER ADDRESS
045472   045556                 .WORD   RM.REG          ;(14) ERROR TABLE POINTER
                                                        ;POINTS TO THE FIRST OF TWENTY
                                                        ;LOCATIONS OF WHERE THE DRIVER
                                                        ;IS TO STORE THE RH/RM
                                                        ;REGISTERS ON AN ERROR. IF LEFT
                                                        ;ZERO REGISTERS ARE NOT SAVED.
045474   000000                 .WORD   0               ;(16) STATUS/ERROR INDICATOR
                                                        ;BIT15=1=>ERROR OCCURRED
                                                        ;BIT07=1=>DONE
                                                        ;BIT14-BIT09 AND BIT06-BIT03
                                                        ;INDICATE TYPE OF ERROR

045476      000         DPB.B:  .BYTE   0               ;(0) DRIVE NUMBER
045477      000                 .BYTE   0               ;(1) OFFSET VALUE OR FMT16, ECI, AND HCI
045500      000                 .BYTE   0               ;(2) COMMAND
045501      000                 .BYTE   0               ;(3) PSEL AND A17 AND A16
045502   177776                 .WORD   -2              ;(4) WORD COUNT (MUST BE NEG.)
045504   052776                 .WORD   BUFFER          ;(6) BUFFER ADDRESS OR
                                                        ;REGISTER TABLE POINTER
045506      000                 .BYTE   0               ;(10) SECTOR ADDRESS OR
                                                        ;FIRST REG. INDEX
045507      000                 .BYTE   0               ;(11) TRACK ADDRESS OR
                                                        ;LAST REG. INDEX
045510   000000                 .WORD   0               ;(12) CYLINDER ADDRESS
045512   045556                 .WORD   RM.REG          ;(14) ERROR TABLE POINTER
                                                        ;POINTS TO THE FIRST OF TWENTY
                                                        ;LOCATIONS OF WHERE THE DRIVER
                                                        ;IS TO STORE THE RH/RM
                                                        ;REGISTERS ON AN ERROR. IF LEFT
                                                        ;ZERO REGISTERS ARE NOT SAVED.
045514   000000                 .WORD   0               ;(16) STATUS/ERROR INDICATOR
                                                        ;BIT15=1=>ERROR OCCURRED
                                                        ;BIT07=1=>DONE
                                                        ;BIT14-BIT09 AND BIT06-BIT03
                                                        ;INDICATE TYPE OF ERROR

045516      000         DPB.C:  .BYTE   0               ;(0) DRIVE NUMBER
045517      000                 .BYTE   0               ;(1) OFFSET VALUE OR FMT16, ECI, AND HCI
045520      000                 .BYTE   0               ;(2) COMMAND
045521      000                 .BYTE   0               ;(3) PSEL AND A17 AND A16
045522   177776                 .WORD   -2              ;(4) WORD COUNT (MUST BE NEG.)
045524   052776                 .WORD   BUFFER          ;(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   RM.REG          ;(14) ERROR TABLE POINTER
                                                                ;POINTS TO THE FIRST OF TWENTY
                                                                ;LOCATIONS OF WHERE THE DRIVER
                                                                ;IS TO STORE THE RH/RM
                                                                ;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

                                DTADPB: .BYTE   0               ;(0) DRIVE NUMBER
                                        .BYTE   0               ;(1) OFFSET VALUE OR FMT16, ECT, AND HCI
                                        .BYTE   0               ;(2) COMMAND
                                        .BYTE   0               ;(3) PSEL AND A17 AND A16
                                        .WORD   0               ;(4) WORD COUNT (MUST BE NEG.)
                                        .WORD   BUFFER          ;(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   RM.REG          ;(14) ERROR TABLE POINTER
                                                                ;POINTS TO THE FIRST OF TWENTY
                                                                ;LOCATIONS OF WHERE THE DRIVER
                                                                ;IS TO STORE THE RH/RM
                                                                ;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

                        ;SAVE RH/RM REGISTERS HERE ON ERROR

                        RM.REG: .WORD   0               ;RMCS1 (776700) CONTROL & STATUS #1
                                .WORD   0               ;RMWC (776702) WORD COUNT
                                .WORD   0               ;RMBA (776704) BUS ADDRESS
                                .WORD   0               ;RMDA (776706) DESIRED SECTOR/TRACK
                                .WORD   0               ;RMCS2 (776710) CONTROL & STATUS #2
                                .WORD   0               ;RMDS (776712) DISK STATUS
                                .WORD   0               ;RMER1 (776714) ERROR REG. #1
                                .WORD   0               ;RMAS (776716) ATTENTION SUMMARY
                                .WORD   0               ;RMLA (776720) LOOK AHEAD
                                .WORD   0               ;RMDB (776722) DATA BUFFER
                                .WORD   0               ;RMMR1 (776724) MAINTAINABILITY
                                .WORD   0               ;RMDT (776726) DRIVE TYPE
                                .WORD   0               ;RMSN (776730) SERIAL NUMBER
                                .WORD   0               ;RMOF (776732) OFFSET
```

```
115 045612  000000                    .WORD   0       ;RMDC (776734) DESIRED CYLINDER
116 045614  000000                    .WORD   0       ;RMHR (776736) CURRENT CYLINDER
117 045616  000000                    .WORD   0       ;RMMR2 (776740) ERROR REG #2
118 045620  000000                    .WORD   0       ;RMER2 (776742) ERROR REG #3
119 045622  000000                    .WORD   0       ;RMEC1 (776744) ECC POSITION
120 045624  000000                    .WORD   0       ;RMEC2 (776746) ECC PATTERN
121                             .EVEN
122
```

```
                                .SBTTL  ASCIZ MESSAGES

045626  122  000       MSG.R:  .ASCIZ  /R/
045630  106  103  000  MSG.FC: .ASCIZ  /FC/
045633  114  103  000  MSG.LC: .ASCIZ  /LC/
045636  111  103  000  MSG.IC: .ASCIZ  /IC/
045641  106  124  000  MSG.FT: .ASCIZ  /FT/
045644  114  124  000  MSG.LT: .ASCIZ  /LT/
045647  111  124  000  MSG.IT: .ASCIZ  /IT/
045652  106  124  047  MES.FT: .ASCIZ  /FT'/
045656  114  124  047  MES.LT: .ASCIZ  /LT'/
045662  111  124  047  MES.IT: .ASCIZ  /IT'/
045666  106  123  000  MSG.FS: .ASCIZ  /FS/
045671  114  123  000  MSG.LS: .ASCIZ  /LS/
045674  120  101  124  MSG.PAT:.ASCIZ  /PAT/
045700  075  000       MSG.EQ: .ASCIZ  /=/
045702  200  103  117  MSG.CS: .ASCIZ  <CRLF>/CONTROL SWITCHES=/

045725  040  057  040  SLASH:  .ASCIZ  @ / @
045731  200  125  116  UNSTAT: .ASCIZ  <CRLF>/UNIT STATUS:/
045747  040  117  106  UNTOFF: .ASCIZ  / OFFLINE/
045760  040  117  116  UNTON·  .ASCIZ  / ONLINE/
045770  040  116  117  NOTPRS: .ASCIZ  / NOT PRESENT/
046005  040  125  116  NOTSAF: .ASCIZ  / UNSAFE/
046015  040  116  117  NOTRM:  .ASCIZ  @ NOT AN RM05/3/2@
046036  040  111  123  LODEV:  .ASCIZ  / IS LOAD DEVICE/
046056  122  115  060  $RM02:  .ASCIZ  /RM02/
046063  122  115  060  $RM03: - .ASCIZ  /RM03/
046070  122  115  060  $RM05:  .ASCIZ  /RM05/
046075  200  104  122  DRIVES: .ASCIZ  <CRLF>/DRIVE(S) TO BE TESTED /
046125  116  117  116  NONE:   .ASCIZ  /NONE/
046132  054  040  000  COMMA:  .ASCIZ  /, /
046135  200  116  117  NOCLOK: .ASCIZ  <CRLF>/NO KW11-P CLOCK, TIMING TESTS WILL NOT BE PERFORMED/
046222  200  012  124  TSTDRV: .ASCIZ  <CRLF><LF>/TESTING DRIVE/
046242  115  102  101  SERIAL: .ASCIZ  /MBA SN# /
046253  200  124  105  MSGTST: .ASCIZ  <CRLF>/TEST/

046261  200  012  122  ROTATE: .ASCIZ  <CRLF><LF>/ROTATIONAL SPEED TIMES/
046312  200  012  117  ONECYL: .ASCIZ  <CRLF><LF>/ONE CYLINDER SEEK TIMES/<CRLF>/ * FORWARD/
046357  200  012  101  ACCESS: .ASCIZ  <CRLF><LF>/ACCESS TIME MEASUREMENT/<CRLF>/ * FORWARD/
046424  200  012  115  MXSEEK: .ASCII  <CRLF><LF>/MAXIMUM SEEK TIMES/
046450  200  040  052  FWD:    .ASCIZ  <CRLF>/ * FORWARD/
046464  200  040  052  REV:    .ASCIZ  <CRLF>/ * REVERSE/

046500  200  115  111  MSGMIN: .ASCIZ  <CRLF>/MIN=/
046506  200  115  101  MSGMAX: .ASCIZ  <CRLF>/MAX=/
046514  200  101  126  MSGAVG: .ASCIZ  <CRLF>/AVG=/
046522  060  040  125  MSGOUS: .ASCIZ  /0 US/
046527  040  102  105  MBELOW: .ASCIZ  / BELOW THE MINIMUM OF /
046556  040  101  102  MABOVE: .ASCIZ  / ABOVE THE MAXIMUM OF /
046605  040  123  105  MSGSEA: .ASCIZ  / SEARCHES TIMED/
046625  040  123  105  MSGNUM: .ASCIZ  / SEEKS TIMED/
046642  040  116  117  MSGNON: .ASCIZ  / NOT TIMED/
046655  040            BLNKS4: .ASCII  / /
046656  040            BLNKS3: .ASCII  / /
046657  040            BLNKS2: .ASCII  / /
046660  040  000       BLNKS1: .ASCIZ  / /
```

```
     58 046662  101  114  114  MSG7XA: .ASCIZ  @ALLOWABLE ROTATIONAL SPEED LIMITS FOR RM05/3@
     59 046737  101  114  114  MSG7XB: .ASCIZ  /ALLOWABLE ROTATIONAL SPEED LIMITS FOR RM02/
     60 047012  101  114  114  MSG10X: .ASCIZ  /ALLOWABLE ONE CYLINDER SEEK LIMIT/
     61 047054  101  114  114  MSG11X: .ASCIZ  /ALLOWABLE ACCESS TIME LIMIT/
     62 047110  101  114  114  MSG12X: .ASCIZ  /ALLOWABLE MAXIMUM SEEK TIME LIMIT/
     63
```

```
                                        .SBTTL  ERROR HEADER (EM) MESSAGES
                                
047152  122  110  057  EM1:    .ASCIZ  @RH/RM INTERRUPT OCCURRED (RMAS = 0)@
047216  125  116  105  EM2:    .ASCIZ  /UNEXPECTED ATTENTION OCCURRED/
047254  115  101  123  EM3:    .ASCIZ  /MASSBUS PARITY ERROR(MCPE=1)/
047311  115  101  123  EM4:    .ASCIZ  /MASSBUS PARITY ERROR(PAR=1)/
047345  101  104  104  EM5:    .ASCIZ  /ADDRESS PLUG CHANGE BIT SET/
047401  122  110  057  EM10:   .ASCIZ  @RH/RM FAILED TO RESPOND TO ADDRESSING@
047447  104  122  111  EM11:   .ASCIZ  /DRIVE SELECTED IS NOT ONLINE/
047504  111  115  120  EM12:   .ASCIZ  /IMPROPER HEADER DATA/
047531  104  101  124  EM13:   .ASCIZ  /DATA COMPARE FAILURE/
047556  104  111  123  EM17:   .ASCIZ  /DISK ERROR IN TIMING TEST/
047610  103  114  117  EM20:   .ASCIZ  /CLOCK (KW11-P) OVERFLOW IN TIMING TEST/
047657  104  111  123  EM23:   .ASCIZ  /DISK ERROR DURING SEEK/
047706  123  105  105  EM24:   .ASCIZ  /SEEK NOT COMPLETE WITHIN 120 MS/
047746  122  110  057  EM41:   .ASCIZ  @RH/RM ERROR@
047762  106  101  124  EM46:   .ASCIZ  /FATAL WRITE CHECK ERROR/

```

```
                                .SBTTL  STATUS/ERROR INDICATOR MESSAGES

050012      117     106     106 MSGB14: .ASCIZ  /OFFLINE OR UNSAFE DRIVE REQUESTED/
050054      125     116     114 MSGB13: .ASCIZ  /UNLOADED DRIVE REQUESTED/
050105      120     105     122 MSGB12: .ASCIZ  /PERSISTENT UNSAFE/
050127      120     101     122 MSGB11: .ASCIZ  /PARITY ERROR OCCURRED/
050155      106     101     124 MSGB10: .ASCIZ  /FATAL PARITY ERROR/
050200      123     117     106 MSGB09: .ASCIZ  /SOFTWARE TIMEOUT ON THIS DRIVE/
050237      123     117     106 MSGB08: .ASCIZ  /SOFTWARE TIMEOUT ON ANOTHER DRIVE/
050301      105     122     122 MSGB06: .ASCIZ  "ERROR OCCURRED DURING I/O OPERATION"
050345      105     122     122 MSGB05: .ASCIZ  "ERROR OCCURRED DURING NON-I/O OPERATION"
050415      125     116     123 MSGB04: .ASCIZ  /UNSAFE OCCURRED/
050435      101     125     124 MSGB03: .ASCIZ  /AUTOMATIC RECALIBRATE SEQUENCE OCCURRED/
050505      104     122     111 MSGB02: .ASCIZ  /DRIVE HAS NOT RESPONDED TO PORT REQUEST/
050555      104     122     111 MSGB01: .ASCIZ  /DRIVE HAS BECOME NON-EXISTENT/
```

```
                                        .SBTTL  DATA HEADER (DT) MESSAGES
                                
 050613     105     122     122  DH1:    .ASCIZ  /ERR PC  RMAS/
 050630     105     122     122  DH2:    .ASCIZ  /ERR PC  DRIVE   RMAS    RMDS    RMER1   RMMR2   RMER2/
 050715     124     105     123  DH3:    .ASCIZ  /TEST    ERR PC  ADDRESS DATA/
 050752     124     105     123  DH4:    .ASCIZ  /TEST    ERR PC  ADDRESS GDDATA  BDDATA/
 051021     122     115     103  DH10:   .ASCIZ  /RMCS1   ERR PC/
 051040     104     122     111  DH11:   .ASCIZ  /DRIVE   ERR PC/
 051057     124     105     123  DH12:   .ASCIZ  /TEST    ERR PC  TST PC  DRIVE   CYLNDR  TRACK   SECTOR/
 051146     107     104     103  DH12A:  .ASCIZ  /GDCYL   GDTRK   GDSCTR  BDCYL   BDTRK   BDSCTR/
 051225     107     104     104  DH13A:  .ASCIZ  /GDDAT   BDDAT   WRDCNT  GDADR   BDADR/
 051273     124     105     123  DH17:   .ASCIZ  /TEST    ERR PC  DRIVE   RMCS1   RMDS    RMER1   RMMR2   RMER2/
 051370     124     105     123  DH21:   .ASCIZ  /TEST    ERR PC  TST PC  DRIVE   CYLNDR  TRACK/
 051446     107     104     104  DH21A:  .ASCIZ  /GDDAT   BDDAT   WRDCNT  SECTOR/
 051505     124     105     123  DH23:   .ASCIZ  /TEST    ERR PC  DRIVE   CYLNDR  RMCS1   RMCS2   RMDS/
 051572     122     115     105  DH23A:  .ASCIZ  /RMER1   RMMR2   RMER2   RMDC    RMHR/
 051637     124     105     123  DH41:   .ASCIZ  /TEST    ERR PC  TST PC  DRIVE/
 051675     124     105     123  DH42:   .ASCIZ  /TEST    ERR PC  TST PC  DRIVE   RMCS1   RMCS2   RMDS/
 051762     122     115     105  DH43A:  .ASCIZ  /RMER1   RMMR2   RMER2/
 052010     122     115     103  DH44A:  .ASCIZ  /RMCS1   RMCS2   RMDS    RMHR    RMDC    RMDA/
 052064     122     115     105  DH44B:  .ASCIZ  /RMER1   RMMR2   RMER2/
 052112     122     115     105  DH45A:  .ASCIZ  /RMER1   RMMR2   RMER2   RMWC    RMBA    RMDB/

                                        .EVEN
```

```
                                .SBTTL  DATA TABLE (DT)

052170  001132  001204          DT1:    .WORD   $ERRPC,$REG3
052174  001132  001200  001204  DT2:    .WORD   $ERRPC,$REG1,$REG3,RMERRS,RMERRS+2,RMERRS+6,RMERRS+4
052212  001212  001132  043762  DT3:    .WORD   $TMPO,$ERRPC,RD.ADR,RD.WRD
052222  001212  001132  044206  DT4:    .WORD   $TMPO,$ERRPC,WRT.ADR,WRT.WD,RD.WRD
052234  001212  001132  001200  DT5:    .WORD   $TMPO,$ERRPC,$REG1,$REG5,RMERRS,RMERRS+2,RMERRS+6,RMERRS+4
052254  001470  001132          DT10:   .WORD   RH.ADR,$ERRPC
052260  001202  001132          DT11:   .WORD   $REG2,$ERRPC
052264  001212  001132  001176  DT12:   .WORD   $TMPO,$ERRPC,$REG0,CHKDRV,CYL.DS,TRK.DS,SEC.DS
052302  001364  001370  001366  DT12A:  .WORD   CYL.DS,TRK.DS,SEC.DS,CYL.RD,TRK.RD,SEC.RD
052316  001212  001132  001176  DT13:   .WORD   $TMPO,$ERRPC,$REG0,CHKDRV,CYL.DS,TRK.DS,SEC.DS
052334  001140  001142  001206  DT13A:  .WORD   $GDDAT,$BDDAT,$REG4,$GDADR,$BDADR
052346  001212  001132  001350  DT17:   .WORD   $TMPO,$ERRPC,CHKDRV,RM.REG,RM.REG+12,RM.REG+14,RM.REG+40,RM.REG+42
052366  001212  001132  001176  DT21:   .WORD   $TMPO,$ERRPC,$REG0,CHKDRV,CYL.DS,TRK.DS
052402  001200  001142  001206  DT21A:  .WORD   $REG1,$BDDAT,$REG4,$REG1
052412  001212  001132  001350  DT23:   .WORD   $TMPO,$ERRPC,CHKDRV,CYL.DS,RM.REG,RM.REG+10,RM.REG+12
052430  045572  045616  045620  DT23A:  .WORD   RM.REG+14,RM.REG+40,RM.REG+42,RM.REG+34,RM.REG+36
052442  001212  001132  001176  DT41:   .WORD   $TMPO,$ERRPC,$REG0,CHKDRV
052452  001212  001132  001176  DT42:   .WORD   $TMPO,$ERRPC,$REG0,CHKDRV,RM.REG,RM.REG+10,RM.REG+12
052470  001212  001132  001176  DT43:   .WORD   $TMPO,$ERRPC,$REG0,CHKDRV,RM.REG,RM.REG+10,RM.REG+12
052506  045572  045616  045620  DT43A:  .WORD   RM.REG+14,RM.REG+40,RM.REG+42
052514  001212  001132  001176  DT44:   .WORD   $TMPO,$ERRPC,$REG0,CHKDRV,CYL.DS,TRK.DS,SEC.DS
052532  045556  045566  045570  DT44A:  .WORD   RM.REG,RM.REG+10,RM.REG+12,RM.REG+36,RM.REG+34,RM.REG+06
052546  045572  045616  045620  DT44B:  .WORD   RM.REG+14,RM.REG+40,RM.REG+42
052554  001212  001132  001176  DT45:   .WORD   $TMPO,$ERRPC,$REG0,CHKDRV,CYL.DS,TRK.DS,SEC.DS
052572  045556  045566  045570  DT45A:  .WORD   RM.REG,RM.REG+10,RM.REG+12,RM.REG+36,RM.REG+34,RM.REG+06
052606  045572  045616  045620  DT45B:  .WORD   RM.REG+14,RM.REG+40,RM.REG+42,RM.REG+2,RM.REG+4,RM.REG+22
```

```
                                .SBTTL  DATA FORMAT (DF) TABLE
                        
052622  000001          DF1:    .WORD   1               ;NUMBER OF DATA HEADERS
052624     002                  .BYTE   2               ;NUMBER OF WORDS IN DATA TABLE
052625     000                  .BYTE   0               ;ALL 3 NUMBERS ARE OCTAL
                        
052626  000001          DF2:    .WORD   1
052630     007                  .BYTE   7
052631     000                  .BYTE   0
                        
052632  000001          DF3:    .WORD   1
052634     004                  .BYTE   4
052635     000                  .BYTE   0
                        
052636  000001          DF4:    .WORD   1
052640     005                  .BYTE   5
052641     000                  .BYTE   0
                        
052642  000001          DF10:   .WORD   1
052644     002                  .BYTE   2
052645     000                  .BYTE   0
                        
052646  000001          DF11:   .WORD   1
052650     002                  .BYTE   2
052651     000                  .BYTE   0
                        
052652  000002          DF12:   .WORD   2               ;2 DH'S TO BE TYPED
052654     007                  .BYTE   7               ;7 DATA WORDS FOLLOW THE 1ST DH
052655     160                  .BYTE   160             ;WORDS 1-4 ARE OCTAL 5-7 ARE DECIMAL
052656  051146                  .WORD   DH12A           ;ADDRESS OF 2ND DH
052660     006                  .BYTE   6               ;6 DATA WORDS FOLLOW THE 2ND DH
052661     000                  .BYTE   0               ;ALL WORDS ARE OCTAL
                        
052662  000002          DF13:   .WORD   2
052664     007                  .BYTE   7
052665     160                  .BYTE   160
052666  051225                  .WORD   DH13A
052670     005                  .BYTE   5
052671     004                  .BYTE   4               ;WORD 3 IS DECIMAL
                        
052672  000000          DF14:   .WORD   0
052674     005                  .BYTE   5
052675     004                  .BYTE   4               ;WORD 3 IS DECIMAL
                        
052676  000001          DF17:   .WORD   1
052700     010                  .BYTE   ^D8
052701     000                  .BYTE   0
                        
052702  000002          DF21:   .WORD   2
052704     006                  .BYTE   6
052705     060                  .BYTE   60
052706  051446                  .WORD   DH21A
052710     004                  .BYTE   4
052711     014                  .BYTE   14
                        
052712  000000          DF22:   .WORD   0
052714     004                  .BYTE   4
```

```
                                                .BYTE   14
                                        DF23:   .WORD   2
                                                .BYTE   7
                                                .BYTE   10              ;WORD 4 IS DECIMAL
                                                .WORD   DH23A
                                                .BYTE   5
                                                .BYTE   0

                                        DF41:   .WORD   1
                                                .BYTE   4
                                                .BYTE   0

                                        DF42:   .WORD   1
                                                .BYTE   7
                                                .BYTE   0

                                        DF43:   .WORD   2
                                                .BYTE   7
                                                .BYTE   0
                                                .WORD   DH43A
                                                .BYTE   3
                                                .BYTE   0

                                        DF44:   .WORD   3
                                                .BYTE   7
                                                .BYTE   160
                                                .WORD   DH44A
                                                .BYTE   6
                                                .BYTE   0
                                                .WORD   DH44B
                                                .BYTE   3
                                                .BYTE   0

                                        DF45:   .WORD   3
                                                .BYTE   7
                                                .BYTE   160
                                                .WORD   DH44A
                                                .BYTE   6
                                                .BYTE   0
                                                .WORD   DH45A
                                                .BYTE   6
                                                .BYTE   0

                                        .SBTTL  START OF READ/WRITE BUFFER
                                        BUFFER:

                                        .END    200
```

```
ABASE = 000000
ACCESS  046357
ACDW1 = 000000
ACDW2 = 000000
ACPUOP= 000000
ACTDRV  037004
ACTSTR  037005
ADDW0 = 000000
ADDW1 = 000000
ADDW10= 000000
ADDW11= 000000
ADDW12= 000000
ADDW13= 000000
ADDW14= 000000
ADDW15= 000000
ADDW2 = 000000
ADDW3 = 000000
ADDW4 = 000000
ADDW5 = 000000
ADDW6 = 000000
ADDW7 = 000000
ADDW8 = 000000
ADDW9 = 000000
ADEVCT= 000000
ADEVM = 000000
AENV  = 000000
AENVM = 000000
AFATAL= 000000
AMADR1= 000000
AMADR2= 000000
AMADR3= 000000
AMADR4= 000000
AMAMS1= 000000
AMAMS2= 000000
AMAMS3= 000000
AMAMS4= 000000
AMSGAD= 000000
AMSGLG= 000000
AMSGTY= 000000
AMTYP1= 000000
AMTYP2= 000000
AMTYP3= 000000
AMTYP4= 000000
AOE   = 001000
APASS = 000000
APRIOR= 000000
APTCSU= 000040
APTENV= 000001
APTSIZ= 000200
APTSPO= 000100
ASWREG= 000000
ATA   = 100000
ATABIT  037054
ATESTN= 000000
AT0   = 000001
AT1   = 000002
AT2   = 000004
AT3   = 000010
AT4   = 000020
AT5   = 000040
AT6   = 000100
AT7   = 000200
AUNIT = 000000
AUSWR = 000000
AVECT1= 000000
AVECT2= 000000
A16   = 000400
A17   = 001000
BAI   = 000010
BASFLG  001464
BITS    001526
BIT0  = 000001
BIT00 = 000001
BIT01 = 000002
BIT02 = 000004
BIT03 = 000010
BIT04 = 000020
BIT05 = 000040
BIT06 = 000100
BIT07 = 000200
BIT08 = 000400
BIT09 = 001000
BIT1  = 000002
BIT10 = 002000
BIT11 = 004000
BIT12 = 010000
BIT13 = 020000
BIT14 = 040000
BIT15 = 100000
BIT2  = 000004
BIT3  = 000010
BIT4  = 000020
BIT5  = 000040
BIT6  = 000100
BIT7  = 000200
BIT8  = 000400
BIT9  = 001000
BLNKS1  046660
BLNKS2  046657
BLNKS3  046656
BLNKS4  046655
BPTVEC= 000014
BUFFER  052776
BUSADR  001322
BYPASS  001346
CALL.A  027132
CALL.B  027264
CALL.C  027466
CHKDRV  001350
C11     040404
C13     040512
C14     040612
C15     041170
C16     041212
C17     041226
C17B    041250
C18     041326
CKSCTR  032362
CKSWR = 104407
CK.CHR  036340
CK.DEC  036312
CK.DIG  036414
CK.NUM  036600
CK.OCT  036264
CLKSTA  001340
CLOSE   036204
CLR   = 000040
CLRBUF  032314
CLRQUE  044772
CLSWDS  036104
CNTCLR  026072
CNTRLC  001320
COMMA   046132
CONT  = 000100
COUNT   031336
CR    = 000015
CRLF  = 000200
CYL.DS  001364
CYL.RD  001356
C.SWR   001314
DATCMP  033022
DCK   = 100000
DDISP = 177570
DECSEC  032232
DECSK   007630
DELTA   001446
DFLT    002514
DF1     052622
DF10    052642
DF11    052646
DF12    052652
DF13    052662
DF14    052672
DF17    052676
DF2     052626
DF21    052702
DF22    052712
DF23    052716
DF3     052632
DF4     052636
DF41    052726
DF42    052732
DF43    052736
DF44    052746
DF45    052762
DH1     050613
DH10    051021
DH11    051040
DH12    051057
DH12A   051146
DH13A   051225
DH17    051273
DH2     050630
DH21    051370
DH21A   051446
DH23    051505
DH23A   051572
DH3     050715
DH4     050752
DH41    051637
DH42    051675
DH43A   051762
DH44A   052010
DH44B   052064
DH45A   052112
DISPLA  001156
DISPRE  000174
DLT   = 100000
DMD   = 000001
DORTI   031272
DPB.A   045456
DPB.B   045476
DPB.C   045516
DPINT   036760
DPR   = 000400
DPRQS   036770
DRIVES  046075
DRQ   = 004000
DRVACT  036730
DRVCAL  027670
DRVCLR= 000111
DRVCL1  027710
DRVINT  037316
DRVMSK  001352
DRVOK   006560
DRVQUE  045070
DRVSEL  001326
DRVSTA  036740
DRVTYP  036750
DRY   = 000200
DSWR  = 177570
DTADPB  045536
DTE   = 010000
DTG   = 000020
DTO   = 020000
DTUW    037052
DT00  = 000001
DT01  = 000002
DT02  = 000004
DT03  = 000010
DT04  = 000020
DT05  = 000040
DT06  = 000100
DT07  = 000200
DT08  = 000400
DT1     052170
DT10    052254
DT11    052260
DT12    052264
DT12A   052302
DT13    052316
DT13A   052334
DT17    052346
DT2     052174
DT21    052366
DT21A   052402
DT23    052412
DT23A   052430
DT3     052212
DT4     052222
DT41    052442
DT42    052452
DT43    052470
DT43A   052506
DT44    052514
DT44A   052532
DT44B   052546
DT45    052554
DT45A   052572
DT45B   052606
DT5     052234
DULP    037320
DVA   = 004000
EBL   = 020000
ECH   = 000100
ECI   = 004000
ECRC  = 001000
EECC  = 000020
EMPTYQ  045050
EMTVEC= 000030
EM1     047152
EM10    047401
EM11    047447
EM12    047504
EM13    047531
EM17    047556
EM2     047216
EM20    047610
EM23    047657
EM24    047706
EM3     047254
EM4     047311
EM41    047746
EM46    047762
EM5     047345
ERINDX  030526
ERR   = 040000
ERROR = 104000
ERRVEC= 000004
ERR.CT  001462
ESRC  = 004000
ES.SAV  045234
EXIT.A  014120
EXIT0   007202
EXIT1   007410
```

```
EXIT10  012746      INCTRK  032152      MSGB10  050155      OPNFLG  001334      PRM14   003402
EXIT11  013212      IOTVEC= 000020      MSGB11  050127      OPNPAT  035572      PRM15   003412
EXIT12  014134      IR    = 000100      MSGB12  050105      OPNPRM  035370      PRM16   003422
EXIT13  014636      ISR     041740      MSGB13  050054      OPNTST  035216      PRM17   003434
EXIT14  015376      IT      002340      MSGB14  050012      OPNWDS  035714      PRM2    003160
EXIT15  016136      ITEM41  002242      MSGMAX  046506      OPN.CT  035222      PRM20   003444
EXIT16  016560      LA      041564      MSGMIN  046500      OPN.N1  036126      PRM21   003500
EXIT17  017224      LACNT   037016      MSGNON  046642      OPN.N2  036132      PRM22   003510
EXIT2   007652      LC      002330      MSGNUM  046625      OPN.X1  036144      PRM3    003200
EXIT20  020004      LDCMD   027066      MSGSEA  046605      OPN.X2  036150      PRM4    003220
EXIT21  020622      LF    = 000012      MSGTST  046253      OPN.1   035226      PRM5    003240
EXIT22  021016      LKS     001514      MSG.CS  045702      OPN.2   035250      PRM6    003260
EXIT3   010072      LKV     001510      MSG.EQ  045700      OPT     040146      PRM7    003300
EXIT4   010504      LODEV   046036      MSG.FC  045630      OR    = 000200      PR0   = 000000
EXIT5   010730      LODFLT  026434      MSG.FS  045666      PACK  = 000123      PR1   = 000040
EXIT6   011220      LODPRM  026672      MSG.FT  045641      PAR   = 000010      PR2   = 000100
EXIT7   011616      LOP.CK  030466      MSG.IC  045636      PAT     002346      PR3   = 000140
FC      002326      LPB     001524      MSG.IT  045647      PAT.PT  003520      PR4   = 000200
FER   = 000020      LPS     001522      MSG.LC  045633      PAT0    003560      PR5   = 000240
FILBUF  032256      LPTAVL  001324      MSG.LS  045671      PAT1    003620      PR6   = 000300
FILRAN  033340      LP.AVL  026114      MSG.LT  045644      PAT10   004260      PR7   = 000340
FMT16 = 010000      LS      002344      MSG.PA  045674      PAT11   004320      PS    = 177776
FS      002342      LSIT  = 000002      MSG.R   045626      PAT12   004360      PSEL  = 002000
FT      002334      LST   = 002000      MSGOUS  046522      PAT13   004420      PSW   = 177776
FWD     046450      LSTRK   001372      MSG10X  047012      PAT14   004460      PTRN15  003476
F1    = 000002      LT      002336      MSG11X  047054      PAT15   004520      PWRVEC= 000024
F2    = 000004      MABOVE  046556      MSG12X  047110      PAT2    003660      QCNT    044500
F3    = 000010      MBELOW  046527      MSG7XA  046662      PAT3    003720      QDRV0   044572
F4    = 000020      MCLK  = 004000      MSG7XB  046737      PAT4    003760      QDRV1   044612
F5    = 000040      MCPE  = 020000      MUR   = 001000      PAT5    004020      QDRV2   044632
GETADR  045260      MCPEMX  037064      MWP   = 000010      PAT6    004060      QDRV3   044652
GETNUM  034070      MDF   = 000100      MXDLTA  037076      PAT7    004120      QDRV4   044672
GETREG= 000141      MES.FT  045652      MXF   = 001000      PAT8    004160      QDRV5   044712
GETREQ  045144      MES.IT  045662      MXLACT  037074      PAT9    004220      QDRV6   044732
GETSWR  033776      MES.LT  045656      MXSEEK  046424      PDA   = 000400      QDRV7   044752
GO    = 000001      MI    = 000004      MXSTAL  001460      PGE   = 002000      QINPT   044510
GTSWR = 104406      MNDLTA  037100      MXWNDW  037102      PGM   = 001000      QOUTPT  044530
GTTST1  034364      MOC   = 000400      NBA   = 100000      PHA   = 000200      QSTART  044550
GTTST2  034450      MOH   = 020000      NC1     002356      PIP   = 020000      QSTOP   044552
GTTST3  035012      MOL   = 010000      NC2     002360      PIRQ  = 177772      QTERM = 044772
GTTST4  035014      MPE   = 000400      NED   = 010000      PIRQVE= 000240      RANADR  033614
GTTST5  035124      MRD   = 002000      NEM   = 004000      PKB     001504      RANCK   033362
GTTST6  035164      MRMCS1  045436      NOCLOK  046135      PKC     001506      RANPAT  033560
GT.PRM  034226      MRMVEC  045446      NONE    046125      PKCS    001502      RDCHR = 104410
GT.PR1  034230      MS    = 000040      NOOP  = 000101      PKV     001476      RDLIN = 104411
GT.PR2  034360      MSC   = 000002      NOTPRS  045770      PLFS  = 002000      RDY   = 000200
HCE   = 000200      MSEN  = 010000      NOTRM   046015      POPQUE  045166      RD.ADR  043762
HCI   = 002000      MSER  = 000200      NOTSAF  046005      PRM     002322      RD.RM   043736
HCRC  = 000400      MSGAVG  046514      OBCK  = 100000      PRMLMT  002432      RD.RM1  043760
HT    = 000011      MSGB01  050555      OBEN  = 040000      PRMMSG  002464      RD.RM2  044104
IAE   = 002000      MSGB02  050505      OCC   = 100000      PRMPT   002362      RD.RM3  044110
IC      002332      MSGB03  050435      OFD   = 000200      PRM0    003116      RD.RM4  044114
IE    = 000100      MSGB04  050415      OFFSET= 000115      PRM1    003132      RD.WRD  043764
ILF   = 000001      MSGB05  050345      OM    = 000001      PRM10   003320      READ  = 000171
ILR   = 000002      MSGB06  050301      ONECYL  046312      PRM11   003336      READHD= 000173
INCCYL  032202      MSGB08  050237      OPE   = 020000      PRM12   003356      READIN= 000121
INCSK   007566      MSGB09  050200      OPI   = 020000      PRM13   003372      RECAL = 000107
```

SYMBOL TABLE

```
RELEAS= 000113          SC5     042242          STRT2A  004616          TEST7   011362          T7B     001616
RESREG= 104413          SC6     042436          ST.CLK  026156          TICKMS  001342          T7B1    001636
RESTAR  006540          SC6A    042552          ST.LCL  026366          TICKUS  001344          ULDFLG  037006
RESVEC= 000010          SC7     042700          ST.PCL  026324          TIMER   037032          UNLOAD= 000103
REV     046464          SC8     042756          SVADR   001436          TIM.DN  001412          UNS    = 040000
REX    = 010000         SEARCH= 000131          SVRH70  044310          TIM.PT  001430          UNSTAT  045731
RHVEC   001472          SEC.DS  001366          SVSTAT  001354          TIM.UP  001374          UNTOFF  045747
RH.ADR  001470          SEC.RD  001362          SWR     001154          TKVEC = 000060          UNTON   045760
RMADR   037066          SEEK   = 000105          SWREG   000176          TPB     001520          UPE    = 020000
RMAS   = 000016         SEEKFG  037030          SW0    = 000001         TPS     001516          US1    = 000001
RMBA   = 000004         SEKCNT  001444          SW00   = 000001         TPS50   014132          US2    = 000002
RMCS1  = 000000         SEKTMR  001442          SW01   = 000002         TPS60   014114          US4    = 000004
RMCS2  = 000010         SELDRV= 000145          SW02   = 000004         TPVEC = 000064          VERIFY  030746
RMDA   = 000006         SERIAL  046242          SW03   = 000010         TP50    014124          VV     = 000100
RMDB   = 000022         SETBUF  032732          SW04   = 000020         TP60    014106          WC     = 000040
RMDC   = 000034         SETFOR= 000143          SW05   = 000040         TRAPVE= 000034          WCE    = 040000
RMDS   = 000012         SETVEC  005760          SW06   = 000100         TRCKWC  001450          WCEFLG  001432
RMDT   = 000026         SET.IE  044426          SW07   = 000200         TRE    = 040000         WCF    = 000040
RMEC1  = 000044         SKI    = 040000         SW08   = 000400         TRK.DS  001370          WD     = 000010
RMEC2  = 000046         SLASH   045725          SW09   = 001000         TRK.RD  001360          WLE    = 004000
RMERRS  036720          SPTYP   031470          SW1    = 000002         TRNSWT  037000          WRCKD  = 000151
RMER1  = 000014         SP10    001726          SW10   = 002000         TRTVEC= 000014          WRCKHD= 000153
RMER2  = 000042         SP11    001734          SW11   = 004000         TSTDRV  046222          WRITE  = 000161
RMHR   = 000036         SP12    001742          SW12   = 010000         TSTNMS  001330          WRL    = 004000
RMINIT  037104          SP7A    001676          SW13   = 020000         TST0    007016          WRTHD  = 000163
RMLA   = 000020         SP7A1   001712          SW14   = 040000         TST1    007204          WRT.AD  044206
RMMR1  = 000024         SP7B    001704          SW15   = 100000         TST10   011620          WRT.RM  044116
RMMR2  = 000040         SP7B1   001720          SW2    = 000004         TST10A  012310          WRT.R1  044202
RMOF   = 000032         SRCHWT  037002          SW3    = 000010         TST10B  012702          WRT.R2  044274
RMR    = 000004         SRCH00  031110          SW4    = 000020         TST11   012750          WRT.R3  044300
RMSN   = 000030         SRTDRV  006232          SW5    = 000040         TST12   013214          WRT.R4  044304
RMTMR   043324          SRTINT  005726          SW6    = 000100         TST13   014136          WRT.R5  044306
RMVEC   037070          SRVCLK  026422          SW7    = 000200         TST14   014640          WRT.WD  044204
RMWC   = 000002         STACK  = 001100         SW8    = 000400         TST15   015400          XXDP    001466
RM.REG  045556          STALL   030664          SW9    = 001000         TST16   016140          $APTHD  001100
RMOS    037654          STALL0  001434          TAB.XY= 001114          TST17   016562          $ATYC   021310
ROTATE  046261          STALL1  001452          TAP    = 040000         TST2    007412          $ATY1   021264
RPT     002324          STALL2  001454          TBITVE= 000014          TST20   017226          $ATY3   021272
RSTRT1  006504          STALL3  001456          TD      042004          TST21   020006          $ATY4   021302
RSTRT2  006530          START   004624          TEST0   007166          TST22   020624          $AUTOB  001150
RTC    = 000117         START1  004570          TEST1   007374          TST3    007654          $BASE   001310
R6     =%000006         START2  004612          TEST10  012052          TST4    010074          $BDADR  001136
R7     =%000007         START3  004560          TEST11  013134          TST5    010506          $BDDAT  001142
SAVCSW  001316          START4  004602          TEST12  013612          TST6    010732          $BELL   001224
SAVEFG  037026          STATBL  001750          TEST13  014346          TST7    011222          $CHARC  022716
SAVREG= 104412          STKLMT= 177774          TEST14  015050          TYPDS  = 104405         $CKSWR  023672
SC      042154          STO     043414          TEST15  015610          TYPE   = 104401         $CMTAG  001114
SCOPE  = 000004         STO1    043450          TEST16  016260          TYPERR  022000          $CM1   = 000006
SCTRWC= 177400          STO2    043544          TEST17  016702          TYPOC  = 104402         $CM2   = 000014
SC0    = 000100         STO3    043574          TEST2   007546          TYPON  = 104404         $CM3   = 000006
SC1    = 000200         STO5    043620          TEST20  017454          TYPOS  = 104403         $CM4   = 000003
SC11    043006          STO6    043626          TEST21  020302          TYPTIM  031576          $CNTLC  024604
SC12    043116          STO7    043664          TEST22  020736          T10     001646          $CNTLG  024616
SC13    043166          STO8    043714          TEST3   010010          T11     001656          $CNTLU  024611
SC2    = 000400         S109    043724          TEST4   010244          T12     001666          $CPUOP  001262
SC3     042224          STRTMR  031274          TEST5   010642          T7A     001606          $CRLF   001231
SC4     042230          STRT1A  004574          TEST6   011066          T7A1    001626          $DBLK   023364
```

```
$DB2D   025314        $GET42  021234        $MTYP1  001265        $RTNAD  021256        $TPB    001166
$DECVL  025474        $GTSWR  023762        $MTYP2  001271        $SAVRE  025100        $TPFLG  001173
$DEVCT  001244        $HD   = 000000        $MTYP3  001275        $SB2D   025260        $TPS    001164
$DEVM   001312        $HIBTS  001100        $MTYP4  001301        $SCOPE  024646        $TRAP   025174
$DIV    025652        $HINUM  025646        $MXCNT  025076        $SETUP= 000167        $TRAP2  025216
$DOAGN  021254        $ICNT   001120        $NULL   001170        $STUP = 177777        $TRP  = 000014
$DTBL   023354        $INTAG  001151        $NWTST= 000001        $SUPRS  025510        $TRPAD  025230
$ENDAD  021244        $ITEMB  001130        $OCNT   023144        $SVLAD  025026        $TSTM   001104
$ENDCT  021200        $LF     001232        $OMODE  023146        $SVPC = 000220        $TSTNM  001116
$ENULL  021260        $LFLG   021527        $OVER   025062        $SWR  = 167000        $TTYIN  02456C
$ENV    001254        $LONUM  025650        $PASS   001242        $SWREG  001256        $TYPDS  023150
$ENVM   001255        $LPADR  001122        $PASTM  001106        $SWRMK= 000000        $TYPE   022366
$EOP    021020        $LPERR  001124        $QUES   001230        $TESTN  001240        $TYPEC  022600
$EOPCT  021172        $MADR1  001266        $RAND   025550        $TIMES  001220        $TYPEX  022720
$ERFLG  001117        $MADR2  001272        $RDCHR  024234        $TKB    001162        $TYPOC  022746
$ERMAX  001131        $MADR3  001276        $RDLIN  024324        $TKCNT  023374        $TYPON  022762
$ERROR  021532        $MADR4  001302        $RDSZ = 000024        $TKINT  023404        $TYPOS  022722
$ERRPC  001132        $MAIL   001234        $REGAD  001174        $TKQEN= 023404        $UNIT   001246
$ERRTB  002002        $MAMS1  001264        $REG0   001176        $TKQIN  023376        $UNITM  001110
$ERTTL  001126        $MAMS2  001270        $REG1   001200        $TKQOU  023400        $USWR   001260
$ESCAP  001222        $MAMS3  001274        $REG2   001202        $TKQSR  023402        $VECT1  001304
$ETABL  001254        $MAMS4  001300        $REG3   001204        $TKS    001160        $VECT2  001306
$ETEND  001314        $MBADR  001102        $REG4   001206        $TKSRV  023454        $XOFF = 000023
$FATAL  001236        $MFLG   021526        $REG5   001210        $TMP0   001212        $XON  = 000021
$FFLG   021530        $MNEW   024634        $RESRE  025136        $TMP1   001214        $XTSTR  024660
$FILLC  001172        $MSGAD  001250        $RM02   046056        $TMP2   001216        $$GET4= 000000
$FILLS  001171        $MSGLG  001252        $RM03   046063        $TN   = 000023        $OFILL  023145
$GDADR  001134        $MSGTY  001234        $RM05   046070        $TNPWR  025424        .$X   = 001100
$GDDAT  001140        $MSWR   024623

. ABS.  052776      000
        000000      001
ERRORS DETECTED:  0

VIRTUAL MEMORY USED:  54064 WORDS  ( 212 PAGES)
DYNAMIC MEMORY AVAILABLE FOR  69 PAGES
CZRMVA.BIC,CZRMVA/C=CZRMVA.DOC,CZRMVA,SYSMAC/M
```

```
$$GET4  14-1      14-1#
$OFILL  14-105    14-105#  14-105*  14-105*
$40CAT  14-5      14-111
$APTHD   4-319     4-319#
$ASTAT  14-3      14-3
$ATY1   14-3#
$ATY3   14-3#     14-103
$ATY4   14-3#     14-5
$ATYC   14-3      14-3#
$AUTOB   5-0#     10-22*   10-66*   14-109   14-109   14-109
$BASE    5-0#
$BDADR   5-0#     14-:54*  14-;18*  22-13
$BDDAT   5-0#     14-986*  14-:56*  14-;20*  22-13    22-16
$BELL    5-0#     10-26    14-5     14-5     14-5     14-109   14-109   14-109
$CHARC  14-103    14-103#  14-103*  14-103*  14-103*
$CKSWR  14-109#   14-115   14-115
$CM1     5-0      5-0      5-0      5-0      5-0      5-0      5-0      5-0      5-0      5-0      5-0      5-0      5-0#     5-0#
         5-0#     5-0#     5-0#     5-0#     5-0#
$CM2     5-0      5-0      5-0      5-0      5-0      5-0      5-0      5-0      5-0      5-0      5-0      5-0      5-0#     5-0#
         5-0#     5-0#     5-0#     5-0#     5-0#
$CM3     5-0      5-0      5-0#
$CM4     5-0      5-0      5-0      5-0      5-0      5-0      5-0#     5-0#     5-0#     5-0#
$CMTAG   5-0#     10-21    10-21    10-21    10-21    10-21    10-21    10-21
$CNTLC  14-109    14-109   14-109   14-109   14-109#
$CNTLG  14-109    14-109#
$CNTLU  14-109    14-109   14-109#
$CPUOP   5-0#
$CRLF    5-0#     10-49    10-71    10-87    10-132   10-250   14-5     14-5     14-5     14-37    14-53    14-64    14-74    14-88
        14-95     14-103   14-103   14-103   14-109   14-109   14-109   14-109   14-761   14-762   14-777   14->74
$DB2D   14-117    14-119#
$DBLK   14-107    14-107   14-107#
$DECVL  14-119    14-119#
$DEVCT   5-0#
$DEVM    5-0#
$DIV    11-324    11-512   14-125#  14-833
$DOAGN  14-1      14-1     14-1#
$DTBL   14-107    14-107#
$ENDAD   4-316    10-22    14-1#    14-5
$ENDCT  10-21     10-168*  10-177*  10-185   14-1#
$ENULL  14-1      14-1#
$ENV     5-0#     10-22    10-66    14-3     14-3     14-5     14-103
$ENVM    5-0#     10-21    14-3     14-103   14-103
$EOP    10-173    10-219   13-373   14-1#    14-544   16-48
$EOPCT  10-21*    10-185*  14-1     14-1#    16-47*
$ERFLG   5-0#     14-5     14-5     14-5*    14-111   14-111   14-111   14-111   14-111   14-111*  14-521   14-971   14-:01   14-:73
        14-;32
$ERMAX   5-0#     10-21*   10-251*  11-44*   11-67*   11-97*   11-129*  11-158*  11-227*  11-262*  11-304*  11-380*  11-527*  12-29*
        12-154*   12-228*   12-305*  13-20*   13-77*   13-156*  13-288*  13-373*  14-111   14-111   14-111   14-111*
$ERROR  10-21     14-5#
$ERRPC   5-0#     14-5     14-5     14-5     14-5*    14-5*    22-3     22-4     22-5     22-6     22-7     22-8     22-9     22-10
        22-12     22-14    22-15    22-17    22-19    22-20    22-21    22-23    22-26
$ERRTB   7-0#     14-33    14-34
$ERTTL   5-0#     14-1     14-1*    14-5     14-5     14-5*
$ESCAP   5-0#     10-21*   12-60*   12-;75*  12-249*  12-326*  14-5     14-5     14-5     14-5     14-111*  14-341*  14-348*  14-368*
        14-375*   14-406*   14-413*  14-447*  14-457*  14-478*  14-485*  14-509*  14-540*  14-648*  14-653*  14-677*  14-684*  14-991*
        14-995*   14-:64*   14-:69*  14-;16*  15-817   15-868   15-870   15-;32   15-;91   15->00   15->10   15->11*  15->13*
```

```
$ETABL   5-0#
$ETEND   4-319    5-0#
$FATAL   5-0#     14-3*
$FFLG    14-3     14-3#    14-3*    14-3*    14-3*
$FILLC   5-0#     14-103   14-103   14-103
$FILLS   5-0#     14-103   14-103
$GDADR   5-0#     14-:55*  14-;19*  22-13
$GDDAT   5-0#     14-:57*  14-;21*  22-13
$GET42   14-1#
$GTSWR   14-109#  14-115   14-115
$HD      4-45     4-45     4-45
$HIBTS   4-319#
$HINUM   10-21*   11-319   11-512   13-294*  14-123   14-123   14-123#  14-123*  14-;03   14-;09*  14-;12*  14-;56   14-;72
$ICNT    5-0#     10-74*   14-111   14-111   14-111   14-111*  14-111*
$INTAG   5-0#     14-109   14-109   14-109   14-109   14-109*
$ITEMB   5-0#     14-5     14-5     14-5     14-5*    14-26
$LF      5-0#     14-5     14-5     14-103   14-103   14-109   14-109   14-109
$LFLG    14-3#    14-3*
$LONUM   10-21*   13-295*  14-123   14-123   14-123#  14-123*  14-616   14-;04   14-;08*  14-;11*  14-;55   14-;70   14-;71
$LPADR   5-0#     10-21*   11-58*   11-80*   11-99*   11-131*  11-160*  11-229*  11-264*  11-313*  11-382*  11-529*  12-36*   12-60*
         12-161*  12-175*  12-235*  12-249*  12-312*  12-326*  13-26*   13-83*   13-161*  13-290*  13-319*  13-378*  14-111   14-111
         14-111   14-111*  14-111*
$LPERR   5-0#     10-21*   11-44*   11-67*   11-97*   11-107*  11-114*  11-129*  11-139*  11-142*  11-158*  11-173*  11-194*  11-227*
         11-238*  11-262*  11-274*  11-304*  11-328*  11-330*  11-380*  11-438*  11-439*  11-527*  12-29*   12-154*  12-228*  12-305*
         13-20*   13-34*   13-38*   13-40*   13-47*   13-53*   13-77*   13-92*   13-102*  13-120*  13-126*  13-156*  13-235*  13-243*
         13-252*  13-288*  13-304*  13-319*  13-329*  13-335*  13-346*  13-352*  13-362*  13-373*  14-5     14-111   14-111   14-111
         14-111*  14-539   14-656
$MADR1   5-0#
$MADR2   5-0#
$MADR3   5-0#
$MADR4   5-0#
$MAIL    4-319    4-319    5-0#     10-21    10-22    10-66    11-44    11-67    11-97    11-129   11-158   11-227   11-262   11-304
         11-380   11-527   12-29    12-154   12-228   12-305   13-20    13-77    13-156   13-288   13-373   14-5     14-103   14-111
$MAMS1   5-0#
$MAMS2   5-0#
$MAMS3   5-0#
$MAMS4   5-0#
$MBADR   4-319#
$MFLG    14-3     14-3#    14-3*    14-3*
$MNEW    14-109   14-109#
$MSGAD   5-0#     14-3     14-3*
$MSGLG   5-0#     14-3*
$MSGTY   5-0#     14-3     14-3     14-3*    14-3*
$MSWR    14-109   14-109#
$MTYP1   5-0#
$MTYP2   5-0#
$MTYP3   5-0#
$MTYP4   5-0#
$MXCNT   14-111   14-111   14-111   14-111#
$NULL    5-0#     14-103   14-103   14-103
$NWTST   11-44    11-44    11-44#   11-44#   11-67    11-67    11-67#   11-67#   11-97    11-97    11-97#   11-97#   11-129   11-129
         11-129#  11-129#  11-158   11-158   11-158#  11-158#  11-227   11-227   11-227#  11-227#  11-262   11-262   11-262#  11-262#
         11-304   11-304   11-304#  11-304#  11-380   11-380   11-380#  11-380#  11-527   11-527   11-527#  11-527#  12-29    12-29
         12-29#   12-29#   12-154   12-154   12-154#  12-154#  12-228   12-228   12-228#  12-228#  12-305   12-305   12-305#  12-305#
         13-20    13-20    13-20#   13-20#   13-77    13-77    13-77#   13-77#   13-156   13-156   13-156#  13-156#  13-288   13-288
         13-288#  13-288#  13-373   13-373   13-373#  13-373#
```

```
.$X       4-319    4-319#
A16       4-88#
A17       4-89#
ABASE     5-0      5-0
ACCESS    6-0      18-40#
ACDW1     5-0
ACDW2     5-0
ACPUOP    5-0      5-0
ACTDRV    15-92#   15-361*  15-412*  15-738*  15-747*  15-:09
ACTSTR    15-98#   15-:11*  15-:25*
ADDW0     5-0
ADDW1     5-0
ADDW10    5-0
ADDW11    5-0
ADDW12    5-0
ADDW13    5-0
ADDW14    5-0
ADDW15    5-0
ADDW2     5-0
ADDW3     5-0
ADDW4     5-0
ADDW5     5-0
ADDW6     5-0
ADDW7     5-0
ADDW8     5-0
ADDW9     5-0
ADEVCT    5-0      5-0
ADEVM     5-0      5-0
AENV      5-0      5-0
AENVM     5-0      5-0
AFATAL    5-0      5-0
AMADR1    5-0      5-0
AMADR2    5-0      5-0
AMADR3    5-0      5-0
AMADR4    5-0      5-0
AMAMS1    5-0      5-0
AMAMS2    5-0      5-0
AMAMS3    5-0      5-0
AMAMS4    5-0      5-0
AMSGAD    5-0      5-0
AMSGLG    5-0      5-0
AMSGTY    5-0      5-0
AMTYP1    5-0      5-0
AMTYP2    5-0      5-0
AMTYP3    5-0      5-0
AMTYP4    5-0      5-0
AOE       4-160#
APASS     5-0      5-0
APRIOR    5-0
APTCSU    14-3#    14-103
APTENV    14-3     14-3#    14-5     14-103
APTSIZ    10-21    14-3#
APTSPO    14-3     14-3#    14-103
ASWREG    5-0      5-0
AT0       4-208#
AT1       4-209#
```

K 16

C.SWR 6-0# 10-139 10-142 10-163 10-221 11-170 12-64 12-124 13-300 14-192 14-202 14-212 14-243 14-325

SEQ 0205

DF21 7-190 23-49#

B 1

SEQ 0207

DTE 4-163#

F 1

SEQ 0211

CROSS REFERENCE TABLE (CREF V01-05) SEQ 0212

```
DTG       4-193#
DTO       4-202#
DTUW      11-465*  15-146#  15-230   15-452   15-508*  15-639*  15-652   15-666   15-668*  15-677*  15-741   15-753*  15-:56   15-:66*
          15-;33
DULP      15-279#  15-291
DVA       4-133#
EBL       4-183#
ECH       4-157#
ECI       4-254#
ECRC      4-179#
EECC      4-174#
EM1       7-11     19-3#
EM10      7-80     19-8#
EM11      7-91     19-9#
EM12      7-105    19-10#
EM13      7-119    7-142    7-187    19-11#
EM17      7-162    19-12#
EM2       7-22     19-4#
EM20      7-173    19-13#
EM23      7-209    19-14#
EM24      7-222    19-15#
EM3       7-33     19-5#
EM4       7-44     19-6#
EM41      7-255    7-266    7-279    7-295    7-311    19-16#
EM46      7-327    19-17#
EM5       7-55     19-7#
EMPTYQ    15-631   15-682   15-783   15-:98   15-=38#
EMTVEC    4-80#    10-21*   10-21*
ERINDX    14-349   14-376   14-414   14-458   14-486   14-510   14-563#  14-685
ERR       4-146#   11-438   12-98    12-107   12-189   12-261   12-273   12-338   12-350
ERR.CT    6-0#     10-251   13-20    13-77    13-156   13-288   14-521   14-971   14-:01   14-:73   14-;32
ERROR     4-80#
ERRVEC    4-80#    10-21    10-21*   10-21*   14-111   14-111*  14-111*  14-111*  14-146*  14-147*  14-152   14-152*  14-172   16-32
          16-33*   16-35*   16-41*
ES.SAV    15-817   15-868   15-870   15-;32   15-;91   15->09#
ESRC      4-181#
EXIT.A    12-138   12-139#
EXIT0     11-58    11-58#
EXIT1     11-80    11-84#
EXIT10    11-438   11-512   11-512   11-514#
EXIT11    11-539   11-550#
EXIT12    12-35    12-60    12-60    12-60    12-138   12-143#
EXIT13    12-160   12-175   12-175   12-175   12-217#
EXIT14    12-234   12-249   12-249   12-249   12-294#
EXIT15    12-311   12-326   12-326   12-326   12-371#
EXIT16    13-26    13-63#
EXIT17    13-83    13-134#
EXIT2     11-105   11-120#
EXIT20    13-219   13-219   13-219   13-261#
EXIT21    13-293   13-364#
EXIT22    13-390#
EXIT3     11-137   11-148#
EXIT4     11-170   11-214#
EXIT5     11-235   11-248   11-251#
EXIT6     11-270   11-278   11-293#
EXIT7     11-313   11-354   11-356#
```

F1 4-128#

H 1

SEQ 0213

```
F2        4-129#
F3        4-130#
F4        4-131#
F5        4-132#
FC        8-7#     11-74    11-106   11-118   11-138   11-171   11-180   11-212   11-236   11-249   11-271   11-315   11-316   11-323
          11-386   11-387   11-388   11-512   11-534   11-535   11-546   12-60    12-60    12-175   12-177   12-202   12-249   12-268
          12-326   12-345   13-29    13-87    13-219   13-296   13-385   14-;87   14-<05
FER       4-155#
FILBUF    13-28    13-85    14-908#
FILRAN    13-302   14-:88#
FMT16     4-255#
FS        8-13#    11-47    11-70    11-100   11-132   11-161   11-230   11-265   11-530   11-531   12-60    13-88    13-165   13-182
          13-190
FT        8-10#    11-48    11-72    11-101   11-133   11-162   11-231   11-266   11-306   11-355   11-385   11-532   11-533   12-60
          13-30    13-219
FWD       18-42#
GETADR    10-69    16-13#
GETNUM    14-<28   14-<50#  16-19    16-26
GETREG    4-298#
GETREQ    15-426   15-654   15-774   15-841   15-875   15-939   15-989   15-:59   15-:87   15-:94   15-=74#
GETSWR    10-75    14-<23#
GNS       4-308    4-308    10-22    10-43    10-54    10-60    10-61    14-1     14-1     14-1     14-1     14-47    14-61    14-115
          14-115   14-115   14-115   14-115   14-115   14-115   14-115   14-115   14-115   14-115   14-115   14-115   14-115   14-115
          14-115   14-115   14-115   14-115   14-115   14-115   14-115   14-<25   14-<90   14-=06   14->72   14-?51
GO        4-127#
GT.PR1    14-<89#  14-<93   14-<93   14-<93   14-<93   14-<93   14-<95   14-<95
GT.PR2    14-=03#  14-=14   14->37   14->46   14-@04
GT.PRM    10-144   14-<88#
GTSWR     10-22    10-66    14-115#
GTTST1    14-=01   14-=06#  14-=54   14-=67   14-=81   14-=84   14->10   14->17   14->26   14->32   14->41   14->61
GTTST2    14-=19#  14->21
GTTST3    14-=22   14-=26   14-=30   14-=34   14-=38   14->03#
GTTST4    14-=40   14-=64   14->01   14->04#
GTTST5    14->05   14->27#
GTTST6    14->29   14->38#
HCE       4-158#
HCI       4-253#
HCRC      4-159#
HT        4-80#    14-103   14-103
IAE       4-161#
IC        8-9#     11-110   11-117   11-141   11-190   11-211   11-243   11-244   11-291   11-389   11-476   11-512   11-512   11-543
          11-547   14-882
IE        4-86#
ILF       4-151#
ILR       4-152#
INCCYL    13-219   14-880#
INCSK     11-108#  11-112
INCTRK    13-258   14-863#
IOTVEC    4-80#    10-21*   10-21*
IR        4-109#
ISR       11-438   12-123   12-213   12-290   12-367   14-671   15-239   15-738#
IT        8-12#    14-865
ITEM41    7-247#   14-33
LA        15-456   15-699#
LACNT     15-113#  15-707*  15-708   15-729*
LC        8-8#     11-49    11-75    11-111   11-113   11-146   11-191   11-193   11-201   11-237   11-245   11-272   11-316   11-321
```

11-477 11-512 11-512 11-544 11-548 12-197 12-256 J 1 12-333 13-223 13-310 13-380 14-880 14-883 14-885
SEQ 0215

```
        14-<00
LDCMD   10-218  14-325#
LF       4-80#  14-1     14-103   14-103   18-34    18-38    18-39    18-40    18-41
LKS      6-0#   14-184   14-216*
LKV      6-0#   11-438   11-438   14-214*  14-215*
LODEV   10-112  18-26#
LODFLT  10-143  10-146   14-229#
LODPRM  11-44   11-67    11-97    11-129   11-158   11-227   11-262   11-304   11-380   11-527   12-29    12-154   12-228   12-305
        13-20   13-77    13-156   13-288   13-373   14-279#
LOP.CK  14-357  14-387   14-425   14-470   14-520   14-537#
LP.AVL  10-72   14-145#
LPB      6-0#   14-5
LPS      6-0#   14-5     14-148
LPTAVL   6-0#   14-5     14-145*  14-149*
LS       8-14#  11-71    13-164
LSIT     4-171#
LST      4-142#
LSTRK    6-0#   13-98    13-109   13-129   13-132   13-219   13-225   13-307   14-300*  14-312*  14-;82
LT       8-11#  11-73    11-351   14-863   14-866   14-868
MABOVE  14-826  18-50#
MBELOW  14-816  18-49#
MCLK     4-200#
MCPE     4-91#
MCPEMX  15-164#  15-;15   15-;61
MDF      4-195#
MES.FT   8-52   18-10#
MES.IT   8-54   18-12#
MES.LT   8-53   18-11#
MI       4-191#
MNDLTA  15-179#  15-727
MOC      4-197#
MOH      4-232#
MOL      4-144#
MPE      4-111#
MRD      4-199#
MRMCS1  16-17   16-50#
MRMVEC  16-24   16-51#
MS       4-194#
MSC      4-190#
MSEN     4-201#
MSER     4-196#
MSG.CS  14-<26  18-17#
MSG.EQ  14-?00  14-?27   14-?53   18-16#
MSG.FC   8-46   18-4#
MSG.FS   8-55   18-13#
MSG.FT   8-49   18-7#
MSG.IC   8-48   18-6#
MSG.IT   8-51   18-9#
MSG.LC   8-47   18-5#
MSG.LS   8-56   18-14#
MSG.LT   8-50   18-8#
MSG.PA  14-?26  18-15#
MSG.R    8-45   18-3#
MSGOUS  14-771  14-776   14-811   14-318   14-821   14-828   14-838   18-48#
MSG10X   6-0    18-60#
MSG11X   6-0    18-61#
```

MSG12X 6-0 18-62#

L 1

SEQ 0217

```
MSG7XA   6-0       6-0       18-58#
MSG7XB   6-0       6-0       18-59#
MSGAVG   14-829    18-47#
MSGB01   6-0       20-15#
MSGB02   6-0       20-14#
MSGB03   6-0       20-13#
MSGB04   6-0       20-12#
MSGB05   6-0       20-11#
MSGB06   6-0       20-10#
MSGB08   6-0       20-9#
MSGB09   6-0       20-8#
MSGB10   6-0       20-7#
MSGB11   6-0       20-6#
MSGB12   6-0       20-5#
MSGB13   6-0       20-4#
MSGB14   6-0       20-3#
MSGMAX   14-772    14-819    18-46#
MSGMIN   14-768    14-809    18-45#
MSGNON   14-853    18-53#
MSGNUM   14-845    18-52#
MSGSEA   14-847    18-51#
MSGTST   11-45     11-68     11-98     11-130    11-159    11-228    11-263    11-305    11-381    11-528    12-30     12-155    12-229    12-306
         13-21     13-78     13-157    13-289    13-374    18-36#
MUR       4-198#
MWP       4-192#
MXDLTA   15-176#   15-725
MXF       4-112#
MXLACT   15-173#   15-708
MXSEEK    6-0      18-41#
MXSTAL    6-0#     11-187    11-208
MXWNDW   15-182#
NBA       4-234#
NC1       8-19#    11-388*   11-476*   11-477    11-512    11-512
NC2       8-20#
NED       4-115#
NEM       4-114#
NOCLOK   10-190    18-33#
NONE     10-172    18-31#
NOOP      4-281#
NOTPRS   10-99     18-23#
NOTRM    10-96     18-25#
NOTSAF   10-105    18-24#
OBCK      4-204#
OBEN      4-203#
OCC       4-185#
OFD       4-252#
OFFSET    4-287#
OM        4-137#
ONECYL    6-0      18-39#
OPE       4-269#
OPI       4-164#
OPN.1    14->55#   14->71    14-?95
OPN.2    14->53    14->59    14->63#
OPN.CT   14->52#   14->55*   14->58    14->63    14->65    14->73
OPN.N1   14-?19    14-?41    14-?92#
OPN.N2   14-?11    14-?34    14-?76    14-?93#
```

OPN.X1 14-?19 14-?41 14-?97#

N 1

SEQ 0219

```
OPN.X2  14-?16   14-?38   14-?80   14-?98#
OPNFLG   6-0#    14-=17*  14-=18*  14->13*  14->22   14->24   14->33   14->35   14->42   14->44   14->70
OPNPAT  14->95   14-?25#  14-?32   14-?36   14-?39   14-?41
OPNPRM  14->81   14->87#  14->91   14-?32   14-?32   14-?44
OPNTST  14->23   14->25   14->34   14->36   14->43   14->45   14->50#
OPNWDS  14-?49#  14-?65
OPT     15-387   15-422#  15-777   15-963   15-:01
OR       4-110#
PACK     4-290#
PAR      4-154#
PAT      8-15#   13-219   14-294
PAT.PT   9-3#    14-:16   14-:36   14-:50
PAT0     9-3      9-8#    14-239   14-?50   14-?84
PAT1     9-6      9-25#
PAT10    9-6      9-178#
PAT11    9-6      9-195#
PAT12    9-6      9-212#
PAT13    9-6      9-229#
PAT14    9-6      9-246#
PAT15    9-6      9-263#
PAT2     9-6      9-42#
PAT3     9-6      9-59#
PAT4     9-6      9-76#
PAT5     9-6      9-93#
PAT6     9-6      9-110#
PAT7     9-6      9-127#
PAT8     9-6      9-144#   14-238
PAT9     9-6      9-161#   14-241
PDA      4-178#
PGE      4-113#
PGM      4-141#
PHA      4-177#
PIP      4-145#
PIRQ     4-80#
PIRQVE   4-80#
PKB      6-0#    12-93*   12-102*  12-183*  12-255*  12-267*  12-332*  12-344*  14-206*
PKC      6-0#    14-726   14-728   14-729   14-733   14-735   14-736   14-740   14-745
PKCS     6-0#    12-94*   12-97*   12-104*  12-106*  12-115*  12-186*  12-188*  12-205*  12-258*  12-260*  12-270*  12-272*  12-282*
        12-335*  12-337*  12-347*  12-349*  12-359*  14-178   14-207*
PKV      6-0#    11-438   11-438   12-91*   12-181*  12-253*  12-330*  14-204*  14-205*
PLFS     4-180#
POPQUE  15-435   15-487   15-603   15-924   15-953   15-99?   15-=80#
PR0      4-80#
PR1      4-80#
PR2      4-80#
PR3      4-80#
PR4      4-80#
PR5      4-80#
PR6      4-80#
PR7      4-80#   10-84    11-438   11-438
PRM      8-4#    14-284   14-288*  14-302*  14-305*  14-308*  14->77   14->87   14-@12
PRM0     8-23     8-84#   14-233
PRM1     8-26     8-92#
PRM10    8-26     8-165#
PRM11    8-26     8-174#
PRM12    8-26     8-184#
```

```
PRM14     8-26      8-198#
PRM15     8-26      8-204#
PRM16     8-26      8-210#
PRM17     8-26      8-217#
PRM2      8-26      8-105#
PRM20     8-26      8-223#
PRM21     8-26      8-239#
PRM22     8-26      8-245#
PRM3      8-26      8-115#
PRM4      8-26      8-125#
PRM5      8-26      8-135#
PRM6      8-26      8-145#
PRM7      8-26      8-155#
PRMLMT    8-30#     11-341    11-450    13-61     13-168    13-181    14-245*   14-246*   14-249*   14-250*   14-266    14-268    14-300    14-312
          14-898    14-916    14-967    14-:89    14-;75    14->94    14-?19
PRMMSG    8-45#     14->97
PRMPT     8-23#     14-252    14-283    14->75    14-@13
PS        4-80      4-80#     10-73*    10-84*    10-134*   11-438*   11-438*   11-438*   11-471*   12-116*   12-206*   12-283*   12-360*   14-670*
          15-222    15-223*   15-252*   15-260*   15-359    15-360*   15-413*   15-423    15-472*   15-714*
PSEL      4-90#
PSW       4-80#
PTRN15    8-236#
PWRVEC    4-80#
QCNT      15-<55#   15-=14    15-=38*   15-=53    15-=55*   15-=75    15-=89*
QDRV0     15-<66    15-<77    15-<86    15-<98#
QDRV1     15-<67    15-<78    15-<87    15-<99#
QDRV2     15-<68    15-<79    15-<88    15-=00#
QDRV3     15-<69    15-<80    15-<89    15-=01#
QDRV4     15-<70    15-<81    15-<90    15-=02#
QDRV5     15-<71    15-<82    15-<91    15-=03#
QDRV6     15-<72    15-<83    15-<92    15-=04#
QDRV7     15-<73    15-<84    15-<93    15-=05#
QINPT     15-<66#   15-=40    15-=57*   15-=58*   15-=59    15-=61*
QOUTPT    15-<77#   15-=40*   15-=78    15-=91    15-=92*   15-=93*   15-=94    15-=96*
QSTART    15-<86#   15-=20    15-=25    15-=61    15-=96
QSTOP     15-<87#   15-=59    15-=94
QTERM     15-<94    15-=06#
R6         4-80#    10-21     10-21*    10-21*
R7         4-80#
RANADR    13-321    13-332    13-349    14-;69#
RANCK     13-339    13-356    14-;03#
RANPAT    13-327    14-:90    14-;10    14-;51#
RD.ADR    15-;17*   15-;18*   15-;20#   22-5
RD.RM     15-292    15-322    15-328    15-551    15-575    15-589    15-701    15-715    15-761    15-811    15-878    15-888    15-903    15-:45
          15-;15#   15-;68    15-;82    15-;88    15-<21
RD.RM1    15-;19#   15-;47
RD.RM2    15-;15*   15-;46#
RD.RM3    15-;26    15-;38    15-;48#
RD.RM4    15-;31    15-;50#
RD.WRD    15-;21#   15-;22    22-5      22-6
RDCHR     14-109    14-115#
RDLIN     14-115#   14-<55    14-<91    14-=07    14-?03    14-?30    14-?55
RDY        4-87#
READ       4-296#   13-42     13-253    13-333    13-350
READHD     4-297#   14-327    14-328    14-389    14-427
READIN     4-289#
```

RECAL 4-284# 10-227 10-227 11-46 11-46 14-651 14-676 E 2

SEQ 0223

```
RELEAS   4-286#
RESREG   12-100  12-109  12-119  12-191  12-209  12-263  12-275  12-286  12-340  12-352  12-363  14-100  14-115#  14-119
         14-709  14-918  14-941  14-975  14-:22  14-:44  14-=03  14->60  14-?75  14-?79  14-@03  14-@29  15-261   15-407
         15-410  15-473  15-686  15-746  15-:24  15-<30  15-=29
RESTAR   10-202#  10-213  14-1
RESVEC   4-80#
REV      6-0      6-0     6-0     18-43#
REX      4-182#
RH.ADR   6-0#     16-16   16-34   16-36   22-8
RHVEC    6-0#
RM.REG   10-234  14-351  14-353  14-378  14-380  14-383  14-416  14-418  14-421  14-460  14-462  14-473  14-488  14-490
         14-512  14-514  14-588  14-590  14-895  17-15   17-39   17-63   17-87   17-101#  22-14   22-14   22-14   22-14
         22-14   22-17   22-17   22-17   22-18   22-18   22-18   22-18   22-18   22-20   22-20   22-20   22-21   22-21
         22-21   22-22   22-22   22-22   22-24   22-24   22-24   22-24   22-24   22-24   22-25   22-25   22-25   22-27
         22-27   22-27   22-27   22-27   22-27   22-28   22-28   22-28   22-28   22-28   22-28
RM05     14-342  14-369  14-407  14-449  14-479  14-496  14-502  14-678  15-359#
RMADR    11-308  11-396  14-133  14-672  15-169#  15-241  15-366  15-490  15-510  15-532  15-699  15-740  15-:43  15-;17
         15-;23  15-;27  15-;41  15-;74  15-;78  15-<10  15-<40  16-37
RMAS     15-193#  15-327*  15-809  15-928*  15-958*  15-979*  15-:68
RMBA     11-461*  15-188#
RMCS1    11-438*  11-459*  11-470*  12-257*  12-269*  12-334*  12-346*  15-186#  15-283*  15-290  15-316  15-430*  15-431  15-506
         15-528  15-570  15-590  15-601  15-616  15-629  15-762  15-812  15-865  15-895  15-934  15-995  15-;69
RMCS2    11-331*  11-438*  11-438*  11-438*  11-438*  11-440*  11-512*  11-512*  14-134*  14-135*  14-673*  15-190#  15-242*  15-282*
         15-284  15-379*  15-429*  15-491*  15-511*  15-533*  15-619  15-642  15-657  15-671  15-680*  15-700*  15-760*  15-876*
         15-965*  15-978*  15-:44*  15-;24  15-;79  15-<11*  15-<17  15-<41*  15-<46
RMDA     11-462*  12-60*  14-688*  14-895  15-189#  15-498  15-524  15-539
RMDB     15-195#  15-<15
RMDC     11-438*  12-60*  12-184*  12-256*  12-268*  12-333*  12-345*  14-689*  15-200#  15-502  15-514  15-546
RMDS     11-438  11-438  12-98  12-107  12-189  12-261  12-273  12-338  12-350  15-191#  15-323  15-702  15-879  15-904
         15-966  15-:46
RMDT     15-197#  15-293
RMEC1    15-204#
RMEC2    15-205#  15-<25
RMER1    14-351  14-378  14-380  14-416  14-418  14-460  14-488  14-512  14-588  15-192#  15-329  15-889  15-967  15-;83
RMER2    14-353  14-383  14-421  14-462  14-490  14-514  14-590  15-203#  15-968
RMERRS   15-22#  15-225  15-847  15-850  15-860  15-966*  15-967*  15-968*  15-969*  22-4  22-4  22-4  22-4  22-7
         22-7  22-7  22-7
RMHR     15-201#
RMINIT   10-78  15-221#
RMLA     11-332  11-441  15-194#  15-716  15-718
RMMR1    15-196#
RMMR2    15-202#  15-969
RMOF     15-199#  15-320  15-552  15-556  15-576  15-580
RMR      4-153#
RMSN     15-198#
RMTMR    14-220  15-:09#
RMVEC    11-438*  11-438*  12-92*  12-123*  12-182*  12-213*  12-254*  12-290*  12-331*  12-367*  14-671*  15-170#  15-238  15-240
         15-360
RMWC     11-460*  15-187#
ROTATE   6-0      6-0     6-0     6-0     18-38#
RPT      8-6#  11-44  11-67  11-97  11-129  11-158  11-227  11-262  11-304  11-380  11-527  12-29  12-154  12-228
         12-305  13-20  13-77  13-156  13-288  13-373  14-?01  14-?20*  14-?28  14-?42*  14-?92*  14-?97*
RSTRT1   10-187  10-189  10-192#  10-205
RSTRT2   10-199#  10-206
RTC      4-288#
SAVCSW   6-0#  10-138  10-142*  14-<31*
```

SAVEFG 10-79* 15-124# 15-605 15-766 15-955

SEQ 0225

```
STALL   11-178   11-184   11-199   11-205   13-383   13-388   14-396   14-434   14-526   14-612#
STALL0   6-0#    11-172*  11-179   11-185   11-186*  11-187   11-189*  11-200   11-206   11-207*  11-208   11-210*
STALL1   6-0#    14-397   14-435
STALL2   6-0#    14-527
STALL3   6-0#    13-384   13-389
START   10-8     10-16#
START1   4-308   10-6#
START2   4-310   10-13#   10-194   14-109   14-109
START3   4-312   10-3#
START4   4-314   10-10#
STATBL   6-0#    14-45
STKLMT   4-80#
STO     15-:19   15-:39#
STO1    15-:51#
STO2    15-:50   15-:52   15-:54   15-:68#
STO3    15-:70   15-:76#
STO5    15-:82#
STO6    15-:72   15-:84#
STO7    15-:74   15-:92#
STO8    15-:91   15-:98#
STO9    15-:48   15-:58   15-:67   15-:75   15-:79   15-:81   15-:83   15-:89   15-:96   15-;00#
STRT1A  10-4     10-7#
STRT2A  10-11    10-14#
STRTMR  12-90    12-180   12-252   12-329   14-701#
SVADR    6-0#    13-322*  13-323*  13-341   13-342   13-358   13-359
SVRH70  11-438   11-512   12-100   12-109   12-119   12-191   12-209   12-263   12-275   12-286   12-340   12-352   12-363   15-607
        15-768   15-784   15-842   15-892   15-910   15-957   15-991   15-:61   15-:99   15-<07#
SVSTAT   6-0#    14-42    14-566*
SW0      4-80#
SW00     4-80     4-80#    13-248
SW01     4-80     4-80#    13-255   14-999   14-:71   14-;34
SW02     4-80     4-80#    13-250
SW03     4-80     4-80#    13-230   13-241
SW04     4-80     4-80#    13-230   13-233   14-475
SW05     4-80     4-80#    14-202   14-212   14-:03
SW06     4-80     4-80#    12-64    12-124   14-192   14-759   14-798
SW07     4-80     4-80#    14-325   14-<23
SW08     4-80     4-80#    10-163   14-5
SW09     4-80     4-80#    14-997   14-;30
SW1      4-80#
SW10     4-80#
SW11     4-80#
SW12     4-80#    11-45    11-68    11-98    11-130   11-159   11-170   11-228   11-263   11-305   11-381   11-528   12-30    12-155
        12-229   12-306   13-21    13-78    13-157   13-289   13-374
SW13     4-80#    14-613
SW14     4-80#    14-394   14-432   14-524
SW15     4-80#    13-300
SW2      4-80#
SW3      4-80#
SW4      4-80#
SW5      4-80#
SW6      4-80#
SW7      4-80#
SW8      4-80#
SW9      4-80#    14-537   14-654
SWR      5-0#     10-21    10-21    10-21*   10-21*   10-21*   10-22    10-66    11-45    11-68    11-98    11-130   11-159   11-228
```

11-263 11-305 11-381 11-528 12-30 12-155 12-229 K 2 12-306 13-21 13-78 13-157 13-228 13-233 13-241
SEQ 0229

```
        13-248   13-250   13-255   13-289   13-374   14-5     14-5     14-5     14-5     14-5     14-109   14-109   14-109*  14-111
        14-111   14-111   14-475   14-537   14-654   14-759   14-798   14-997   14-999   14-:03   14-:71   14-;30   14-;34   14-<23
SWREG    4-308#  10-21    10-22    10-66    14-109   14-109   14-109
T10      6-0#    12-175   12-214
T11      6-0#    12-249   12-29?
T12      6-0#    12-326   12-3?8
T7A      6-0#    12-80    12-81    12-138
T7A1     6-0#    12-75    12-77
T7B      6-0#    12-71    12-72    12-140
T7B1     6-0#    12-66    12-67
TAB.XY   4-320#   5-0
TAP      4-233#
TBITVE   4-80#
TD      15-743   15-752#
TEST0   11-44    11-58    11-58#
TEST1   11-67    11-80    11-80#
TEST10  11-380   11-438#  11-512   11-512
TEST11  11-527   11-539#
TEST12  12-87#
TEST13  12-175#
TEST14  12-249#
TEST15  12-326#
TEST16  13-20    13-26    13-26#
TEST17  13-77    13-83    13-83#
TEST2   11-97    11-105#
TEST20  13-156   13-219#
TEST21  13-288   13-319   13-319   13-319#
TEST22  13-373   13-378   13-378#
TEST3   11-129   11-137#
TEST4   11-158   11-170   11-170#
TEST5   11-227   11-235#
TEST6   11-262   11-270#
TEST7   11-304   11-313   11-313#
TICKMS   6-0#    14-194*  14-197*  14-219
TICKUS   6-0#    14-195*  14-198*
TIM.DN   6-0#    14-708*  14-725
TIM.PT   6-0#    14-704   14-743   14-745*  14-746*
TIM.UP   6-0#    14-702   14-707*  14-722   14-804
TIMER   11-468*  15-137#  15-343*  15-466*  15-610*  15-663*  15-755*  15-873*  15-918*  15-961*  15-976*  15-998*  15-:15   15-:17*
        15-:80*  15-:86*  15-:92*
TKVEC    4-80#   14-109*  14-109*
TP50    12-67*   12-72*   12-140#
TP60    12-75*   12-80*   12-138#
TPB      6-0#    10-29    14-5
TPS      6-0#    10-28    14-5
TPS50   12-68*   12-73*   12-142#
TPS60   12-76*   12-82*   12-138#
TPVEC    4-80#
TRAPVE   4-80#   10-21*   10-21*
TRCKWC   6-0#    13-32    13-55    13-297   14-247*  14-251*
TRE      4-92#
TRK.DS   6-0#    14-349*  14-376*  14-414*  14-458*  14-486*  14-510*  14-646*  14-685*  14-990*  14-:61*  22-10    22-11    22-12
        22-15    22-23    22-26
TRK.RD   6-0#    14-643*  22-11
TRNSWT  11-464*  15-78#   15-488*  15-635   15-638*  15-651   15-669*  15-678*  15-757   15-758*  15-:55   15-:65*
TRTVEC   4-80#
```

```
TST1     11-44    11-67#
TST10    11-304   11-380#
TST10A   11-438   11-438#
TST10B   11-438   11-512#
TST11    11-380   11-527#
TST12    11-527   12-29#
TST13    12-29    12-29    12-154#
TST14    12-154   12-154   12-228#
TST15    12-228   12-228   12-305#
TST16    12-305   12-305   13-20#
TST17    13-20    13-77#
TST2     11-67    11-97#
TST20    13-77    13-156#
TST21    13-156   13-288#
TST22    13-288   13-373#
TST3     11-97    11-129#
TST4     11-129   11-158#
TST5     11-158   11-227#
TST6     11-227   11-262#
TST7     11-262   11-304#
TSTDRV   10-229   18-34#
TSTNMS    6-0#    10-188   11-44    11-67    11-97    11-129   11-158   11-227   11-262   11-304   11-380   11-527   12-29    12-154
         12-228   12-305   13-20    13-77    13-156   13-288   13-373   14-230*  14-231*  14-<99*  14-=00*  14-=15*  14-=16*  14-=21*
         14-=25*  14-=29*  14-=33*  14-=37*  14-=62*  14-=96*  14->06   14->08
TYPDS    14-81    14-115#
TYPE     10-22    10-43    10-49    10-54    10-60    10-61    10-71    10-86    10-87    10-89    10-96    10-99    10-102   10-105
         10-112   10-114   10-115   10-126   10-132   10-167   10-172   10-181   10-190   10-229   10-231   10-232   10-247   10-250
         11-45    11-68    11-98    11-130   11-159   11-228   11-263   11-305   11-381   11-528   12-30    12-155   12-229   12-306
         13-21    13-78    13-157   13-289   13-374   14-1     14-1     14-1     14-1     14-1     14-5     14-5     14-37    14-38
         14-47    14-53    14-54    14-58    14-61    14-64    14-65    14-74    14-84    14-88    14-91    14-93    14-95    14-98
         14-103   14-105   14-107   14-109   14-109   14-109   14-109   14-109   14-109   14-109   14-109   14-109   14-109   14-109
         14-109   14-109   14-109   14-109   14-109   14-109   14-109   14-115#  14-121   14-761   14-762   14-764   14-768   14-771
         14-772   14-776   14-777   14-805   14-809   14-811   14-814   14-816   14-818   14-819   14-821   14-824   14-826   14-828
         14-829   14-838   14-839   14-845   14-847   14-853   14-<25   14-<51   14-<54   14-<90   14-=06   14->72   14->74   14->96
         14->98   14-?00   14-?02   14-?25   14-?26   14-?27   14-?29   14-?51   14-?53
TYPERR   14-5     14-17#
TYPOC    14-1     14-79    14-109   14-115#  14-<53   14-?28   14-?54
TYPON    14-115#
TYPOS    10-46    10-57    10-88    10-178   10-230   11-45    11-68    11-98    11-130   11-159   11-228   11-263   11-305   11-381
         11-528   12-30    12-155   12-229   12-306   13-21    13-78    13-157   13-289   13-374   14-1     14-115#  14->73
TYPTIM   12-138   12-139   12-214   12-291   12-368   14-797#
ULDFLG   15-105#  15-281*  15-369   15-384*  15-567*  15-633*  15-670*  15-839   15-925   15-927*  15-946   15-948*  15-:64*
UNLOAD    4-282#
UNS       4-165#
UNSTAT   10-86    18-20#
UNTOFF   10-102   18-21#
UNTON    10-114   18-22#
UPE       4-116#
US1       4-103#
US2       4-104#
US4       4-105#
VERIFY   14-391   14-429   14-635#
VV        4-138#
WC        4-175#
WCE       4-117#
WCEFLG    6-0#    13-219*  13-246   14-448*  14-508*
```

```
WCHKX    15-769   15-790
WD        4-173#
WLE       4-162#
WRCKD     4-292#  13-37    13-54    13-104   13-127   13-244   13-343   13-360   14-471   14-495
WRCKHD    4-293#
WRITE     4-294#  11-384   13-35    13-48    13-86    13-121   13-236   13-299   13-324   14-477
WRL       4-143#
WRT.AD   15-;64*  15-;74*  15-;77#  22-6
WRT.R1   15-;75#  15-;95
WRT.R2   15-;61*  15-;94#
WRT.R3   15-;70   15-;81   15-;84   15-;90   15-;96#
WRT.R4   15-;86   15-;98#
WRT.R5   15-;97   15-;99#
WRT.RM   15-315   15-319   15-497   15-501   15-505   15-513   15-523   15-527   15-538   15-545   15-555   15-569   15-579   15-600
         15-615   15-628   15-864   15-894   15-933   15-;61#
WRT.WD   15-;62*  15-;66   15-;73*  15-;76#  22-6
WRTHD     4-295#
XXDP      6-0#    10-34*   10-37*   10-38    10-40*   10-45    10-56    10-108   10-110   10-153   10-155
```

```
$$CMRE   4-525#  5-0     5-0     5-0     5-0     5-0     5-0
$$CMTM   4-525#  5-0     5-0     5-0
$$ESCA   4-80#
$$NEWT   4-80#   11-44   11-67   11-97   11-129  11-158  11-227  11-262  11-304  11-380  11-527  12-29   12-154  12-228
        12-305   13-20   13-77   13-156  13-288  13-373
$$SET    14-115  14-115  14-115  14-115  14-115  14-115  14-115  14-115  14-115  14-115  14-115  14-115#
$$SETM   10-21   10-21#
$$SKIP   4-80#
.$ACT1   4-40#   4-316
.$APTB   4-41#   5-0     5-0#
.$APTH   4-41#   4-319
.$APTY   4-41#   14-3
.$CATC   4-38#   4-308
.$CMTA   4-38#   4-525
.$DB2D   4-40#   14-119
.$DIV    4-39#   14-125
.$EOP    4-38#   14-1
.$ERRO   4-38#   14-5
.$RAND   4-40#   14-123
.$RDDE   4-39#
.$RDOC   4-39#
.$READ   4-39#   14-109
.$SAVE   4-40#   14-113
.$SB2D   4-40#   14-117
.$SCOP   4-40#   14-111
.$SIZE   4-40#
.$SUPR   4-40#   14-121
.$TRAP   4-38#   14-115
.$TYPD   4-39#   14-107
.$TYPE   4-38#   14-103
.$TYPO   4-39#   14-105
.EQUAT   4-38#   4-80
.HEADE   4-39#   4-45
.SETUP   4-39#   4-306
.SWRHI   4-38#   4-46    4-47
.SWRLO   4-46#   4-47#   4-48    4-49    4-50    4-51    4-52    4-53    4-54    4-55
CKCHR    9-432#  14-<57  14-<93  14-?05  14-?32  14-?57  14-A18  14-A25  14-A30  14-A75
CKDIG    9-442#  14-<95  14-?19
CKNUM    9-452#  14-<59  14-?41  14-?59
COMMEN   4-80#
COMND    9-353#  10-227  11-46   11-307  11-383  12-60   12-175  12-249  12-326  13-381  13-386
DO       9-362#  10-225  10-228  11-58   11-58   11-82   11-83   11-109  11-116  11-140  11-144  11-175  11-181  11-196
        11-202   11-240  11-242  11-280  11-282  11-284  11-286  11-288  11-290  11-329  11-349  11-541  11-542  12-60
        12-175   12-249  12-326  13-382  13-387  14-652
DODTA    9-377#  13-36   13-39   13-43   13-52   13-58   13-95   13-106  13-122  13-128  13-240  13-245  13-254  13-305
        13-330   13-336  13-347  13-353  13-363
ENDCOM   4-80#
ENDPAS   9-381#  14-1
ER.NDX   9-397#  14-349  14-376  14-414  14-458  14-486  14-510  14-685
ERRCAL  14-A89#  15-817  15-868  15-870  15-;32  15-;91
ERREND   9-425#  14-5
ERROR    4-8#    10-211  11-438  11-512  12-101  12-110  12-120  12-192  12-210  12-264  12-276  12-287  12-341  12-353
        12-364   14-349  14-349  14-349  14-349  14-355  14-376  14-376  14-376  14-376  14-385  14-414  14-414  14-414
        14-414   14-423  14-458  14-458  14-458  14-458  14-466  14-486  14-486  14-486  14-486  14-494  14-510  14-510
        14-510   14-510  14-518  14-650  14-685  14-685  14-685  14-685  14-685  14-992  14-996  14-:65  14-:70  14-;28
        14-;29   16-43
```

CROSS REFERENCE TABLE (CREF V01-05)

```
ERRTYP   9-416#  14-5
ESCAPE    4-80#  12-60   12-175  12-249  12-326  14-348  14-375  14-413  14-457  14-485  14-509  14-648  14-684  14-991
         14-995  14-:64  14-:69  14-;16
EXIT      4-16#  11-58   11-84   11-120  11-148  11-214  11-251  11-293  11-356  11-514  11-550  12-143  12-217  12-294
         12-371  13-63   13-134  13-261  13-364  13-390
GETPRI    4-80#  14-125
GETSWR    4-40#   4-80#  10-22   10-22#  10-66
LOOP      9-366#  11-107  11-114  11-139  11-142  11-173  11-194  11-238  11-274  11-328  11-330  11-438  11-439  13-34
         13-38   13-40   13-47   13-53   13-92   13-102  13-120  13-126  13-235  13-243  13-252  13-304  13-329  13-335
         13-346  13-352  13-362
MORE.S    9-327#  11-44   11-67   11-97   11-129  11-158  11-227  11-262  11-304  11-380  11-527  12-29   12-154  12-228
         12-305  13-20   13-77   13-156  13-288  13-373
MORETA    4-322#   5-0     7-2#
MSG      11-37#  11-44   11-60#  11-67   11-86#  11-97   11-122#  11-129  11-150#  11-158  11-216#  11-227  11-253#  11-262
         11-295#  11-304  11-358#  11-380  11-516#  11-527  12-13#  12-29  12-145#  12-154  12-219#  12-228  12-296#  12-305
         13-9#   13-20   13-65#  13-77   13-136#  13-156  13-263#  13-288  13-366#  13-373
MULT      4-80#
NEWTST    4-80#  11-44   11-67   11-97   11-129  11-158  11-227  11-262  11-304  11-380  11-527  12-29   12-154  12-228
         12-305  13-20   13-77   13-156  13-288  13-373
POP       4-80#  14-3    14-3    14-107  14-113  14-123  14-125  14-270  14-315
PUSH      4-80#  14-3    14-3    14-3    14-107  14-113  14-123  14-125  14-229  14-279
REPORT    4-80#   9-297#  12-214  12-291  12-368
SAV.RH    9-281#  12-100  12-109  12-119  12-191  12-209  12-263  12-275  12-286  12-340  12-352  12-363
SET.TN    9-302#  11-44   11-67   11-97   11-129  11-158  11-227  11-262  11-304  11-380  11-527  12-29   12-154  12-228
         12-305  13-20   13-77   13-156  13-288  13-373
SETPRI    4-80#  14-109
SETTRA   14-115  14-115  14-115  14-115  14-115  14-115  14-115  14-115  14-115  14-115  14-115  14-115#
SETUP     4-80#  10-21
SKIP      4-80#
SLASH     4-80#  11-1    11-25   11-29   11-35   12-3    12-11   13-3    13-7
STARS     4-80#   4-278   4-316   4-319   4-319   4-319   5-0     5-0     5-0     7-227   7-245  11-44   11-44   11-67
         11-67   11-97   11-97   11-129  11-129  11-158  11-158  11-227  11-227  11-262  11-262  11-304  11-304  11-380
         11-380  11-527  11-527  12-29   12-29   12-154  12-154  12-228  12-228  12-305  12-305  13-20   13-20   13-77
         13-77   13-156  13-156  13-288  13-288  13-373  13-373  14-1    14-3    14-5    14-103  14-105  14-107  14-109
         14-109  14-109  14-109  14-109  14-111  14-113  14-115  14-117  14-119  14-121  14-123  14-125  15-14
SWRSU     4-80#  10-21   10-21#
TRMTRP   14-115#
TSTTYP    4-22#  11-45   11-68   11-98   11-130  11-159  11-228  11-263  11-305  11-381  11-528  12-30   12-155  12-229
         12-306  13-21   13-78   13-157  13-289  13-374
TYPB2D    9-289#  14-810  14-815  14-817  14-820  14-825  14-827  14-837  14-840  14-?01  14-?52
TYPBIN    4-80#
TYPDEC    4-80#  14-81
TYPEND    9-386#  14-1
TYPNAM    4-80#  10-22
TYPNUM    4-80#
TYPOCS    4-80#  10-88   10-178  10-230  11-45   11-68   11-98   11-130  11-159  11-228  11-263  11-305  11-381  11-528
         12-30   12-155  12-229  12-306  13-21   13-78   13-157  13-289  13-374  14-1    14->73
TYPOCT    4-80#  14-1    14-79   14-109  14-<53  14-?28  14-?54
TYPTXT    4-80#  10-43   10-54   10-60   10-61   14-1    14-1    14-1    14-1    14-47   14-61   14-<25  14-<90  14-=06
         14->72  14-?51
```